Конфигуратор СП210



Руководство пользователя

10.2026
версия 1.6

Содержание

Используемые аббревиатуры и термины ... 6
1 О программе ... 7
2 Характеристики панелей оператора СП210 ... 8
3 Отличия панелей оператора СП210 от СП3хх ... 9
4 Установка и удаление конфигуратора ... 10
4.1 Установка ... 10
4.2 Удаление ... 10
5 Системное меню и экранный конфигуратор ... 11
5.1 Системное меню ... 11
5.1.1 Вкладка «Имя» ... 12
5.1.2 Вкладка «Пароли» ... 13
5.1.3 Вкладка «Сеть» ... 13
5.1.4 Вкладка «Время» ... 14
5.1.5 Вкладка «VNC» ... 14
5.1.6 Вкладка «Система» ... 15
5.2 Сервисное меню и экранный конфигуратор ... 16
5.2.1 Экранный конфигуратор ... 16
5.2.2 Обновление прошивки ... 17
6 Первый старт ... 18
6.1 Основная информация ... 18
6.2 Настройка MasterOPC Universal Modbus Server ... 18
6.3 Первый старт с панелью (часть 1) ... 22
6.4 Первый старт с эмулятором (часть 1) ... 27
6.5 Первый старт (часть 2) ... 32
6.6 Заключение ... 37
7 Память панели ... 38
7.1 Области памяти ... 38
7.2 Адреса объектов. Типы и форматы данных ... 39
7.3 Динамическая адресация ... 41
7.4 Переменные ... 42
8 Обзор интерфейса конфигуратора ... 43
8.1 Основная информация ... 43
8.2 Панель меню ... 44
8.3 Панель ярлыков ... 44
8.4 Дерево проекта ... 45
8.4.1 Системные экраны и окна ... 48
8.4.2 Библиотеки ... 50
8.4.2.1 Библиотека локализации ... 50
8.4.2.2 Библиотека переменных ... 51
8.4.2.3 Библиотека изображений ... 54
8.4.2.4 Библиотека аудиофайлов ... 55
8.5 Редактор визуализации ... 56
8.5.1 Вкладки ... 57
8.5.2 Панель команд ... 57
8.5.3 Сетка и масштаб ... 59
8.5.4 Слои ... 59
8.5.5 Контекстное меню элемента ... 60
8.6 Карта адресов ... 61
8.7 Список элементов ... 62

8.8 Панель сообщений ... 63
8.9 Строка информации ... 64
8.10 Инструменты – Настройки конфигуратора ... 65
8.11 Меню «Файл» ... 67
8.12 Меню «Правка» ... 69
8.12.1 Поиск и замена адресов ... 70
8.12.2 Поиск и замена шрифта ... 71
8.13 Настройки проекта ... 72
8.13.1 Вкладка «Общие» ... 72
8.13.2 Вкладка «Панель» ... 74
8.13.3 Вкладка «Управление» ... 75
8.13.4 Вкладка «Права доступа» ... 75
8.13.5 Вкладка «Время» ... 76
8.14 Данные PFW-регистров ... 77
9 Загрузка проекта и режим эмуляции ... 80
9.1 Загрузка проекта с ПК ... 80
9.2 Загрузка проекта с USB-накопителя ... 82
9.3 Эмуляция ... 82
9.4 Сохранение эмуляции ... 83
10 Графические элементы ... 85
10.1 Типовые настройки ... 85
10.1.1 Вкладка «Основное» ... 85
10.1.2 Вкладка «Безопасность» ... 86
10.1.3 Вкладка «Расположение» ... 87
10.2 Графика ... 89
10.2.1 Линия ... 89
10.2.2 Окружность ... 90
10.2.3 Прямоугольник ... 91
10.2.4 Дуга ... 92
10.2.5 Ломаная/Многоугольник ... 93
10.2.6 Таблица ... 94
10.2.7 Шкала ... 95
10.2.8 QR-код ... 96
10.2.9 Изображение ... 98
10.2.10 Динамическое изображение ... 100
10.2.11 Анимация перемещения ... 102
10.2.12 Холст ... 104
10.3 Текст ... 105
10.3.1 Статический текст ... 105
10.3.2 Динамический текст ... 106
10.3.3 Бегущая строка ... 107
10.4 Ввод ... 109
10.4.1 Цифровой ввод ... 109
10.4.2 Текстовый ввод ... 113
10.4.3 Ползунок ... 116
10.5 Дисплей ... 119
10.5.1 Цифровой дисплей ... 119
10.5.2 Текстовый дисплей ... 119
10.6 Переключатели/индикаторы ... 120
10.6.1 Ввод символа ... 120
10.6.2 Индикатор ... 122

10.6.3 Переключатель ... 124
10.6.4 Переключатель с индикацией ... 125
10.6.5 Многопозиционный индикатор ... 126
10.6.6 Многопозиционный переключатель ... 128
10.6.7 Функциональная кнопка ... 130
10.6.8 Функциональная область ... 136
10.6.9 Выпадающий список ... 138
10.6.10 Проводник ... 140
10.7 Приборы и инструменты ... 142
10.7.1 Время ... 142
10.7.2 Дата ... 144
10.7.3 Трубопровод ... 145
10.7.4 Аналоговый дисплей ... 147
10.7.5 Двигатель ... 150
10.7.6 Емкость ... 152
10.7.7 Зуммер ... 156
10.7.8 Подсветка ... 157
10.7.9 Вентилятор ... 158
10.7.10 Перемешиватель ... 160
10.7.11 Насос ... 162
10.7.12 Клапан ... 163
10.7.13 Таймер ... 165
10.8 Тревоги ... 168
10.8.1 Настройка тревог ... 168
10.8.2 Таблица тревог ... 171
10.8.3 Баннер тревог ... 176
10.9 Графики и таблицы ... 179
10.9.1 Выборка данных ... 179
10.9.2 Тренд ... 184
10.9.3 Таблица выборки ... 189
10.9.4 Таблица данных ... 194
10.9.5 XY график ... 197
10.9.6 XY тренд ... 202
10.9.7 Секторная диаграмма ... 205
10.10 Рецепты ... 207
10.10.1 Настройка рецептов ... 207
10.10.2 Таблица рецептов ... 208
10.10.3 Чтение/запись рецепта ... 212
10.10.4 Управление таблицей рецептов ... 215
10.11 Журнал действий ... 217
10.11.1 Настройка журнала действий ... 217
10.11.2 Отображение журнала действий ... 219
10.12 Мультимедия ... 223
10.12.1 Отображение видеопотока (RTSP) ... 223
10.12.2 Воспроизведение видеофайлов ... 225
10.12.3 Коды ошибок ... 226
11 Настройка обмена с другими устройствами ... 227
11.1 Поддерживаемые протоколы ... 227
11.2 Общий принцип настройки обмена ... 228
11.3 Настройка обмена по протоколу Modbus ... 230
11.3.1 Modbus RTU/ASCII Master ... 230

11.3.2 Modbus TCP Master ... 233
11.3.3 Modbus Master – способы настройки обмена ... 236
11.3.4 Modbus RTU Slave ... 241
11.3.5 Modbus TCP Slave ... 243
11.3.6 Modbus Slave – адресация регистров панели ... 244
11.4 Пользовательский протокол ... 244
11.5 Символьный протокол CODESYS V3 ... 244
11.6 OPC UA Client ... 248
11.7 OPC UA Server ... 249
11.8 Протоколы производителей ПЛК ... 253
11.9 Системные биты/регистры диагностики и управления обменом ... 253
11.10 MQTT ... 256
11.11 Сканер штрихкодов ... 264
12 Макросы ... 266
12.1 Основная информация ... 266
12.2 Работа с макросами в интерфейсе конфигуратора ... 266
12.3 Вызов макросов ... 269
12.4 Особенности поддержки языка Си в панелях СП210 ... 269
12.5 Базовая информация о языке C99 ... 269
12.5.1 Файлы исходного кода ... 270
12.5.2 Заголовочные файлы ... 271
12.5.3 Типы данных ... 271
12.5.4 Переменные ... 305
12.5.5 Спецификаторы и квалификаторы ... 308
12.5.6 Литералы ... 311
12.5.7 Идентификаторы ... 312
12.5.8 Комментарии ... 313
12.5.9 Операторы и выражения ... 313
12.5.10 Функции ... 315
12.5.11 Препроцессор ... 317
12.5.12 Стандартная библиотека языка Си ... 321
12.6 Встроенные функции и макросы ... 322
12.6.1 Работа с внутренними битами и регистрами панели ... 323
12.6.2 Функции для работы с памятью панели и slave-устройств ... 324
12.6.3 Функции для обмена данными с ПЛК CODESYS V3.5 ... 339
12.6.4 Функции для реализации пользовательского протокола ... 344
12.6.5 Функции для работы с элементом Холст ... 347
12.6.6 Функции для работы с экранами и окнами ... 355
12.6.7 Функции для обработки данных ... 358
12.6.8 Остальные функции ... 369
12.6.9 Добавление пользовательских макросов на вкладку Встроенные функции ... 372
12.6.10 Синхронное и асинхронное выполнение макросов ... 372
12.6.11 Особенности работы с файлами ... 374
13 Приложение ... 375
13.1 Системные биты и регистры ... 375
13.2 VNC ... 382
13.3 Контроль оплаты ... 384
13.4 Подключение устройств к порту USB A ... 386
13.5 Системное время ... 386
13.6 Редактирование шаблона проекта ... 387

Используемые аббревиатуры и термины

COM-порт – порт устройства, обеспечивающий обмен данными по последовательным интерфейсам RS-485 и/или RS-232.

Modbus – открытый промышленный протокол обмена, основанный на архитектуре master/slave.

VNC – технология удаленного доступа к экрану панели с ПК или мобильного устройства.

Адрес – идентификатор ячейки памяти панели или опрашиваемого устройства. Состоит из кода области памяти и порядкового номера ячейки.

Идентификатор (ID) панели – уникальный идентификационный номер, используемый для обнаружения панели в локальной сети.

Макрос – программа на языке Си (C99), используемая для выполнения операций, которые не могут быть реализованы с помощью стандартных элементов конфигуратора.

Переменная – символьное имя, ассоциированное с адресом ячейки памяти и соответствующим типом/форматом данных.

Проект – совокупность экранов, окон, элементов, переменных, конфигурации обмена, макросов библиотек и настроек, представленная в виде файла с расширением **.xjp** или, в случае шифрования, **.xep**. Проект компилируется (преобразуется) в бинарный исполняемый файл, который загружается в панель и формирует графический интерфейс, а также логику работы устройства.

Элемент – графический объект с заданным набором параметров, используемый для отображения и ввода значений, а также выполнения других функций.

Эмуляция – способ запуска проекта на ПК для проверки его работы без использования панели.

ПК – персональный компьютер.

ЛКМ/ПКМ – левая кнопка мыши/правая кнопка мыши.

1 О программе

Конфигуратор СП210 – программа, позволяющая создавать проекты для панелей оператора СП210. Проект включает:

- экраны и окна с расположенными на них элементами, формирующими пользовательский графический интерфейс;
- библиотеки текстов, переменных, изображений и аудиофайлов;
- настройки обмена с другими устройствами;
- макросы – программы на языке Си, содержащие пользовательские алгоритмы;
- настройки панели оператора.

Конфигуратор позволяет загрузить проект в панель, а также проверить его работу в режиме эмуляции.

Минимальные системные требования конфигуратора:

- процессор Intel Core i3 (2 ГГц) или любой другой с аналогичными характеристиками;
- оперативная память – 4 Гб;
- свободное место на диске – 1 Гб;
- монитор с диагональю не менее 15" и разрешением 1920x1080;
- операционная система – Windows 10 или Windows 11.

ПРИМЕЧАНИЕ
Для корректного отображения интерфейса конфигуратора в параметрах Windows на вкладке **Система / Дисплей** для настройки **Масштаб и разметка** должно быть установлено значение 100%.

В случае некорректного отображения элементов графического интерфейса, в свойствах ярлыка конфигуратора на вкладке **Совместимость** нажмите кнопку **Изменить параметры высокого DPI**, установите галочку **Переопределите режим масштабирования высокого разрешения. Масштабирование выполняется:** и выберите вариант **Система**. Перезапустите конфигуратор для применения настроек.

2 Характеристики панелей оператора СП210

<table>
<tr><th>Модификация</th><th>Экран</th><th>Процессор</th><th>RAM</th><th>Flash-память</th><th>Интерфейсы</th></tr>
<tr><td>СП210-05-Б</td><td>4.3”
480x272</td><td rowspan="5">ARM Cortex-A7 Dual-core 1 ГГц</td><td rowspan="2">128 Мб</td><td rowspan="2">64 Мб</td><td rowspan="2">1xRS-485/RS-232
1xUSB B
1xUSB A</td></tr>
<tr><td>СП210-07-Б</td><td>7”
800x480</td></tr>
<tr><td>СП210-05-Р</td><td>4.3”
480x272</td><td rowspan="3">128 Мб</td><td rowspan="3">128 Мб</td><td>1xEthernet
1xRS-485/RS-232
1xUSB A</td></tr>
<tr><td>СП210-07-Р</td><td>7”
800x480</td><td>1xEthernet
2xRS-485/RS-232
1xUSB A
Аудиовыход</td></tr>
<tr><td>СП210-10-Р</td><td>10.1”
1024x600</td><td>1xEthernet
2xRS-485/RS-232
2xUSB A
Аудиовыход</td></tr>
<tr><td>СП210-15-Р</td><td>15.6”
1920x1080</td><td>ARM Cortex-A7 Dual-core 1.2 ГГц</td><td>256 Мб</td><td>256 Мб</td><td>2xEthernet*
2xRS-485/RS-232
2xUSB A
1xUSB Type C
Аудиовыход</td></tr>
<tr><td colspan="6">ПРИМЕЧАНИЕ
* В текущей версии прошивки поддерживается использование только интерфейса Eth1.</td></tr>
</table>

Базовые модификации СП210 (-Б) не имеют интерфейса Ethernet, аудиовыхода и не поддерживают следующий функционал:

- обмен с другими устройствами по интерфейсу Ethernet;
- загрузку и выгрузку проекта по интерфейсу Ethernet;
- подключение к экрану панели с помощью технологии VNC;
- элемент Отображение видеопотока (RTSP);
- воспроизведение аудиофайлов.

Количество энергонезависимых внутренних регистров (PFW-регистров) у панелей базовых модификаций составляет **1 000 000**, у панелей расширенных модификаций – **3 000 000**.

Элемент Воспроизведение видеофайлов поддерживается только в модификации СП210-15-Р.

3 Отличия панелей оператора СП210 от СП3хх

Линейка панелей оператора СП210 является развитием линейки СП3хх. При этом с точки зрения ПО линейки панелей являются несовместимыми – проекты, созданные в ПО **Конфигуратор СП300**, не могут быть импортированы ПО **Конфигуратор СП210**.

Таблица 3.1 – Основные отличия ПО Конфигуратор СП210 от ПО Конфигуратор СП3хх

Функционал	Назначение
Расширенный набор элементов визуализации: Ползунок, Выпадающий список, Таймер, QR-код, Секторная диаграмма и др.	Создание удобного и функционального операторского интерфейса
Поддержка векторной графики в формате .svg и использование его как основного формата для стандартных элементов визуализации	Масштабирование визуализации под разрешение различных модификаций панелей без ухудшения качества
Возможность создания переменных и их привязки к элементам визуализации	Вместо адресов памяти (например, PSW300) к элементам можно привязывать переменные с содержательными именами (например, «Температура теплообменника Т1»). Для каждой переменной задается тип данных и формат отображения
Карта адресов памяти	Позволяет наглядно оценить, какие адреса памяти панели (и опрашиваемых slave-устройств) используются в проекте
Функция **Контроль оплаты**	Для ограничения времени работы проекта и генерации специальных кодов, требуемых для продолжения его работы. Позволяет снизить риск неполной оплаты труда разработчика недобросовестным заказчиком
Расширенный набор промышленных протоколов: OPC UA, символьный протокол CODESYS, MQTT и др.	Для простой настройки обмена с ПЛК и SCADA-системами
Возможность загрузки и выгрузки проекта через интерфейс Ethernet	Для удаленного обновления проекта в тех случаях, когда доступ к панели затруднен
Продвинутый функционал управления пользователями	Для ограничения доступа к экранам/окнам и элементам визуализации
Функция **Журнал действий оператора**	Для сохранения истории действий пользователей визуализации, которая облегчает анализ причин возникновения нештатных ситуаций и поведения персонала
Поддержка технологии VNC	Для удалённого подключения к экрану панели с целью просмотра и управления
Широкие возможности экспорта и импорта	Экспорт и импорт переменных, текстовых меток, текстов локализации, настроек трендов, тревог и других компонентов для упрощения переноса между проектами и создания проектов с помощью дополнительного ПО (например, список тревог может быть создан в Microsoft Excel и затем импортирован в конфигуратор)
Продвинутые возможности для создания макросов на языке Си	Для создания вспомогательных пользовательских алгоритмов. Полное соответствие стандарту С99, поддержка стандартной библиотеки языка Си, возможность создания комментариев на русском языке, проверка работы макросов в режиме эмуляции
Продвинутые возможности эмуляции проекта	Тестирование практически всего функционала проекта, включая обмен с другими устройствами (в т. ч. в режиме Slave), сохранение файлов, выполнение макросов и т. д.

4 Установка и удаление конфигуратора

4.1 Установка

Порядок установки:

1. Загрузите дистрибутив конфигуратора с сайта ОВЕН.
2. Извлеките файл инсталлятора с разрешением .exe из загруженного .zip-архива.
3. Запустите инсталлятор от имени администратора.

В интерфейсе инсталлятора требуется:

- ознакомиться и согласиться с лицензионным соглашением;
- выбрать директорию, в которую будет установлен конфигуратор. Пользователь ПК должен иметь права администратора на работу в данной директории (т. к. в процессе работы конфигуратор создает в ней различные файлы);
- указать, требуется ли создать ярлык.

Если на ПК уже установлен **Конфигуратор СП210** (например, более ранняя версия), то при запуске инсталлятора будет предложен выбор:

- удалить установленную версию;
- установить новую версию конфигуратора параллельно имеющейся версии (потребуется выбрать другую директорию для установки).

4.2 Удаление

Для удаления конфигуратора следует перейти в директорию его установки, открыть папку **unins** и запустить утилиту **unins000.exe**.

5 Системное меню и экранный конфигуратор

5.1 Системное меню

Системное меню – приложение панели, которое позволяет провести ее настройку в процессе работы проекта.

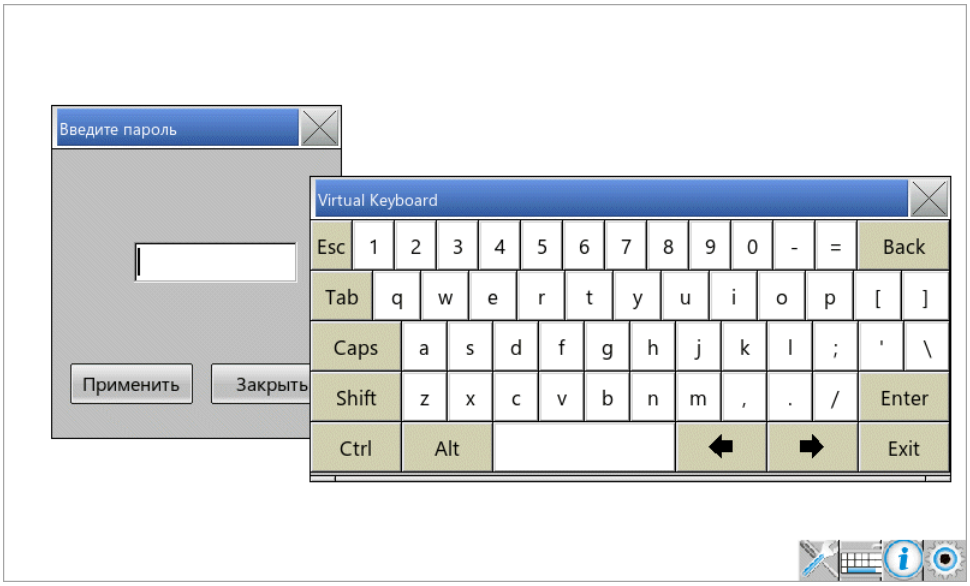


Рисунок 5.1 – Вход в системное меню

ПРИМЕЧАНИЕ
Для конкретного проекта доступ в системное меню может быть отключен с помощью установки галочки **Скрыть системное меню** в **окне загрузки проекта**.

Для входа в системное меню следует нажать кнопку **Настройки** (), расположенную в правом нижнем углу экрана панели. В открывшейся панели отобразятся следующие кнопки:


– кнопка вызова системного меню. Для входа в системное меню нужно ввести пароль (пароль по умолчанию – **123456**);


– кнопка вызова экранной клавиатуры;


– кнопка вызова окна с информацией о панели. Окно содержит следующую информацию:
- версия панели;
- версия прошивки;
- аппаратная ревизия;
- IP-адрес панели;
- ID панели – уникальный идентификатор панели (используется, например, для поиска панелей в локальной сети).

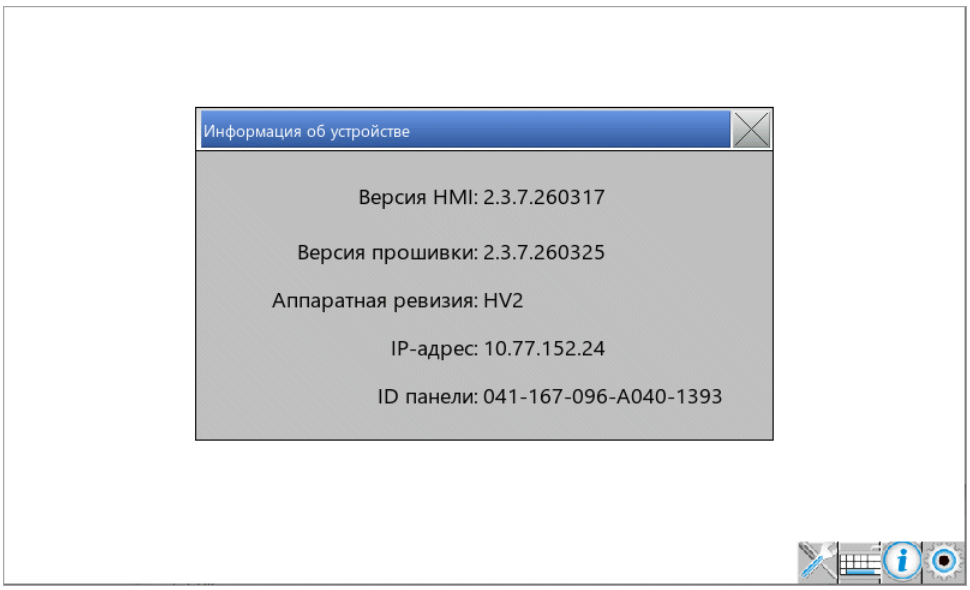


Рисунок 5.2 – Окно информации о панели

При повторном нажатии кнопки **Настройки** дополнительные кнопки скроются.

ПРИМЕЧАНИЕ
Для перемещения кнопки **Настройки** по экрану, зажмите кнопку и переместите курсор в нужное место.

ПРИМЕЧАНИЕ
Для перемещения системного меню, окна с информацией о панели и экранной клавиатуры, зажмите строку их заголовка и переместите курсор в нужное место.

5.1.1 Вкладка «Имя»

На вкладке отображается имя панели, используемое при поиске панелей в локальной сети. Для изменения имени следует нажать кнопку **Изменить**, ввести новое имя и нажать кнопку **Применить**.

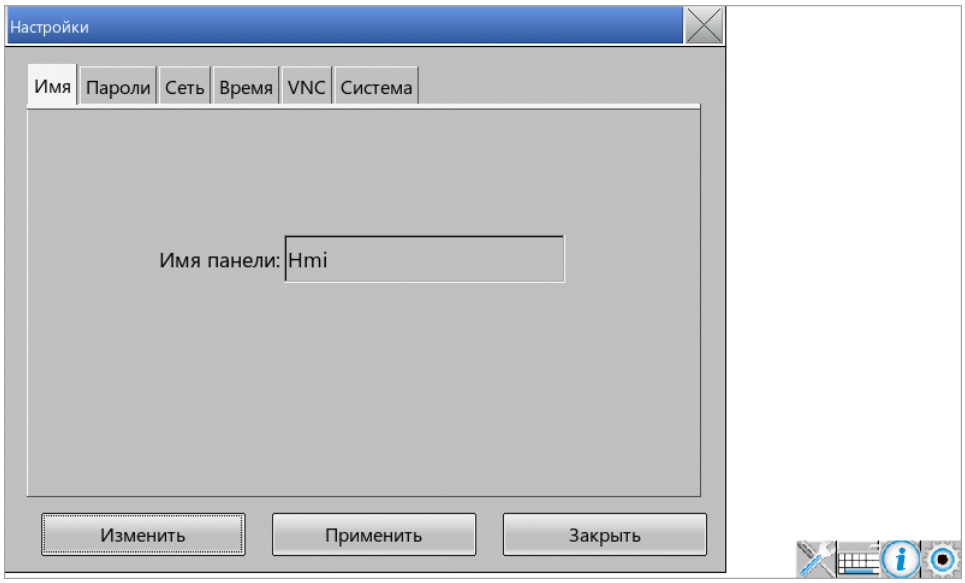


Рисунок 5.3 – Внешний вид вкладки «Имя»

5.1.2 Вкладка «Пароли»

На вкладке производится изменение паролей, используемых для ограничения доступа к различным функциям панели:

- пароль на загрузку проекта (значение по умолчанию – **123456**);
- пароль на выгрузку проекта;
- пароль системного меню (значение по умолчанию – **123456**);
- пароль на подключение к экрану панели с помощью технологии VNC (значение по умолчанию – **123456**);
- пароль для подключения к панели из конфигуратора для настройки функций VNC и MQTT (значение по умолчанию – **12345678**). Доступен только для расширенных модификаций (-P).

При изменении любого пароля требуется ввести текущий пароль и два раза новый пароль. Ограничение на длину пароля – 8 символов.

Кнопка **Сброс паролей** позволяет сбросить все пароли на значения по умолчанию (пароль для сброса – **1101**).

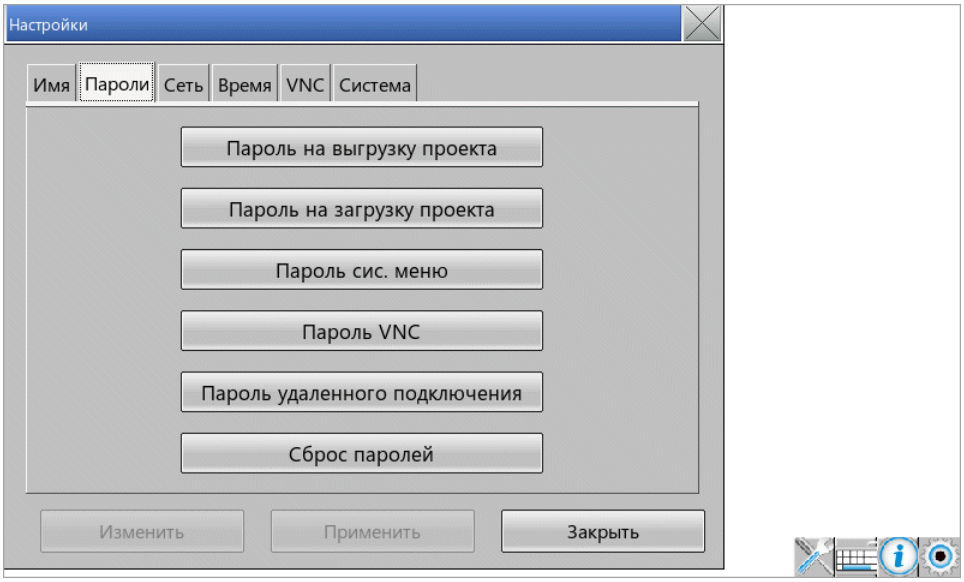


Рисунок 5.4 – Внешний вид вкладки Пароли

5.1.3 Вкладка «Сеть»

На вкладке производится установка сетевых настроек панели. Возможны два варианта:

1. Получение сетевых настроек по DHCP (панель должна быть подключена в локальную сеть, в которой присутствует DHCP-сервер);
2. Ввод настроек (IP-адреса, маски подсети, IP-адреса шлюза и IP-адреса DNS-сервера) вручную.

ПРИМЕЧАНИЕ
Данная вкладка является функциональной только у расширенных модификаций панелей (-P).

Для применения настроек необходимо нажать кнопку **Применить**.

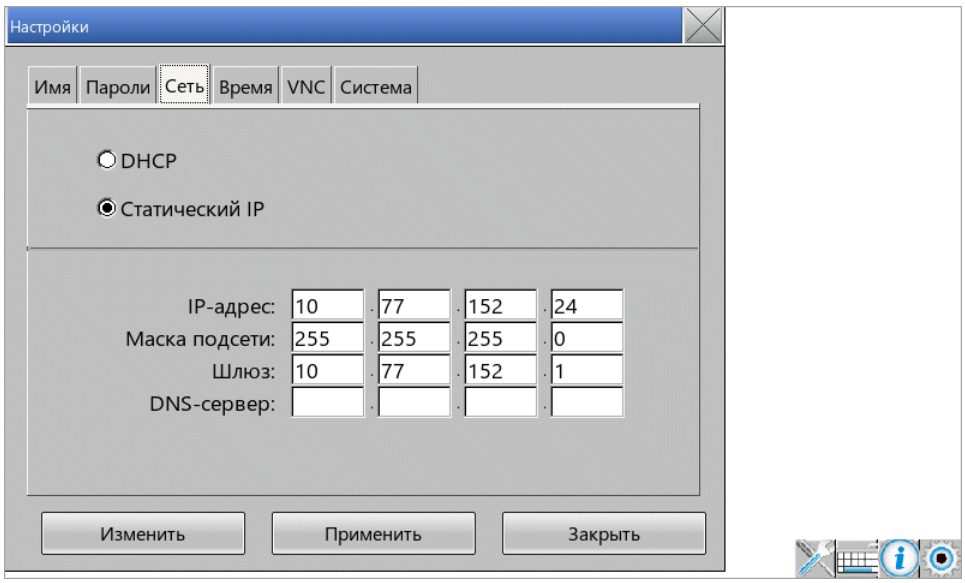


Рисунок 5.5 – Внешний вид вкладки Сеть

5.1.4 Вкладка «Время»

На вкладке производится установка системного времени панели. Для ввода нового значения системного времени следует нажать кнопку **Изменить**, ввести нужные значения и нажать кнопку **Применить**.

ПРИМЕЧАНИЕ
Возможность изменения системного времени можно отключать с помощью соответствующей настройки.

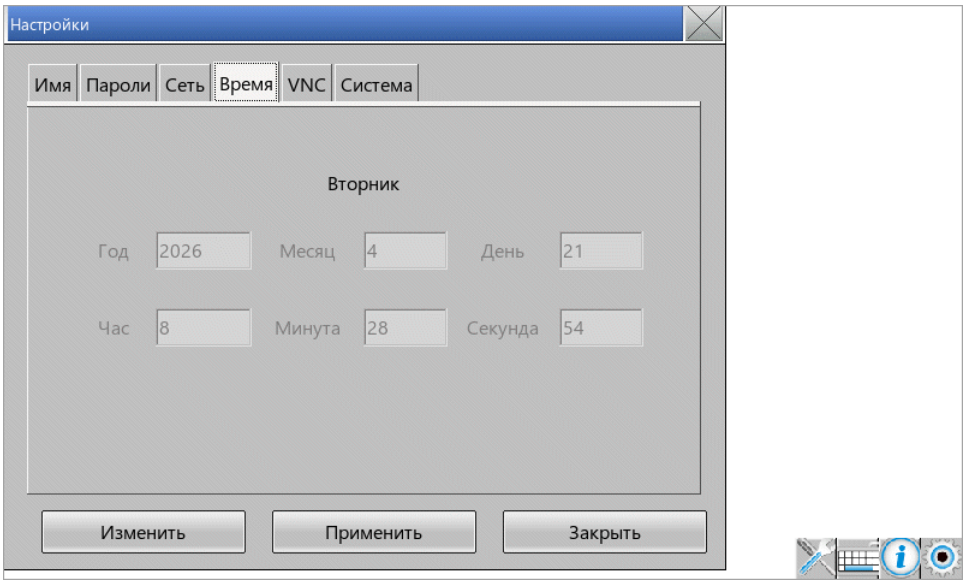


Рисунок 5.6 – Внешний вид вкладки Время

5.1.5 Вкладка «VNC»

На вкладке производится выбор режима подключения к экрану панели с помощью технологии VNC:

- **Один клиент** – поддерживается только одно одновременное подключение;
- **Несколько клиентов** – поддерживается несколько одновременных клиентских подключений (максимальное количество одновременных подключений зависит от свободных ресурсов панели);
- **Отключить** – подключение не поддерживается.

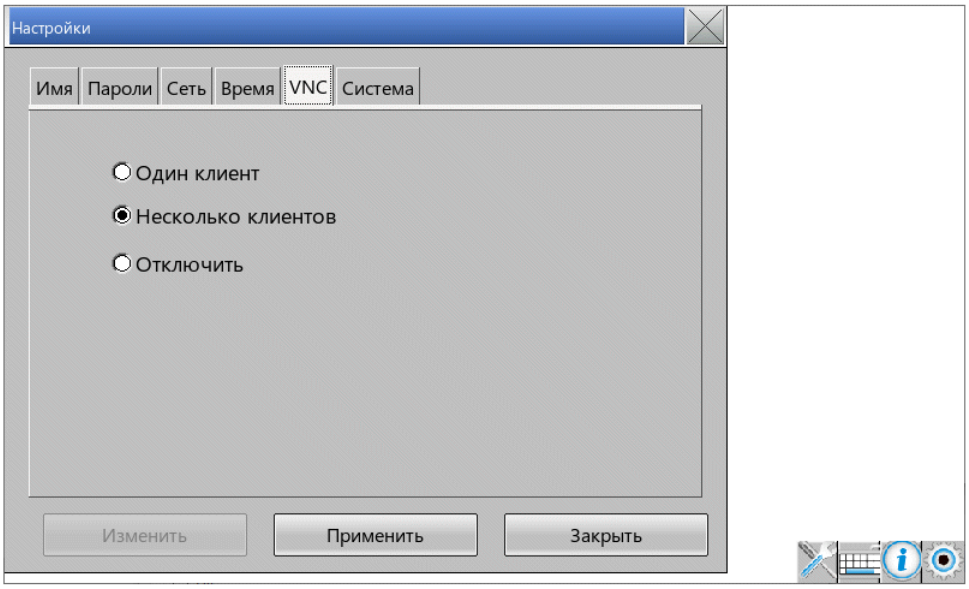


Рисунок 5.7 – Внешний вид вкладки VNC

5.1.6 Вкладка «Система»

На вкладке отображается системная и диагностическая информация:

- версия операционной системы;
- версия прошивки;
- версия панели;
- ID панели;
- MAC-адрес панели;
- объём свободной и общей оперативной памяти;
- объём свободной и общей встроенной flash-памяти;
- процент нагрузки на процессор, создаваемой пользовательским проектом и системным ПО панели.

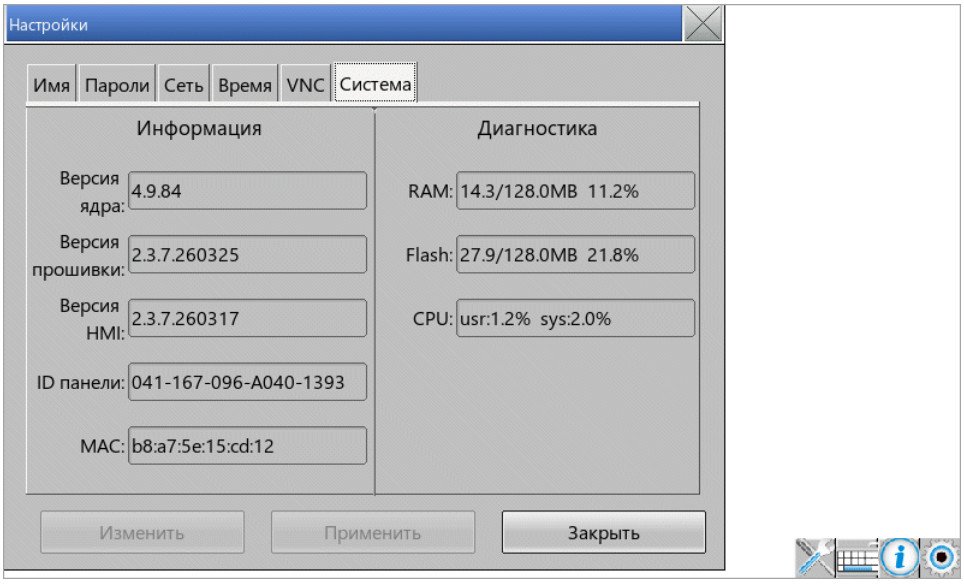


Рисунок 5.8 – Внешний вид вкладки Система

5.2 Сервисное меню и экранный конфигуратор

Экранный конфигуратор – приложение панели, которое позволяет произвести настройку панели. В отличие от системного меню:

- для доступа в экранный конфигуратор требуется перезагрузка панели;
- для доступа в экранный конфигуратор не требуется ввод пароля.

Для перехода в экранный конфигуратор следует:

- отключить питание панели;
- зажать любую точку экрана панели;
- включить питание панели, не отпуская экран;
- дождаться появления сервисного меню.

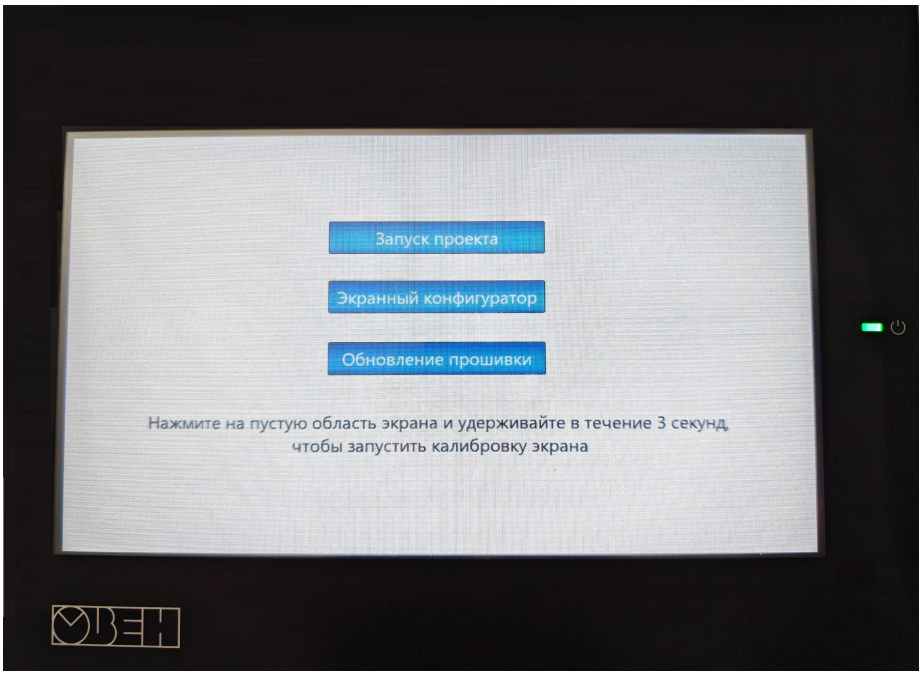


Рисунок 5.9 – Внешний вид сервисного меню

Сервисное меню содержит следующие кнопки:

- кнопку запуска пользовательского проекта;
- кнопку запуска экранного конфигуратора;
- кнопку обновления прошивки панели с USB-накопителя.

Для запуска калибровки экрана зажмите на три секунды любую точку экрана. В процессе калибровки необходимо последовательно нажать подсвечивающиеся на экране ключевые точки. Новые коэффициенты калибровки применятся после перезагрузки панели.

5.2.1 Экранный конфигуратор

Конфигуратор включает в себя следующие экраны:

- **Информация** – соответствует окну **Информация** и вкладке **Система** системного меню;
- **Время** – соответствует вкладке **Время** системного меню;
- **Сеть** – соответствует вкладке **Сеть** системного меню;
- **Пароли** – соответствует вкладке **Пароли** системного меню (без возможности изменения пароля VNC и пароля удаленного подключения).

Внешний вид экранов конфигуратора может быть изменен в проекте панели (см. **Дерево проекта**/узел **Системные экраны**).

ПРИМЕЧАНИЕ
Некорректные изменения могут нарушить нормальную работу экранного конфигуратора.

5.2.2 Обновление прошивки

Для обновления прошивки панели с USB-накопителя следует:

- подготовить USB-накопитель (файловая система FAT32, объём – не более 32 Гб);
- разместить в корне накопителя файл прошивки (.bin) для нужной модификации панели;
- перейти в сервисное меню;
- нажать кнопку **Обновление прошивки**;
- дождаться завершения процедуры обновления.

Также обновление прошивки можно выполнить путем загрузки проекта из новой версии конфигуратора СП210. В каждую версию конфигуратора включены файлы соответствующих версий прошивок.

6 Первый старт

6.1 Основная информация

В данной главе приводится информация по созданию и проверке работы простого проекта.

В рамках данного проекта:

- панель функционирует в режиме **Modbus RTU Master**, считывая из slave-устройства значение с плавающей точкой;
- считанное значение отображается с помощью элемента **Цифровой дисплей**;
- если это значение превышает заданную уставку, то элемент **Индикатор** находится во включенном состоянии. Если это значение меньше или равно уставке – в выключенном состоянии.

Для имитации slave-устройства будет использоваться программа **MasterOPC Universal Modbus Server** (возможно использование любой аналогичной программы).

Будут рассмотрены два варианта:

1. Использование настоящей панели, интерфейс RS-485 которой подключен к ПК с помощью конвертера RS-485/USB (например, ОВЕН АС4-М);
2. Использование режима онлайн-эмуляции.

6.2 Настройка MasterOPC Universal Modbus Server

1. Загрузите установщик по ссылке (любая версия для ОС Windows) и установите OPC-сервер.
2. Запустите OPC-сервер.
3. Создайте новую конфигурацию с помощью кнопки **Создать**.

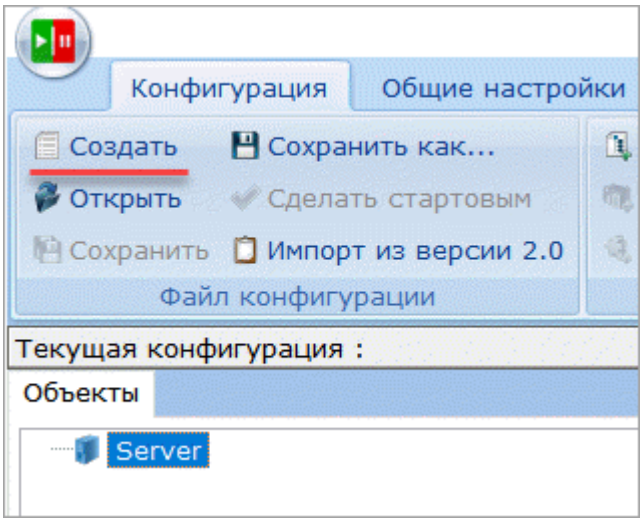


Рисунок 6.1 – Создание конфигурации в OPC-сервере

4. Нажмите правой кнопкой мыши на узел **Server** и добавьте новый коммуникационный узел.

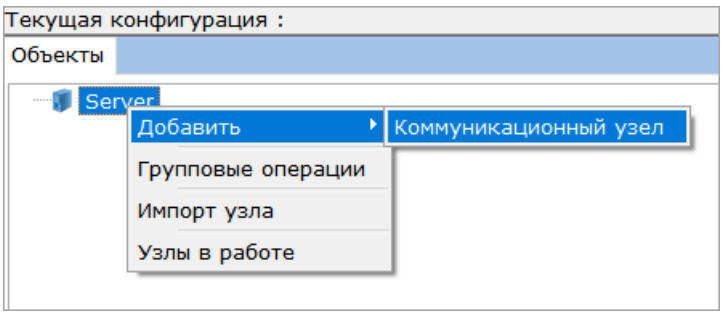


Рисунок 6.2 – Добавление коммуникационного узла

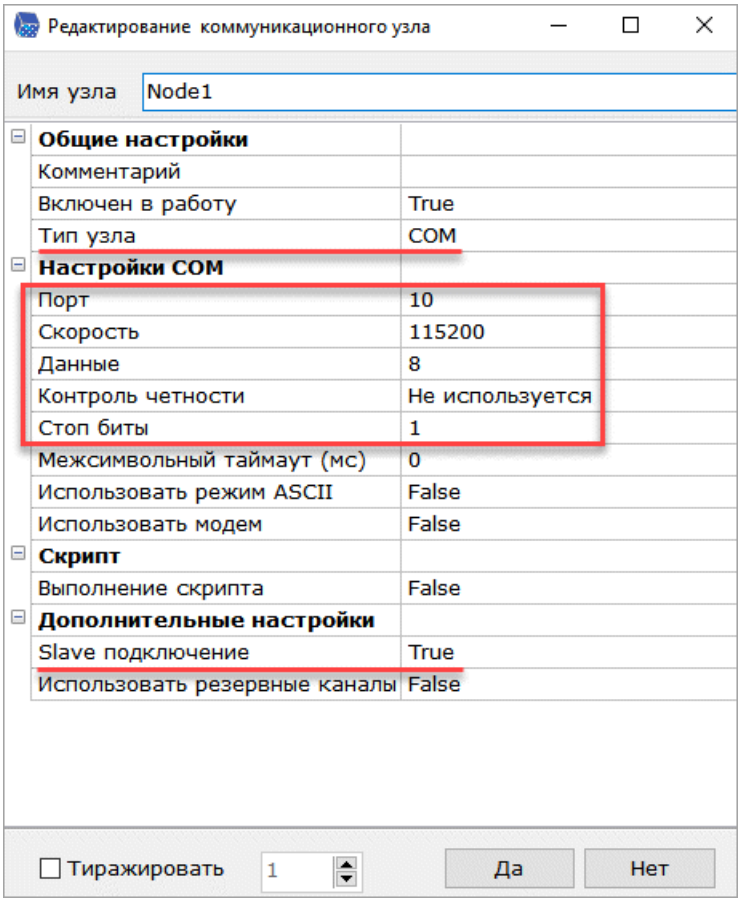


Рисунок 6.3 – Настройки коммуникационного узла

В настройках узла выберите:

- **Тип узла** – COM;
- **Порт** – метод выбора номера порта будет описан в следующих пунктах;
- **Скорость/Данные/Контроль четности/Стоп биты** – настройки COM-порта. В примере будут использоваться настройки: 115200/8/Не используется/1;
- **Slave-подключение** – True.

Нажмите кнопку **Да**.

5. Нажмите правой кнопкой мыши на коммуникационный узел и добавьте устройство.

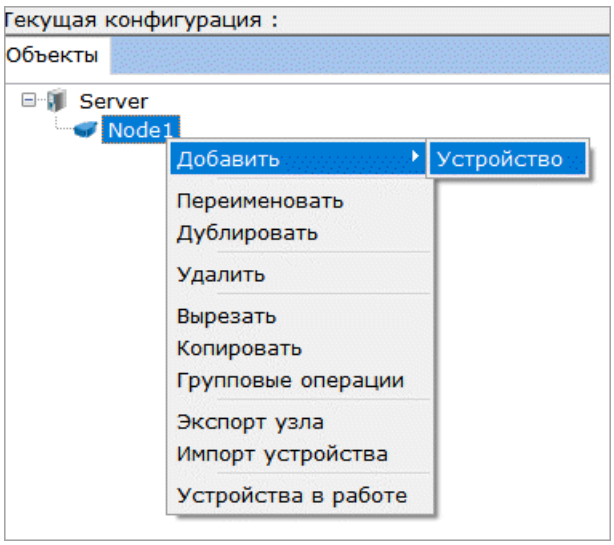


Рисунок 6.4 – Добавление устройства

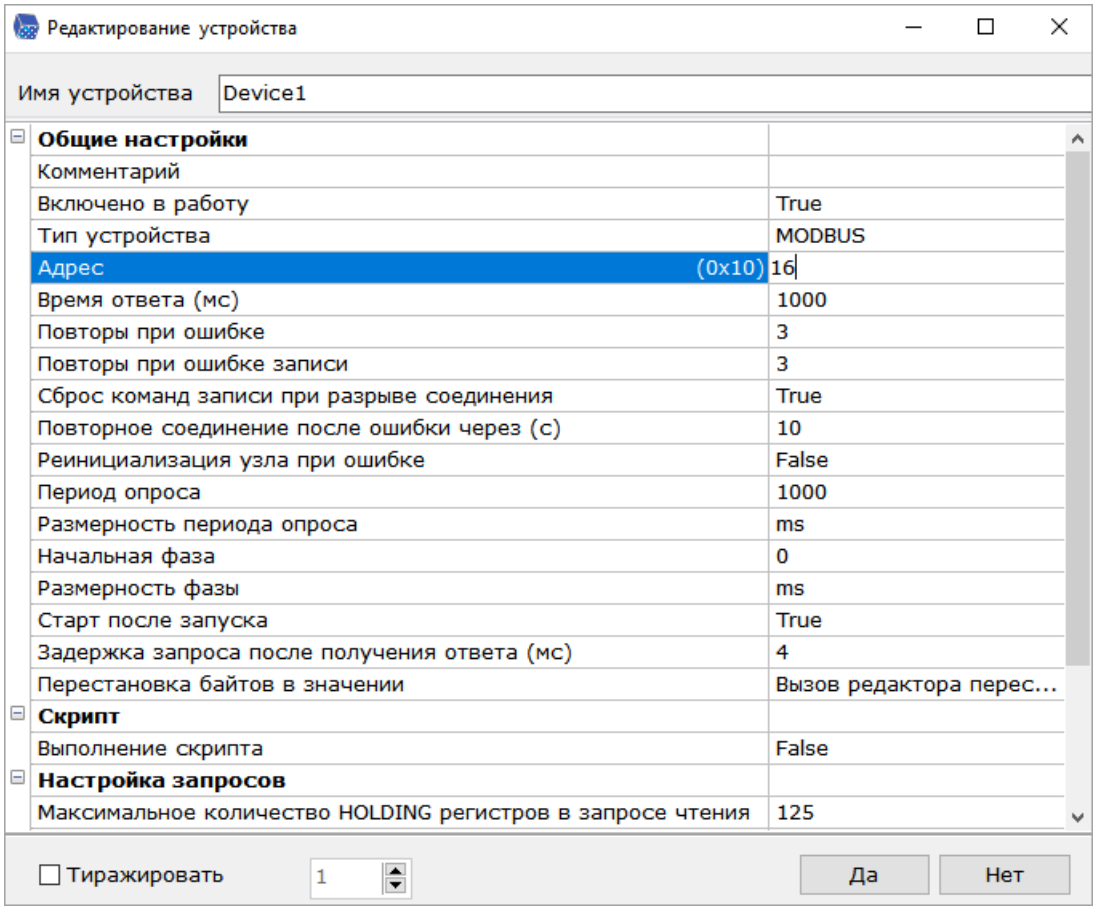


Рисунок 6.5 – Настройки устройства

В настройках устройства укажите его адрес (в примере используется адрес — **16**).

6. Нажмите правой кнопкой мыши на устройство и добавьте тег.

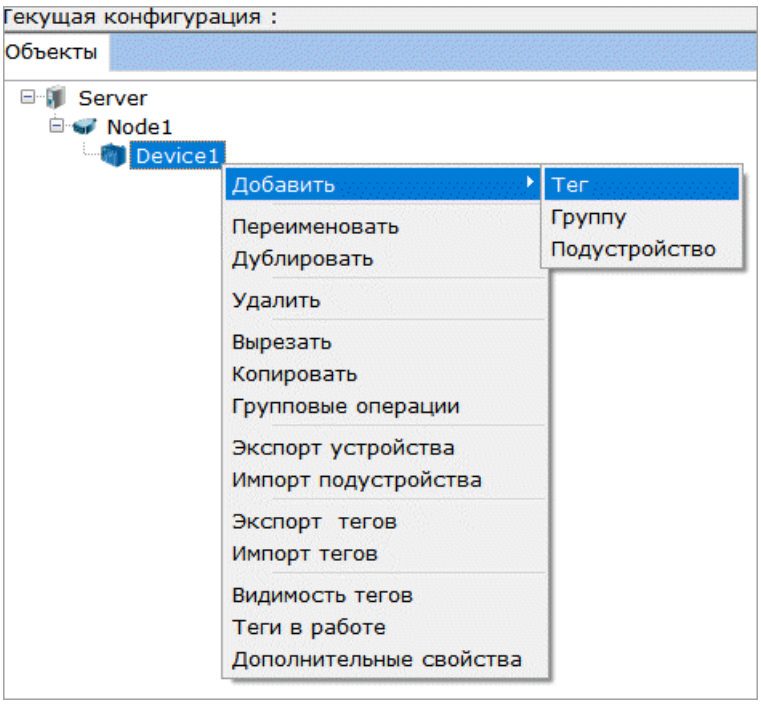


Рисунок 6.6 – Добавление тега

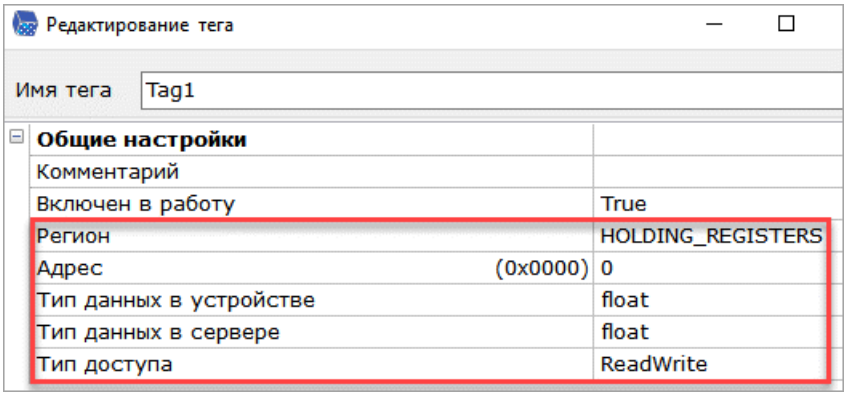


Рисунок 6.7 – Настройки тега

В настройках тега выберите:

- **Регион** – HOLDING_REGISTERS;
- **Адрес** – 0;
- **Тип данных в устройстве** – Float;
- **Тип данных в сервере** – Float;
- **Тип доступа** – ReadWrite.

Таким образом, OPC-сервер будет функционировать как Modbus RTU Slave устройство с адресом **16**, которое содержит один параметр с плавающей точкой, занимающий holding-регистры **0-1** (значение типа Float занимает 2 регистра).

6.3 Первый старт с панелью (часть 1)

1. Подключите к ПК конвертер RS-485/USB. В диспетчере устройств определите номер созданного виртуального COM-порта. В настройках OPC-сервера в параметре **Порт** укажите номер COM-порта (см. рисунок 6.3).
2. Подключите к COM-порту панели адаптер, входящий в комплект поставки. Соедините клеммы A и B интерфейса RS-485 с аналогичными клеммами конвертера.

   ПРИМЕЧАНИЕ
   Клеммы A и B соответствуют сигналам D+ и D-.

3. Подключите к панели источник питания.
4. Для панели расширенной модификации (-Р) перейдите в системное меню и задайте сетевые настройки панели таким образом, чтобы они соответствовали сетевым настройкам ПК (при подключении панели напрямую к сетевой карте ПК) или локальной сети (при подключении панели к локальной сети, к которой подключен ПК).

   Панель базовой модификации (-Б) подключите к ПК с помощью кабеля USB B.

5. Запустите **Конфигуратор СП210** и выполните следующие действия:
   - Нажмите кнопку **Файл – Создать проект** и выберите модификацию панели. Нажмите **Далее**.

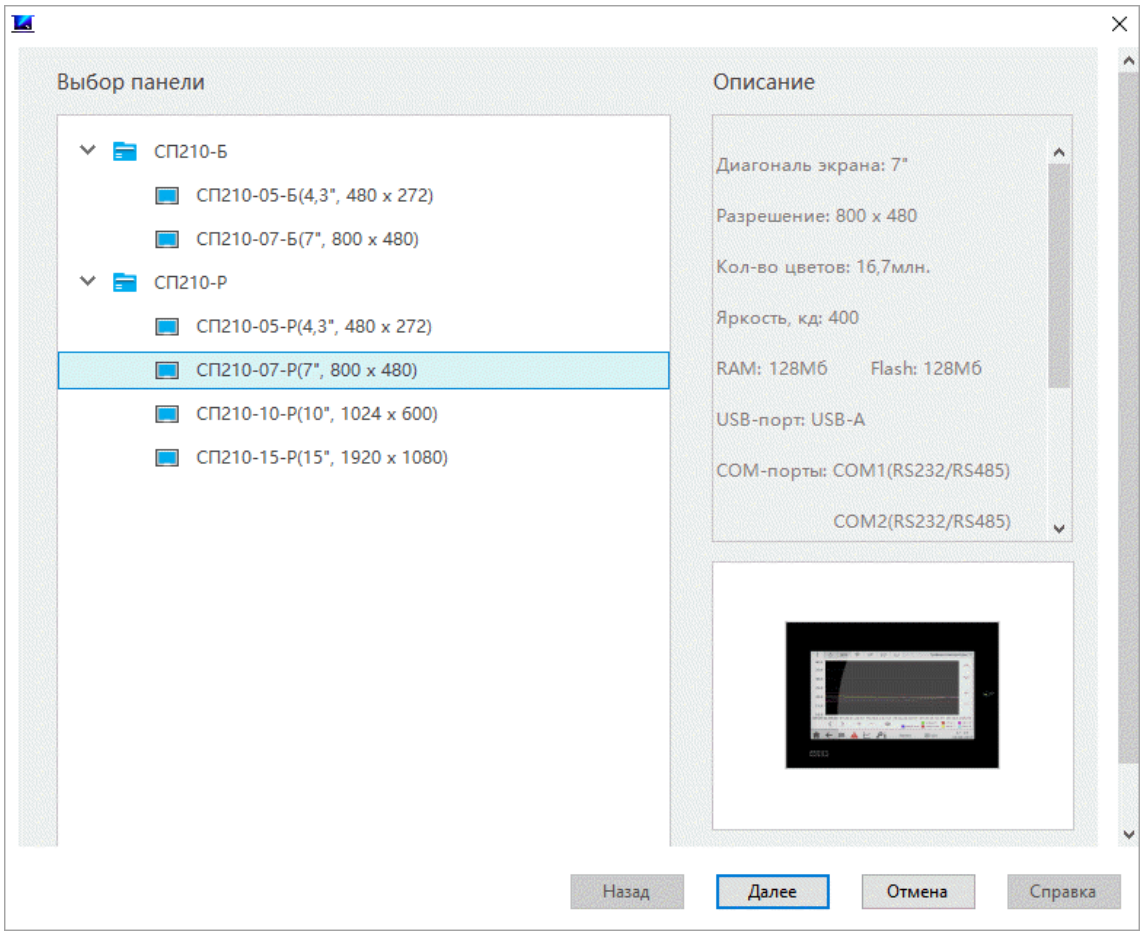


Рисунок 6.8 – Выбор модификации панели

Для панели расширенной модификации (-Р) выберите COM-порт панели, который будет использоваться.

Для панели базовой модификации выбор не требуется (т. к. доступен только один COM-порт).

• Выберите протокол **Modbus**, режим **Modbus RTU Master** и нажмите кнопку **Добавить устройство**.

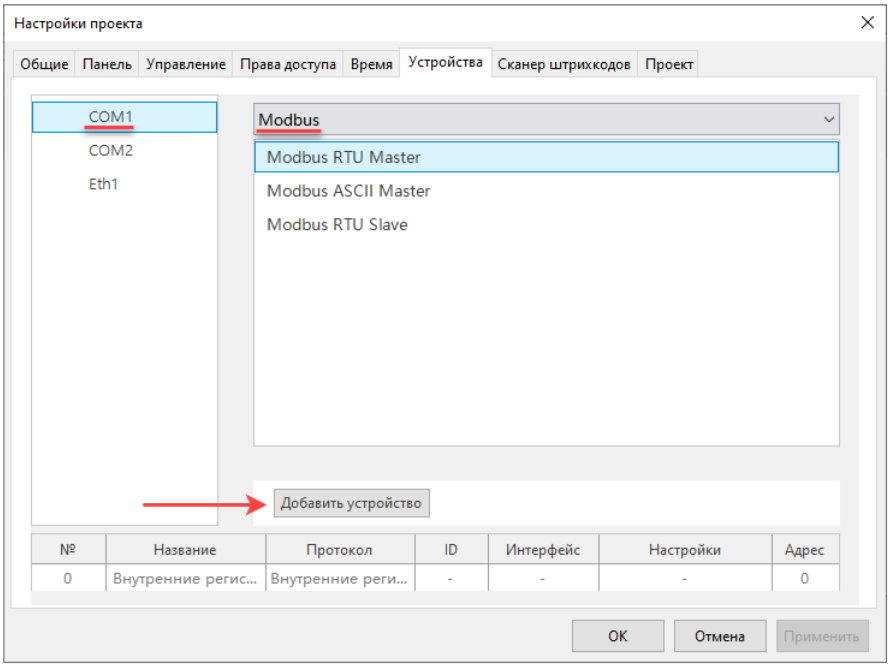


Рисунок 6.9 – Настройка панели в режиме Modbus RTU Master

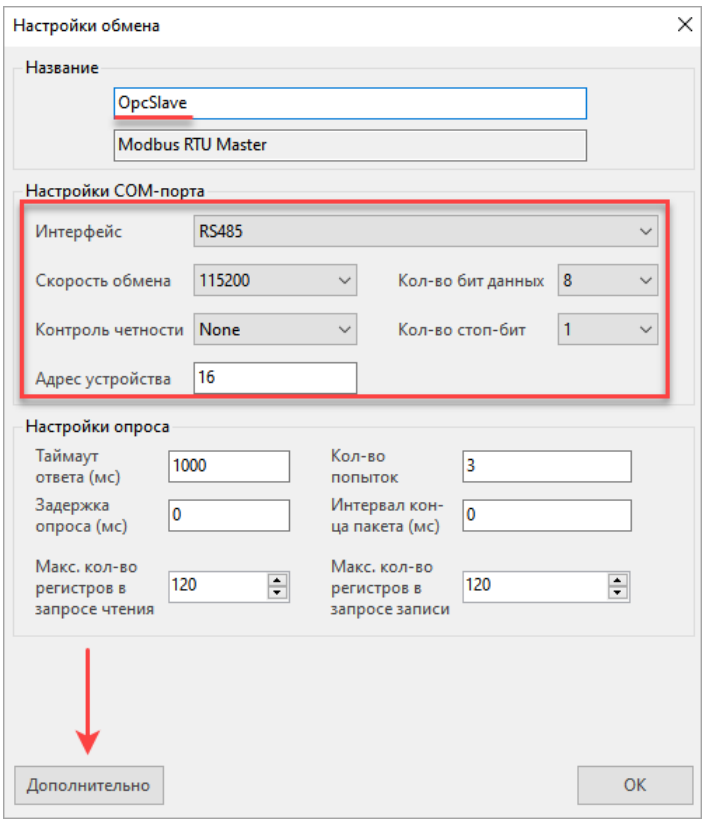


Рисунок 6.10 – Настройки опрашиваемого slave-устройства

• Введите параметры опрашиваемого устройства (соответствуют настройкам OPC-сервера, см. рисунок 6.3):
  – **Имя** – имя устройства (в рамках примера – OpcSlave);
  – **Скорость** – 115200;

- **Количество бит данных** – 8;
- **Режим контроля чётности** – None;
- **Количество стоп-бит** – 1;
- **Адрес** – 16 (см. рисунок 6.5).

Нажмите на кнопку **Дополнительно** и для типа **DWord** выберите порядок регистров **CDAB** (т. к. в панели и OPC-сервере используется разный порядок байт для передачи 32-битных значений).

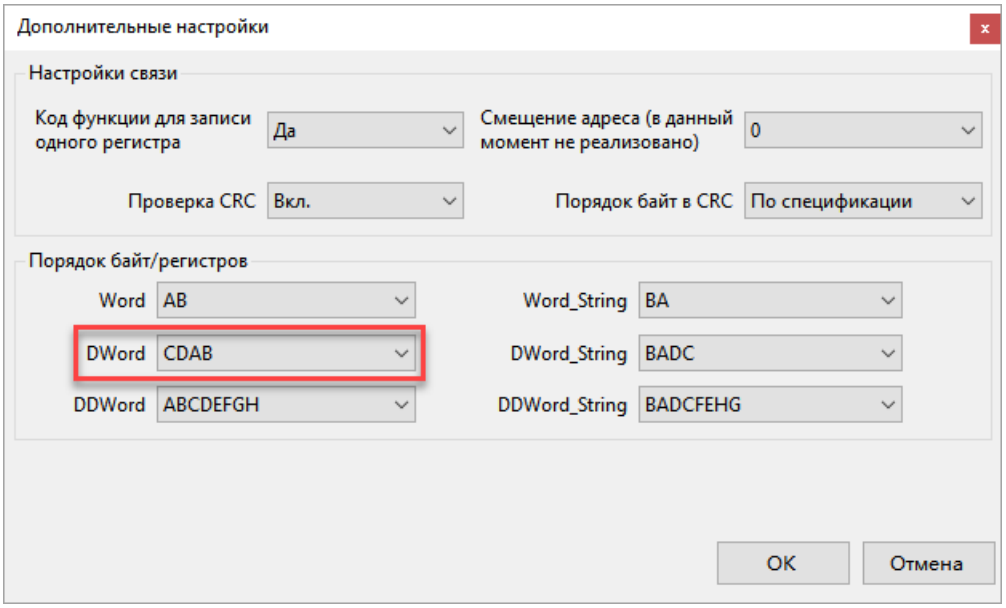


Рисунок 6.11 – Настройка порядка регистров для 32-битных значений

6. Добавьте на экран панели элемент **Цифровой дисплей** (доступен на панели элементов в группе **Базовые элементы**, а также в меню **Элементы / Дисплей**).

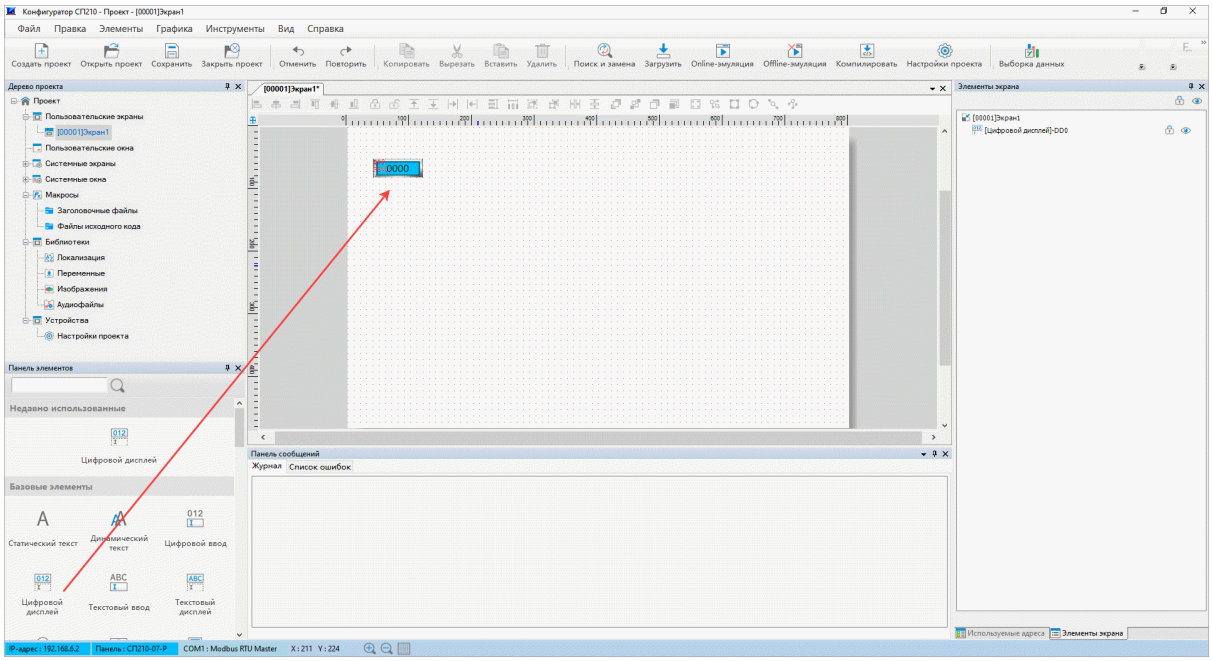


Рисунок 6.12 – Добавление элемента Цифровой дисплей

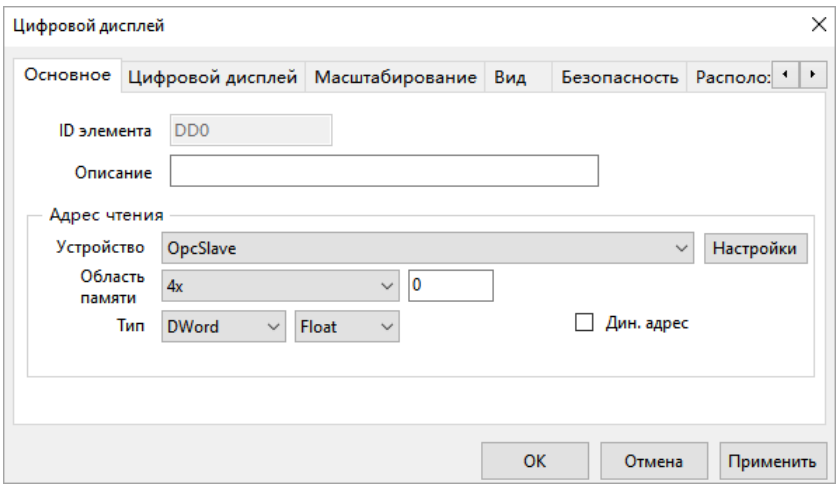


Рисунок 6.13 – Настройки элемента Цифровой дисплей (вкладка Основное)

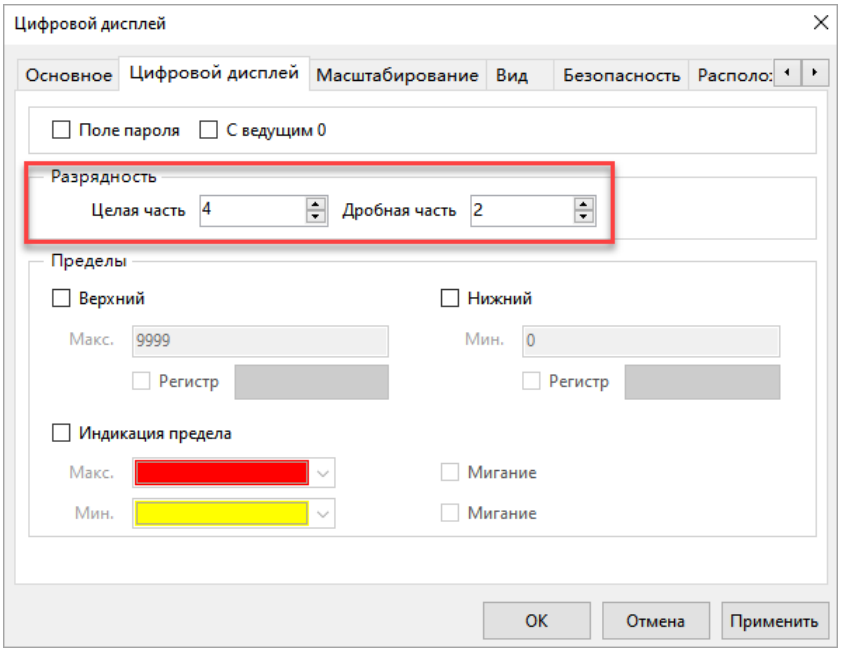


Рисунок 6.14 – Настройки элемента Цифровой дисплей (вкладка Цифровой дисплей)

В настройках элемента на вкладке **Основное** укажите:

- **Устройство** – OpcSlave;
- **Адрес** – 4x0 (4x – это идентификатор области holding-регистров; 0 – адрес начального регистра параметра);
- **Тип данных** – Dword (значение типа Float занимает 2 регистра);
- **Формат** – Float.

На вкладке **Цифровой дисплей** для параметра **Дробная часть** установите значение **2**.

Таким образом, элемент будет отображать значение, целая часть которого не превышает 4 разряда, а дробная – 2 разряда.

7. Запустите OPC-сервер (если кнопка **Старт** неактивна — сохраните конфигурацию).

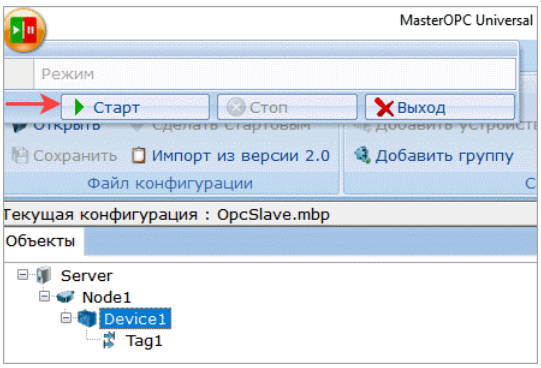


Рисунок 6.15 – Кнопка запуска OPC-сервера

8. В конфигураторе СП210 нажмите кнопку **Загрузить**.

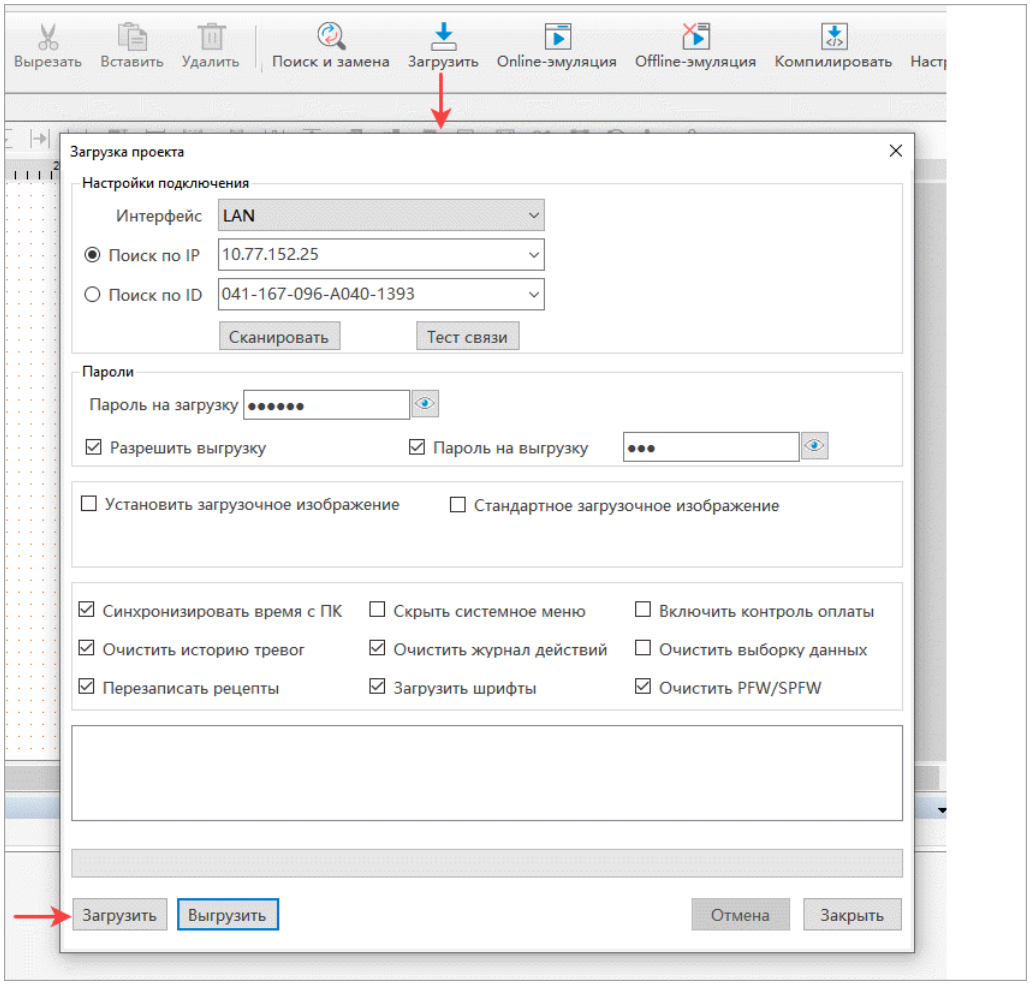


Рисунок 6.16 – Загрузка проекта в панель оператора

Выберите тип подключения:

- для панелей базовой модификации (-Б) – USB;
- для панелей расширенной модификации (-Р) – LAN (и введите IP-адрес панели).

Нажмите кнопку **Загрузить**.

9. Введите в OPC-сервере новое значение для добавленного тега.

   Это же значение должно отобразиться на экране панели.

После этого можно перейти к разделу Первый старт (часть 2).

6.4 Первый старт с эмулятором (часть 1)

1. Загрузите и установите на ПК утилиту для создания виртуальных COM-портов (в примере используется утилита — VSPE).
2. Запустите VSPE.

   Нажмите кнопку **Устройство — Создать новое устройство**:

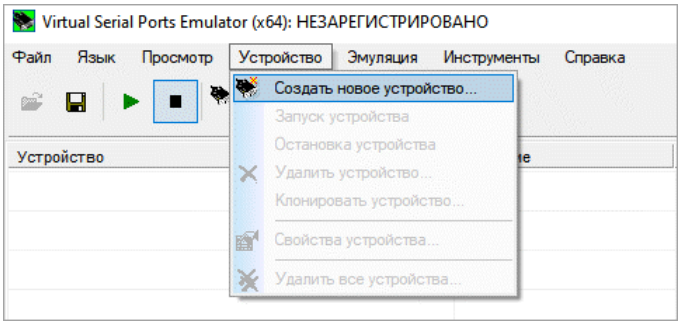


Рисунок 6.17 – Создание нового устройства

- Выберите тип устройства **Virtual Pair** (пара виртуальных COM-портов, соединённых друг с другом).

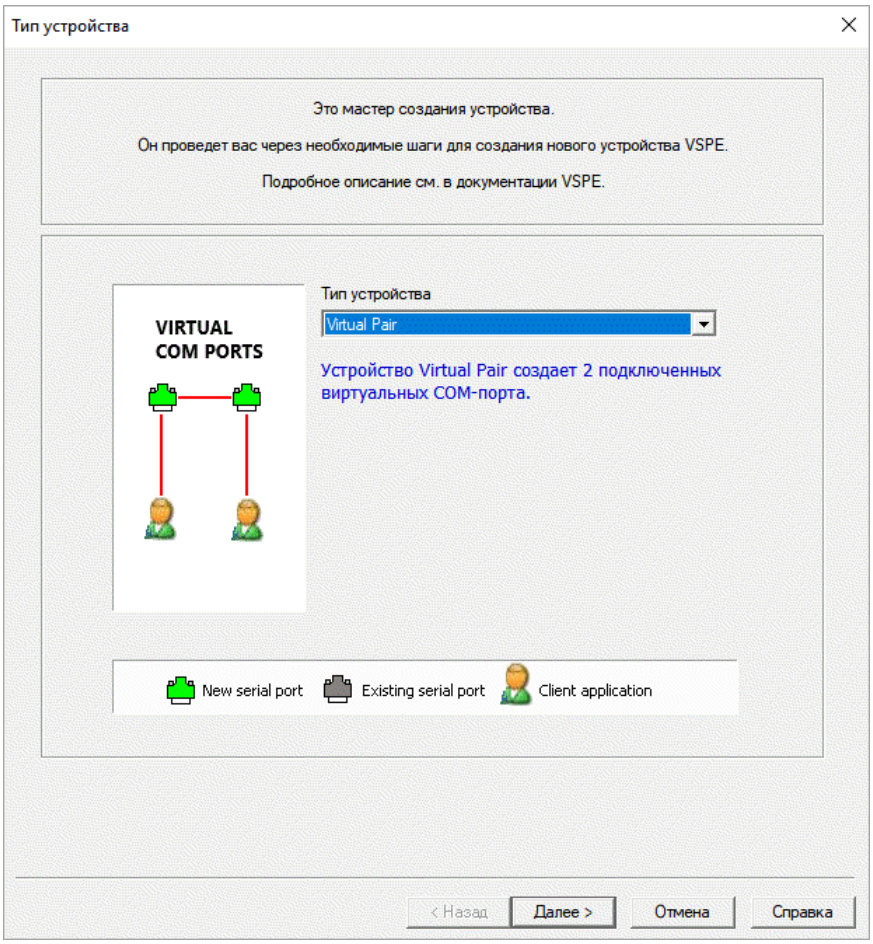


Рисунок 6.18 – Выбор типа устройства

- Укажите номера для создаваемых виртуальных COM-портов (в примере — COM10 и COM11).

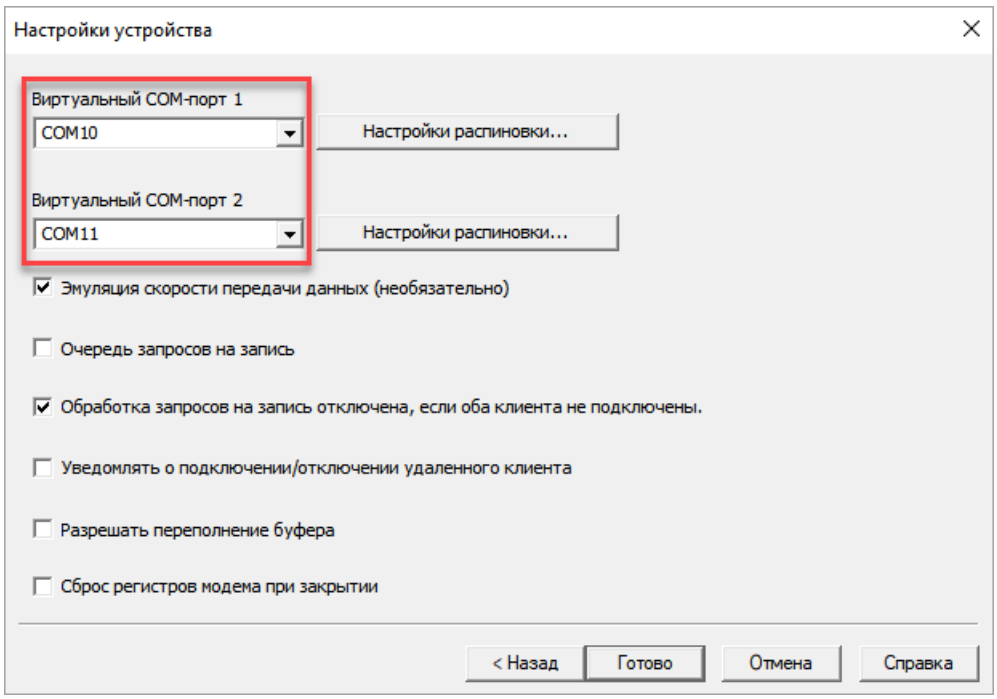


Рисунок 6.19 – Выбор номеров виртуальных COM-портов

ПРИМЕЧАНИЕ
Эти номера не должны использоваться операционной системой. Проверьте их отсутствие в диспетчере устройств на вкладке **Порты** (COM и LPT).

Нажмите на кнопку **Готово**. Созданные порты можно будет использовать (даже если номера не будут добавлены в диспетчере устройств).

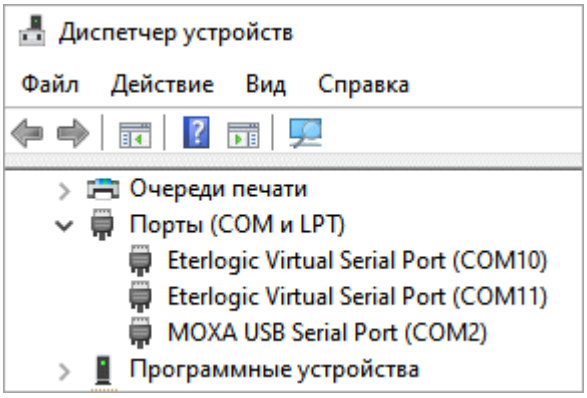


Рисунок 6.20 – Отображение созданных виртуальных COM-портов в Диспетчере устройств Windows

Созданное устройство (pair) отобразится в интерфейсе утилиты.

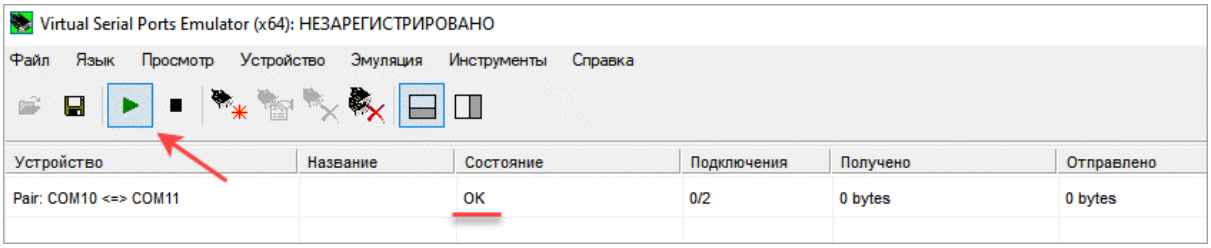


Рисунок 6.21 – Запуск устройства

3. Убедитесь, что утилита находится в режиме запуска. Укажите номер одного из созданных виртуальных COM-портов (например, 10) в параметре **Порт** в настройках OPC-сервера (см. рисунок 6.3).
4. Запустите **Конфигуратор СП210** и выполните следующие действия:
   - Нажмите кнопку **Файл – Создать проект** и выберите модификацию панели. Нажмите **Далее**.
   - Выберите порт **COM1**.
   - Выберите протокол **Modbus**, режим **Modbus RTU Master** и нажмите кнопку **Добавить устройство**.
   - Введите параметры опрашиваемого устройства (соответствуют настройкам OPC-сервера):
     – **Имя** – имя устройства (в рамках примера – OpcSlave);
     – **Скорость** – 115200;
     – **Количество бит данных** – 8;
     – **Режим контроля чётности** – None;
     – **Количество стоп-бит** – 1;
     – **Адрес** – 16 (см. рисунок 6.5).

   Нажмите на кнопку **Дополнительно** и для типа **DWord** выберите порядок байт CDAB (т. к. в панели и OPC-сервере используется разный порядок регистров для передачи 32-битных значений).
5. Добавьте на экран панели элемент **Цифровой дисплей** (доступен на панели элементов в группе **Базовые элементы**, а также в меню **Элементы / Дисплей**).

   В настройках элемента на вкладке **Основное** укажите:
   - **Устройство** – OpcSlave;
   - **Адрес** – 4x0 (4x – это идентификатор области holding-регистров; 0 – адрес начального регистра параметра);
   - **Тип данных** – Dword (значение типа Float занимает 2 регистра);
   - **Формат** – Float.

   На вкладке **Цифровой дисплей** для параметра **Дробная часть** установите значение **2**.

   Таким образом, элемент будет отображать значение, целая часть которого не превышает 4 разряда, а дробная – 2 разряда.
6. Запустите OPC-сервер (если кнопка **Старт** неактивна — сохраните конфигурацию).
7. В конфигураторе СП210 нажмите кнопку **Онлайн-эмуляция**.
   - Откроется окно, в котором следует сохранить файл проекта.
   - Откроется окно **Маппинг COM-портов**, в котором выберите порт панели **COM1** и порт ПК **COM11**. Нажмите кнопку **>>>>**.

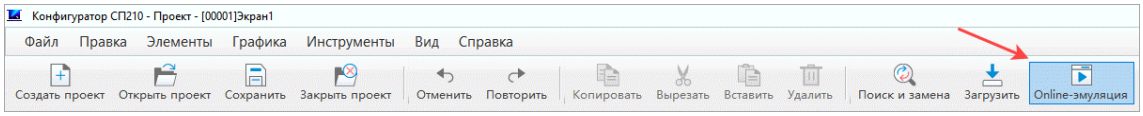


Рисунок 6.22 – Запуск онлайн-эмуляции работы панели

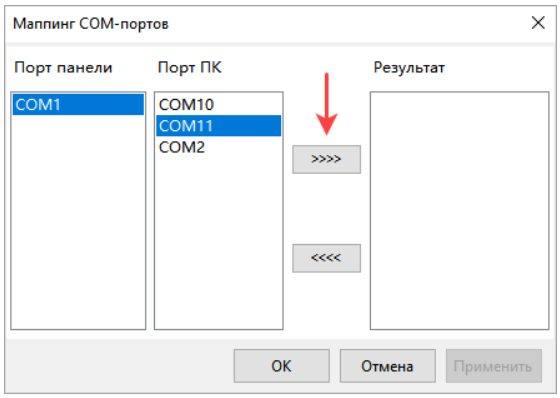


Рисунок 6.23 – Установка соответствия между COM-портом проекта панели и COM-портом ПК

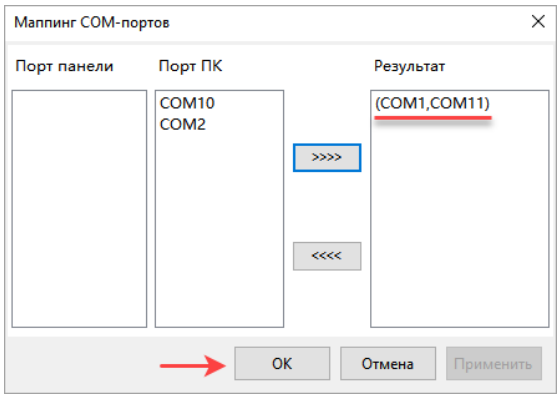


Рисунок 6.24 – Установка соответствия между COM-портом проекта панели и COM-портом ПК

Таким образом, порту **COM1** эмулируемой панели будет соответствовать порт **COM11** ПК.

- Нажмите кнопку **ОК**.
- Откроется окно со списком действий, которые будут выполнены при запуске эмулятора.

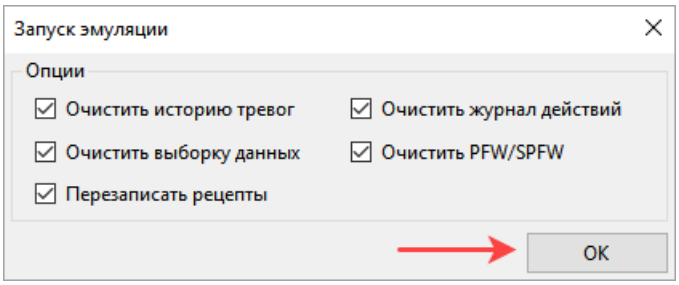


Рисунок 6.25 – Настройки запуска режима онлайн-эмуляции

- Нажмите кнопку ОК.

8. Введите в OPC-сервере новое значение для добавленного тега.

   Это же значение должно отобразиться в окне эмулятора.

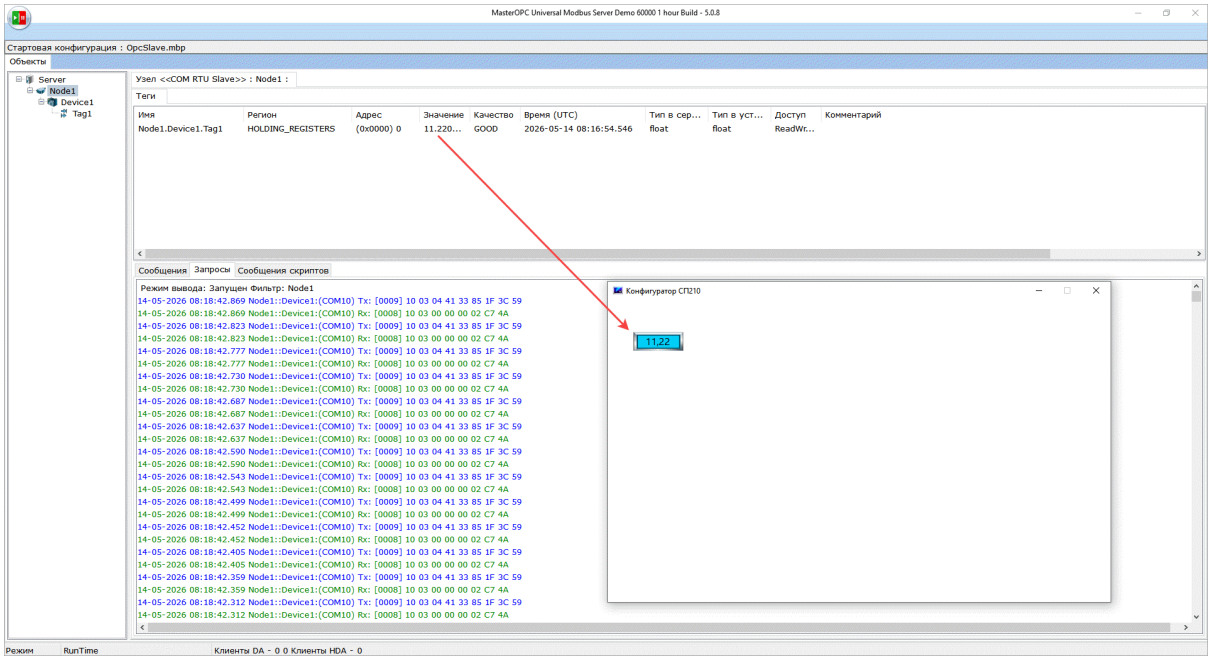


Рисунок 6.26 – Передача данных между OPC-сервером и эмулятором панели

После этого можно перейти к разделу Первый старт (часть 2).

6.5 Первый старт (часть 2)

Ранее была рассмотрена настройка опроса между панелью (или эмулятором панели) и имитатором slave-устройства, и отображение считанного значения на экране.

Так как возможность обработки считанного значения отсутствует (за исключением масштабирования в настройках элемента **Цифровой дисплей**), внесем в проект некоторые изменения:

1. Перейдите в настройки **Цифрового дисплея** и замените устройство на внутренние регистры. Выберите тип регистров **PSW** (оперативные регистры, которые обнуляются после перезагрузки панели). Адрес, тип и формат остаются прежними.

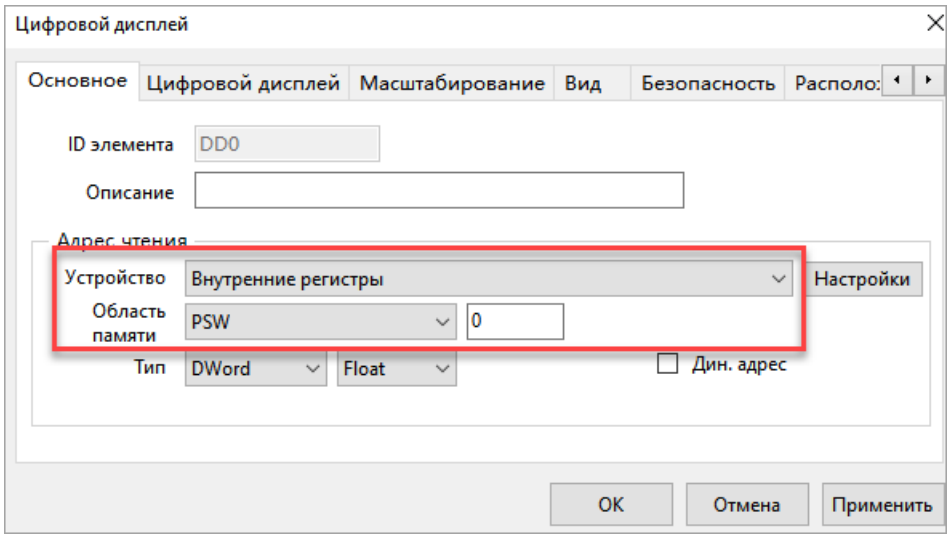


Рисунок 6.27 – Изменение настроек элемента Цифровой дисплей

2. Добавьте на экран элемент **Индикатор**. Настройки индикатора остаются в значениях по умолчанию. Таким образом, к элементу будет привязан внутренний бит панели **PSB0**.

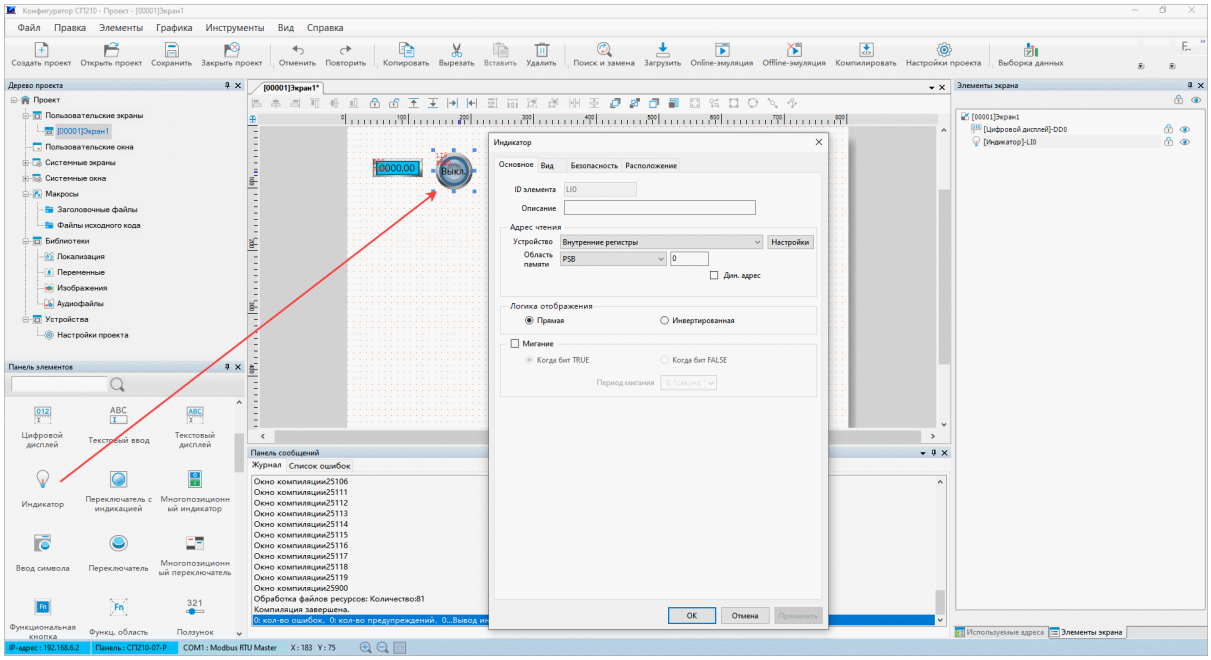


Рисунок 6.28 – Добавление элемента Индикатор

3. В дереве проекта нажмите правой кнопкой мыши на узел **Файлы исходного кода** и создайте новый файл с названием **Func0**.

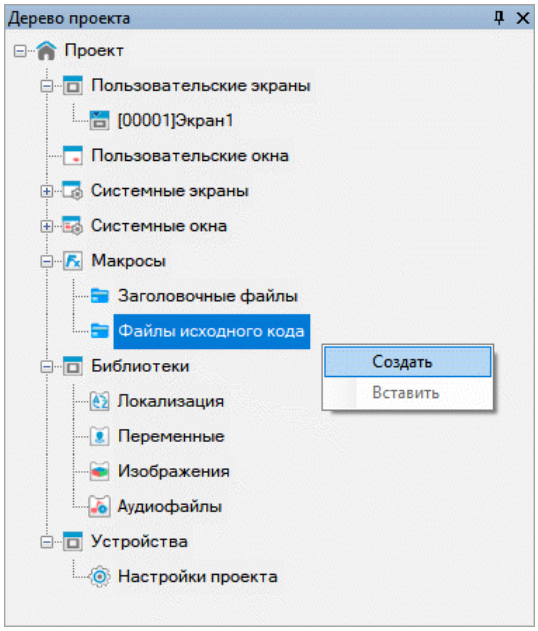


Рисунок 6.29 – Создание макроса

В конструкцию **__MAIN__** добавьте следующий код:

```
__MAIN__()
{
    float rValue;
    rValue = *(float*)(PSW + 0);

    if (rValue > 30.0)
    {
          // Первый аргумент – адрес PSB, второй – значение (0 или 1)
      SetPSB(0, 1);
    }
    else
    {
      SetPSB(0, 0);
    }
}
```

То есть если отображаемое цифровым дисплеем значение превысит **30.0**, то бит **PSB0**, привязанный к индикатору, будет включен, а если оно будет равным или меньше **30.0** – то отключен.

Нажмите кнопку **Компилировать**.

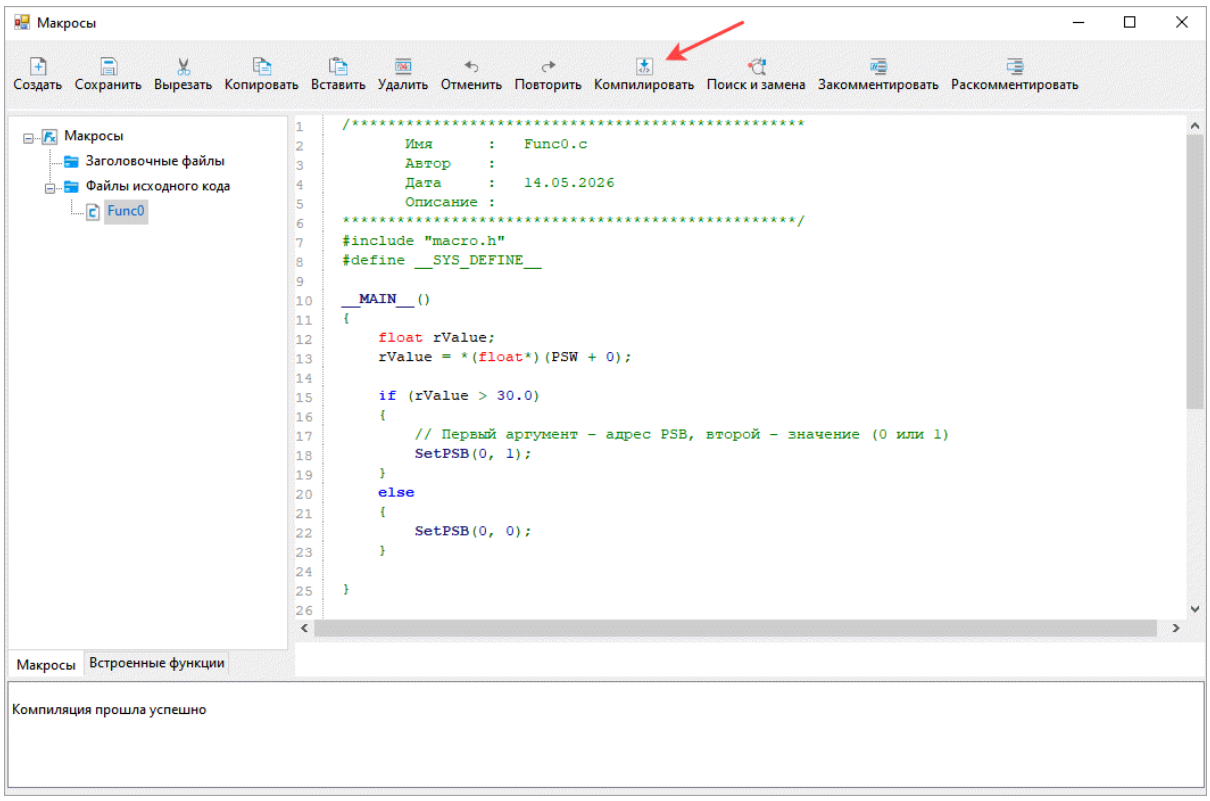


Рисунок 6.30 – Компиляция кода макроса

В случае успешной компиляции отобразится сообщение **Компиляция прошла успешно**. Если после компиляции будет выведена информация об ошибках – то, приведённый выше код был скопирован или набран некорректно.

4. Добавьте на экран элемент **Функциональная область**.

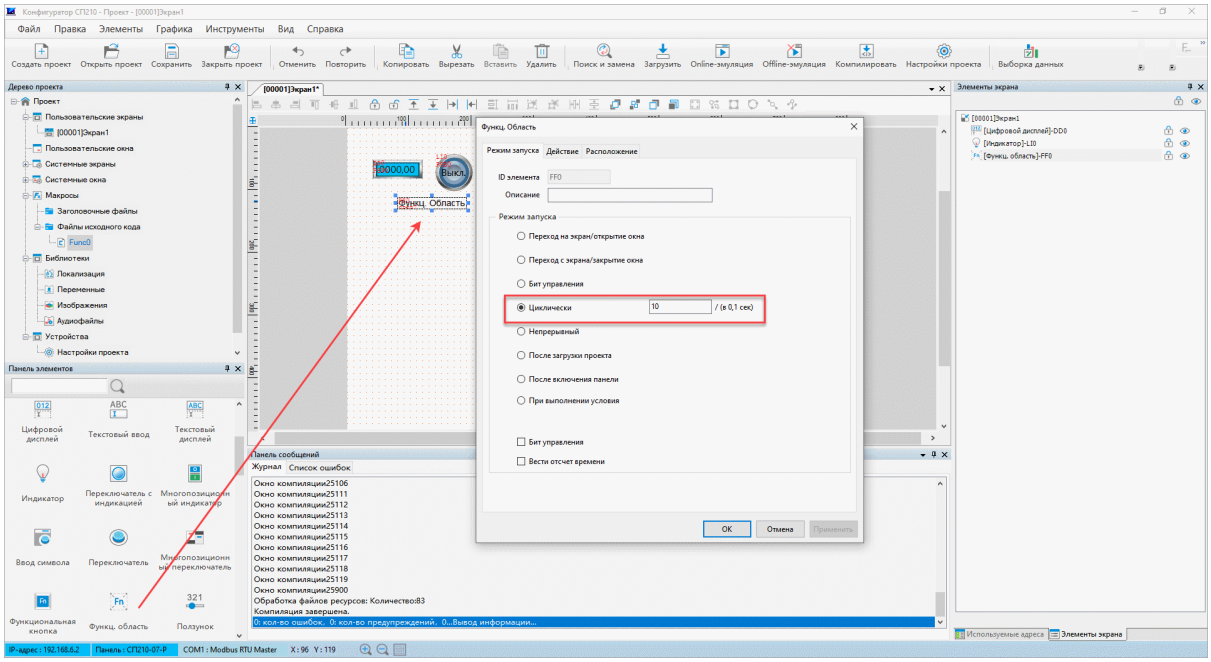


Рисунок 6.31 – Добавление элемента Функциональная область

В настройках выберите режим запуска – **Циклически**, период (в десятых долях секунды) – **10** (1 с).

На вкладке **Действие** добавьте два действия – **Копирование данных** и **Вызов макроса**.

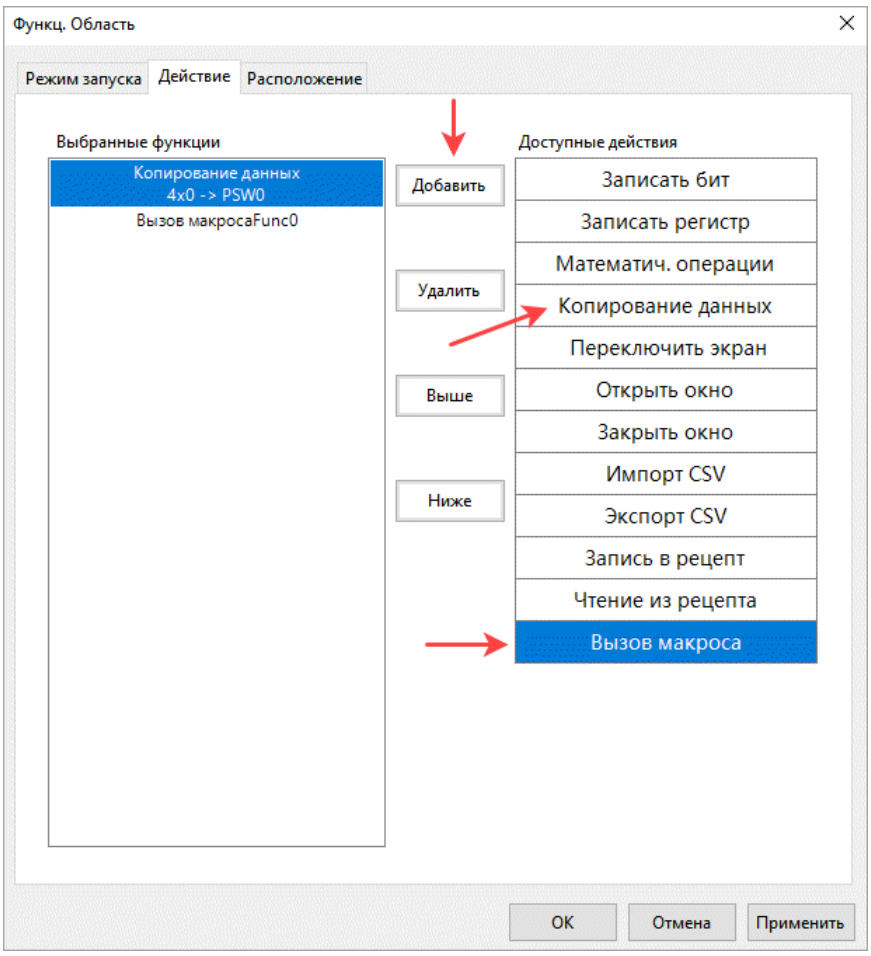


Рисунок 6.32 – Добавление действий функциональной области

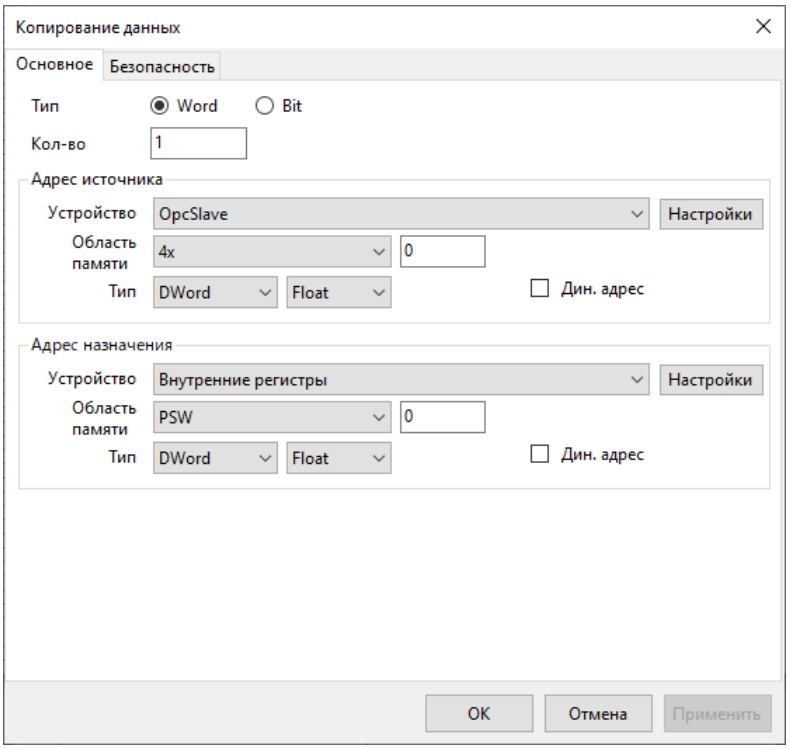


Рисунок 6.33 – Настройки действия Копирование данных

В настройках действия **Копирование данных** укажите:

- **Тип** – Word;
- **Количество** – 1 (будет считано одно значение заданного типа);
- **Адрес источника** – OpcSlave, 4x0, Dword, Float;
- **Адрес назначения** – Внутренние регистры, PSW0, Dword, Float.

Таким образом, данное действие осуществит чтение одного значения типа Float из slave-устройства во внутренние регистры панели.

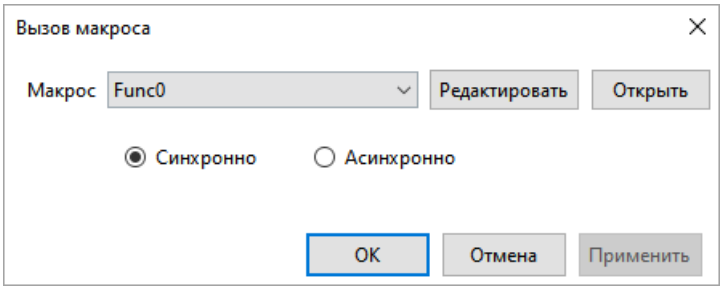


Рисунок 6.34 – Настройки действия Вызов макроса

В настройках действия **Вызов макроса** укажите:

- **Имя макроса** – Func0;
- **Режим исполнения** – синхронный.

5. Если вы перешли в данный пункт:
   - из пункта Первый старт с панелью (часть 1) — загрузите проект в панель.
   - из пункта Первый старт с эмулятором (часть 1) — запустите онлайн-эмуляцию.
6. Введите в OPC-сервере значение **35** для добавленного тега.

   Наблюдайте, как включится индикатор на панели.

   Введите в OPC-сервере значение **25**.

   Наблюдайте, как индикатор отключится.

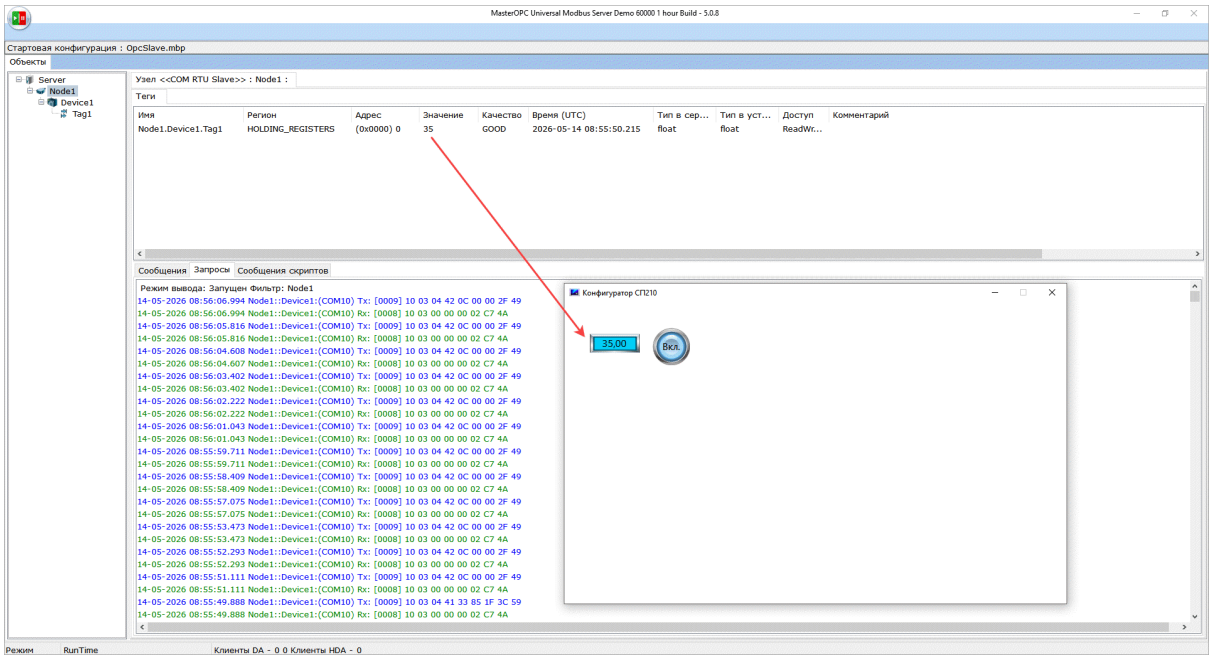


Рисунок 6.35 – Передача данных между OPC-сервером и эмулятором панели

6.6 Заключение

В рамках первого старта рассмотрены следующие темы:

- создание нового проекта в Конфигураторе СП210;
- настройка элементов визуализации;
- настройка опроса в режиме Modbus RTU (через элементы и через функциональную область);
- создание макроса для обработки считанного значения;
- загрузка проекта в панель;
- проверка работы проекта в режиме онлайн-эмуляции.

7 Память панели

7.1 Области памяти

Память панели используется для размещения переменных или значений, привязываемых к параметрам графических элементов и обрабатываемых в макросах. Она подразделяется на 6 областей:

Таблица 7.1 – Области памяти панелей СП210

Область	Объекты	Диапазон	Энергонезависимость
PSB	Биты	0…9999	—
SPSB		См. примечание 1	
PSW	Регистры	0…9999	
SPSW		См. примечание 1	
PFW		Базовые модификации (-Б): 0…999999 Расширенные модификации (-Р): 0…2999999	+ (См. примечание 2)
SPFW		См. примечание 1	

ПРИМЕЧАНИЕ
Примечание 1. Области памяти SPSB, SPSW и SPFW используются для хранения системных переменных. Эти переменные имеют фиксированный адрес и назначение. Список переменных приведен в Библиотеке переменных. Набор системных переменных зависит от версии конфигуратора.

ПРИМЕЧАНИЕ
Примечание 2. Переменные областей PFW и SPFW являются энергонезависимыми и сохраняют свои значения после перезагрузки панели. Физически переменные данных областей размещаются во встроенной flash-памяти панели. Flash-память разделена на блоки размером 512 регистров каждый. Запись во flash-память всегда происходит путём перезаписи как минимум одного блока целиком и выполняется в следующих случаях:

- если в пределах блока изменилось значение хотя бы одного регистра, то весь блок будет записан при очередном срабатывании таймера циклической записи (период – 60 секунд);
- если изменилось содержимое 20 и более блоков (хотя бы по регистру в каждом блоке), то произойдёт внеочередная запись всех затронутых блоков;
- по переднему фронту системного бита **SPSB205** происходит перезапись всех блоков.

Каждый блок рассчитан на **100 000** перезаписей. Для обеспечения равномерного износа в прошивке панели реализован алгоритм балансировки используемых блоков flash-памяти.

7.2 Адреса объектов. Типы и форматы данных

При настройке графических элементов требуется определить источники данных для некоторых параметров. Например, для элемента **Цифровой ввод** ключевым параметром является отображаемое и записываемое элементом значение.

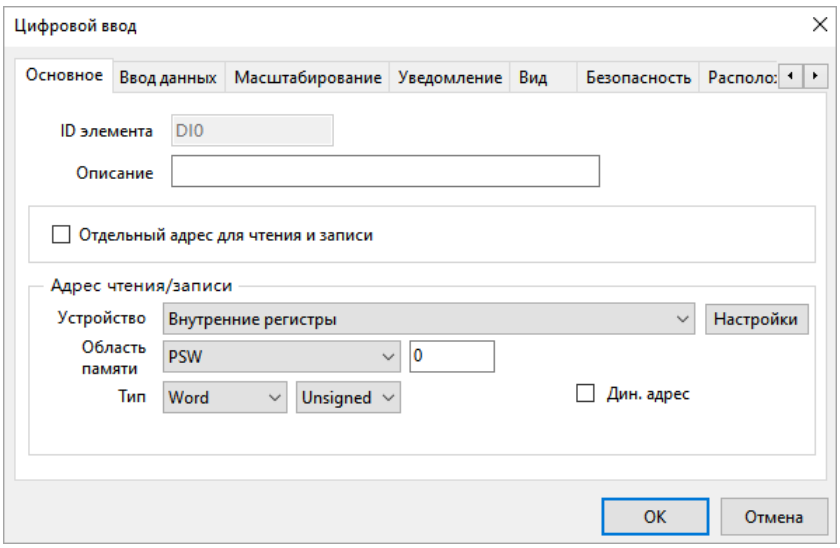


Рисунок 7.1 – Настройки параметра

К настройкам этого значения относятся:

- **Устройство** – место хранения значения (например, одно из slave-устройств, подключенных к панели, или внутренние регистры);
- **Область памяти** и **адрес регистра** – область памяти и адрес регистра, начиная с которого размещается значение;
- **Тип данных** – количество регистров, занимаемых значением;
- **Формат** – формат представления значения.

Таблица 7.2 – Типы числовых данных панелей СП210

Тип	Размер объекта в байтах
Word	2
Dword	4
DDword	8

Таблица 7.3 – Форматы числовых данных панелей СП210

Формат	Описание
BCD	Двоично-десятичный формат Если набор байт является некорректным с точки зрения BCD, то вместо значения отображается ****
Hex	Шестнадцатеричный формат
Signed	Целочисленный знаковый формат
Unsigned	Целочисленный беззнаковый формат
Float	Формат с плавающей точкой. Если набор байтов соответствует NaN или бесконечности, то вместо целой части отображается nan или inf соответственно

Таблица 7.4 – Поддерживаемые сочетания типа и формата с указанием диапазона значений

Формат/Тип	Word	Dword	DDword
BCD	0…9999	0…99999999	0…9999999999999999
Hex	0…FFFF	0…FFFFFFFF	0…FFFFFFFFFFFFFFFF
Signed	-32768…32767	-2^{31}…2^{31}-1	-2^{63}…2^{63}-1

Продолжение таблицы 7.4

Формат/Тип	Word	Dword	DDword
Unsigned	0…65535	$0…2^{32}$	$0…2^{64}$
Float	Сочетание не поддерживается	-9 999 999 999… 9 999 999 999 Точность: 7 знаков после запятой	-9 999 999 999… 9 999 999 999 Точность: 14 знаков после запятой

Для графических элементов, используемых для отображения и записи строк (например, Тестовый дисплей и Текстовый ввод), поддерживается свой набор строковых типов.

Таблица 7.5 – Типы строковых данных панелей СП210

Тип	Размер одного «блока» (не символа) в байтах
Word_String	2
Dword_String	4
DDword_String	8

Данные типы отличаются только размерами одного «блока». Размеры «блока» не связаны с выбранной в настройках элемента кодировкой и влияют только на число байт, выделяемых под значение элемента (параметр **Кол-во**).

Например, если выбрана кодировка **WIN-1251** и **Кол-во** = 2:

- для значения типа Word_String будет выделено 2·2 = 4 байта;
- для значения типа Dword_String будет выделено 2·4 = 8 байт;
- для значения типа DDword_String будет выделено 2·8 = 16 байт.

Таблица 7.6 – Поддерживаемые кодировки панелей СП210

Тип	Описание
WIN-1251	ASCII-подобная кодировка, соответствующая Win1251. Каждый символ занимает 1 байт. Порядок байт устанавливается в настройках элемента; по умолчанию начальный символ строки размещается в младшем байте начального регистра элемента
UTF-16	Unicode, стандарт кодирования UTF-16/UCS2. Каждый символ занимает 2 байта. Порядок байт: начальный символ строки размещается в младшем байте начального регистра элемента
UTF-8	Unicode, стандарт кодирования UTF-8. Каждый символ занимает от 1 до 4 байт. Порядок байт устанавливается в настройках элемента; по умолчанию начальный символ строки размещается в младшем байте начального регистра элемента. Символы, занимающие больше одного байта, размещаются с «отраженным» порядком байт. Например, символ ‘я’ (U+044F), которому соответствует набор байт 0xD1 0x8F, в памяти панели будет размещен с порядком 0x8F 0xD1.

7.3 Динамическая адресация

В некоторых случаях требуется с помощью одного графического элемента в разные моменты времени отображать значения различных регистров. Для этого используется механизм динамической адресации. В настройках параметра элемента нужно установить галочку **Дин. адрес** и указать область памяти и адрес регистра, в который будет записываться «смещение» адреса объекта (бита или регистра) параметра.

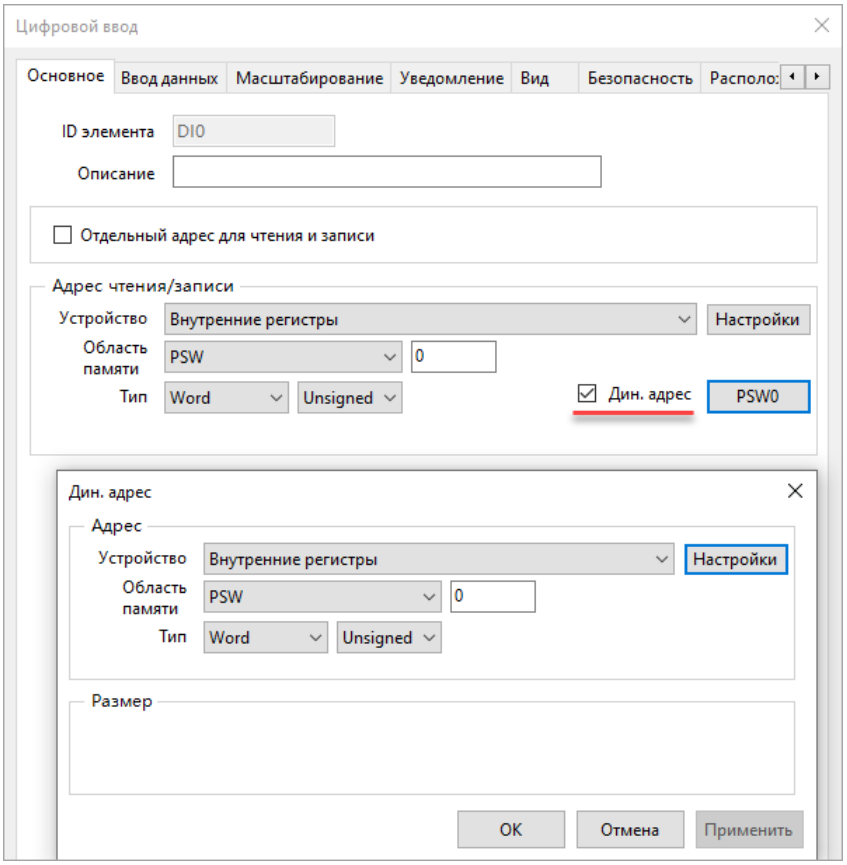


Рисунок 7.2 – Динамическая адресация

Пример:

- Начальные условия: к элементу **Цифровой дисплей** привязан регистр **PSW0**;
- Требуется: в некоторые моменты времени отображать в элементе значения регистра **PSW1** или **PSW2**;
- Настройка: в настройках цифрового дисплея установить галочку **Дин. адрес** и в открывшемся окне выбрать адрес регистра «смещения» – например, **PSW10**.
- Итог:
  - если регистр **PSW10** будет иметь значение **0**, то цифровой дисплей будет отображать значение регистра **PSW0**;
  - если регистр **PSW10** будет иметь значение **1**, то цифровой дисплей будет отображать значение регистра **PSW1**;
  - если регистр **PSW10** будет иметь значение **2**, то цифровой дисплей будет отображать значение регистра **PSW2**.

7.4 Переменные

Вместо выбора в настройках графических элементов адресов областей памяти панели или slave-устройств, можно привязать к ним заранее созданные переменные. Использование переменных упрощает сопровождение проекта за счет введения символьных имен (например, **Давление P1** или **Уставка температуры SP1** гораздо более удобны, чем **PSW0** или **PFW10**).

Также, если один и тот же адрес регистра используется в нескольких графических элементах и потребуется его изменить, в случае привязки переменной достаточно выполнить замену один раз — в настройках этой переменной.

Переменные создаются в библиотеке переменных, открыть которую можно из дерева проекта (**Библиотеки – Переменные**) или в меню **Инструменты – Библиотека переменных.**

В настройках параметра графического элемента для привязки переменной следует нажать на кнопку **Настройки**.

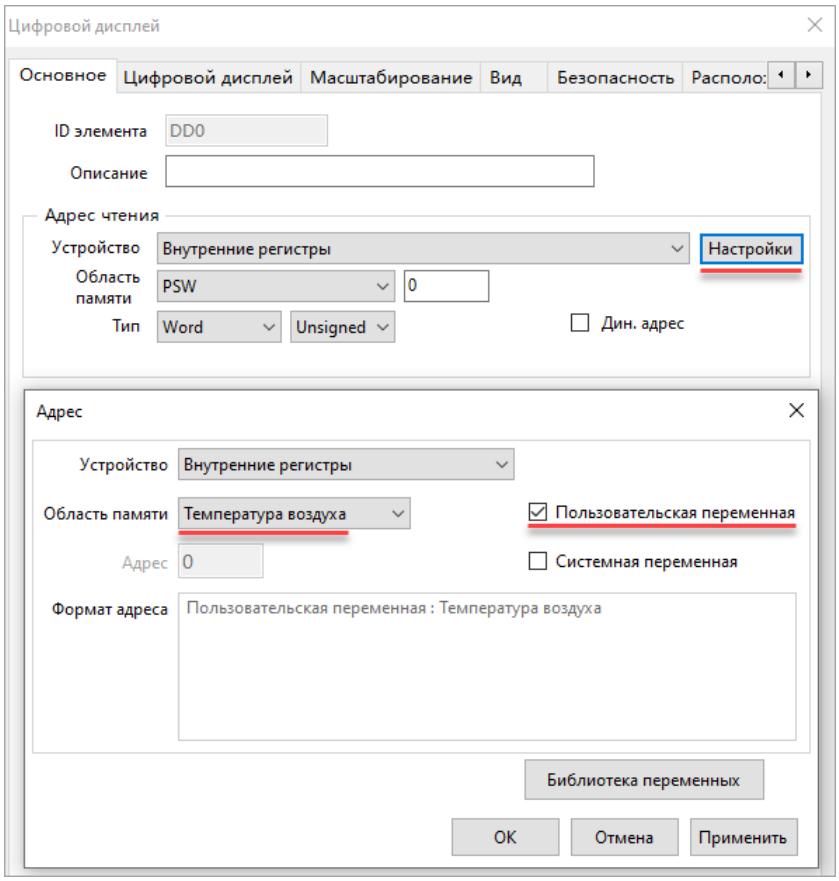


Рисунок 7.3 – Переменные

Откроется окно, в котором доступны следующие настройки:

- область памяти, адрес объекта, тип и формат;
- галочка **Пользовательская переменная**, используемая для привязки к элементу созданной пользователем переменной;
- галочка **Системная переменная**, используемая для привязки к элементу системной переменной;
- кнопка перехода в библиотеку переменных;
- адрес slave-устройства (в случае привязки к параметру адреса бита или регистра slave-устройства);
- диапазон доступных адресов для выбранной области памяти.

8 Обзор интерфейса конфигуратора

8.1 Основная информация

После создания нового проекта окно конфигуратора будет выглядеть следующим образом:

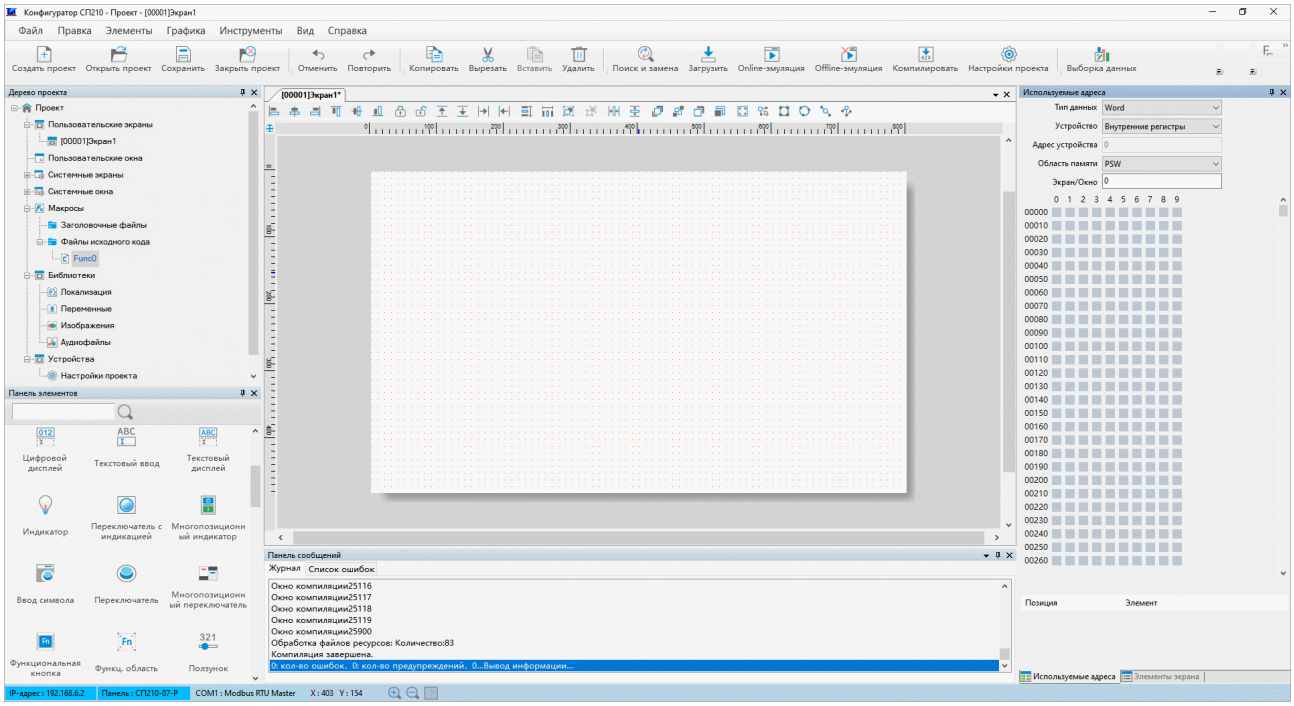


Рисунок 8.1 – Интерфейс конфигуратора СП210

Окно включает в себя следующие области:

- **панель меню** – содержит пункты меню для работы с проектом;
- **панель ярлыков** – содержит ярлыки, дублирующие наиболее часто используемые команды меню;
- **дерево проекта** – содержит список используемых в проекте экранов, окон и макросов, а также ярлыки для доступа к библиотекам проекта;
- **панель элементов** – дублирует ярлыки элементов визуализации из меню **Графика** и **Элементы**;
- **редактор визуализации** – позволяет добавлять на экраны и окна графические элементы и настраивать их параметры;
- карта адресов используемых в проекте битов и регистров с возможностью перехода к элементам, к которым они привязаны;
- список элементов открытого в редакторе визуализации экрана/окна;
- **панель сообщений** – содержит сообщения и ошибки компиляции;
- **строка информации** – содержит информацию о модификации и IP-адресе панели (с возможностью перехода к окнам их выбора), настроенных протоколах обмена, а также кнопки для управления масштабом и видимостью сетки редактора визуализации.

Дерево проекта, панель элементов, карта адресов, список элементов и панель сообщений являются докируемыми окнами, которые можно перемещать по экрану и закреплять в различных областях в доступных для этого слотах.

Для возвращения вида графического интерфейса в состояние «по умолчанию» используется команда **Вид – Восстановить интерфейс по умолчанию**.

8.2 Панель меню

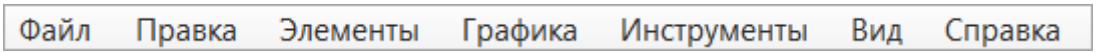


Рисунок 8.2 – Панель меню

Панель меню включает в себя следующие пункты:

- **Файл** – содержит команды для работы с проектом и эмулятором. Команда **Настройки проекта** отвечает за выбор модификации панели, создание пользователей, настройку обмена с другими устройствами и др.;
- **Правка** – содержит команды для работы с графическими элементами (копирование, вставка, удаление, замена привязанных к элементам адресов битов/регистров и т. д.);
- **Элементы** – содержит графические элементы, используемые для создания визуализации проекта (индикаторы, кнопки, элементы отображения и ввода и т. д.);
- **Графика** – содержит отдельную группу наиболее простых графических элементов (линии, прямоугольники и т. д.), элементы отображения внешних графических файлов и некоторые другие элементы;
- **Инструменты** – содержит команды доступа к библиотекам проекта, а также команды перехода к настройкам функции **Контроль оплаты**, окну настроек конфигуратора и команду для подключения к панели с целью настройки сервисов MQTT и VNC;
- **Вид** – содержит кнопки скрытия/отображения отдельных областей конфигуратора;
- **Справка** – содержит команду вызова справки по работе с конфигуратором и информацию о текущей версии конфигуратора.

8.3 Панель ярлыков

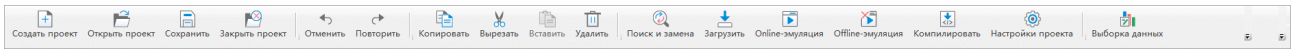


Рисунок 8.3 – Панель ярлыков

Панель ярлыков содержит ярлыки для выполнения:

- команд **Создать, Открыть, Сохранить, Закрыть, Загрузить, Online-эмуляция, Offline-эмуляция, Компилировать, Настройки проекта** (из меню **Файл**);
- команд **Отменить, Повторить, Копировать, Вырезать, Вставить, Удалить, Поиск и замена** (из меню **Правка**);
- команд **Выборка данных, Тревоги, Рецепты, Журнал действий** (из меню **Элементы**);
- команды **Контроль оплаты** (из меню **Инструменты**);
- команды **Масштаб по умолчанию**, которая восстанавливает исходный масштаб редактора визуализации.

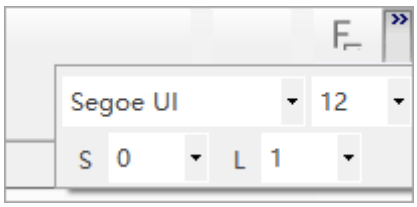


Рисунок 8.4 – Выпадающие списки панели ярлыков

Также панель содержит выпадающие списки:

- **F** – изменение шрифта и размера текста выделенных в редакторе визуализации элементов;
- **S** – изменение состояния всех отображаемых в редакторе элементов (например, чтобы посмотреть, как будут выглядеть кнопки в нажатом/отжатом состоянии);

- **L** – изменение языка всех отображаемых в редакторе элементов (например, чтобы посмотреть, как будет выглядеть экран в случае переключения языка и проверить, вписываются ли тексты локализации в размеры элементов).

8.4 Дерево проекта

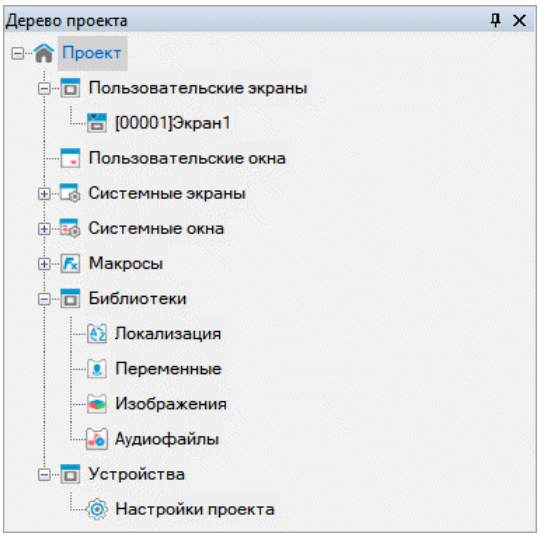


Рисунок 8.5 – Дерево проекта

Дерево проекта содержит следующие узлы:

- **Пользовательские экраны** – создаваемые пользователем экраны графического интерфейса оператора;
- **Пользовательские окна** – создаваемые пользователем окна, открываемые поверх экранов графического интерфейса оператора;
- **Системные экраны** – экраны экранного конфигуратора;
- **Системные окна** – окна системных сообщений и диалогов ввода (клавиатур);
- **Макросы** – заголовочные файлы и файлы исходного кода на языке С, используемые в рамках данного проекта;
- **Библиотеки** – библиотеки ресурсов проекта: текстов, переменных, изображений и аудиофайлов;
- **Устройства** – узел **Настройки проекта**, нажатие на который открывает меню **Файл – Настройки проекта – Устройства**, в котором настраивается обмен с другими устройствами.

Для добавления пользовательского экрана следует нажать **ПКМ** на узел **Пользовательские экраны** и использовать команду **Добавить**. Откроется окно настроек экрана:

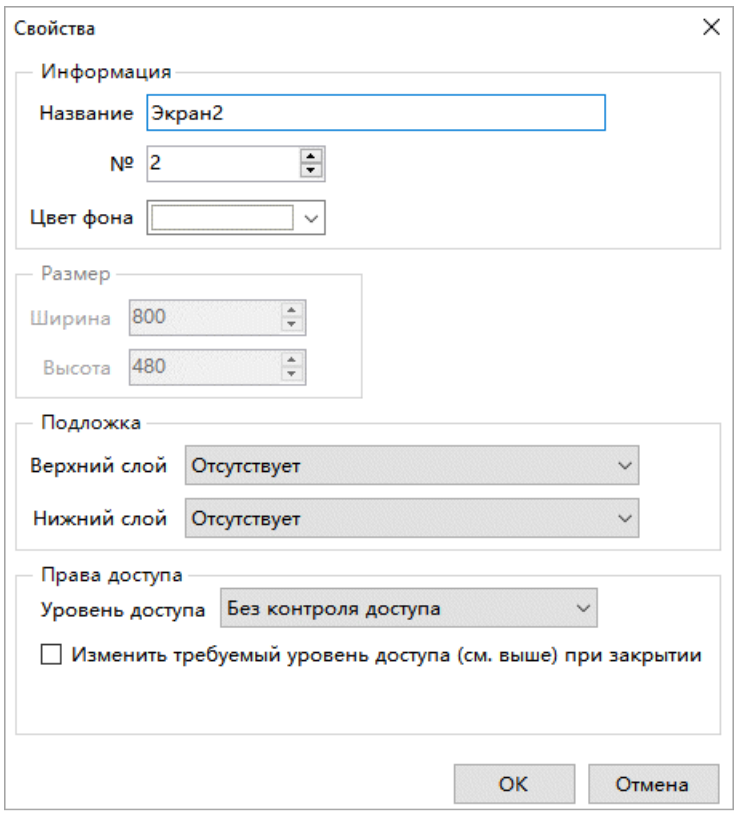


Рисунок 8.6 – Свойства пользовательского экрана

Таблица 8.1 – Настройки пользовательского экрана

Параметр	Описание
Название	Название экрана. Отображается в дереве проекта, используется в действии Переход на экран элементов Функциональная кнопка/Функциональная область и в меню настроек проекта на вкладке Общее (см. параметры Начальный экран и Экран заставки)
Номер экрана	Номер экрана. Используется в макросах в функции **ScreenJump**, в действии Переход на экран элементов Функциональная кнопка/Функциональная область и при работе с параметром Номер открытого экрана.
Цвет фона	Цвет фона экрана
Ширина и высота	Ширина и высота экрана. Определяются разрешением выбранной модификации панели оператора (недоступны для изменения)
Подложка	Выбор экрана и окна, содержимое которых будет отображаться в верхнем и нижнем слое данного экрана. Порядок отображения слоев (от верхнего к нижнему): • слой элементов экрана; • слой верхней подложки; • слой нижней подложки. В редакторе визуализации элементы подложки отображаются более серыми по сравнению с элементами самого экрана
Уровень доступа	Уровень доступа, необходимый для перехода на данный экран
Изменить требуемый уровень доступа при закрытии	Если установлена галочка, то после перехода с данного экрана на другой экран, требуемый для перехода на данный экран уровень доступа будет изменен на заданный

Максимальное количество пользовательских экранов в проекте – **5000**.

Для добавления пользовательского окна следует нажать **ПКМ** на узел **Пользовательские окна** и использовать команду **Добавить**. Откроется окно настроек:

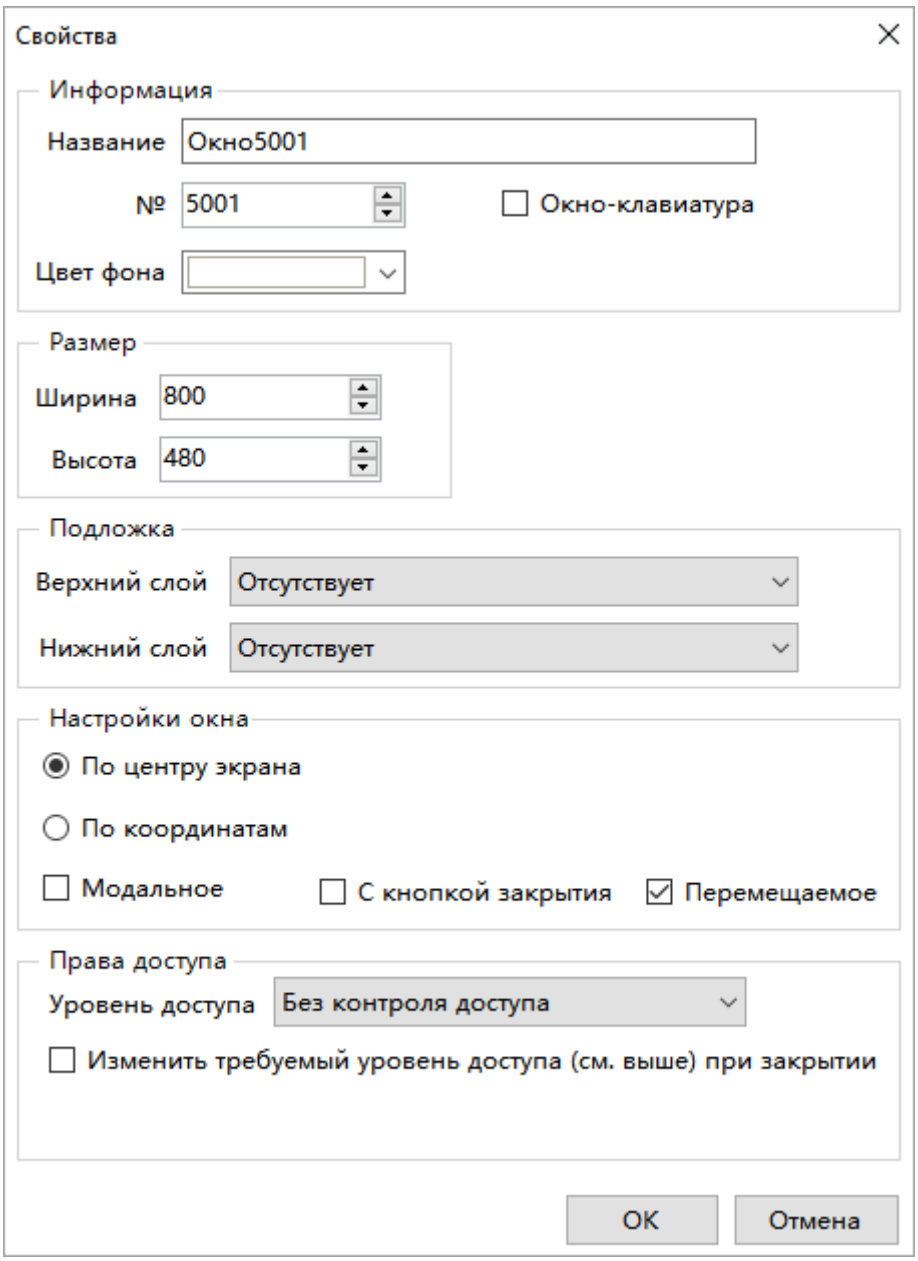


Рисунок 8.7 – Свойства пользовательского окна

Большая часть настроек аналогична настройкам экрана. В таблице приведены настройки, присутствующие только у окна:

Таблица 8.2 – Настройки пользовательского окна

Параметр	Описание
Окно-клавиатура	В случае установки галочки создаваемое окно может использоваться в качестве клавиатуры ввода для элементов типа **Цифровой ввод** и **Текстовый ввод**
По центру экрана	Окно будет открываться по центру экрана
По координатам	Окно будет открываться по заданным координатам (которые соответствуют положению левой верхней точки открытого окна)
Модальное	В случае установки галочки элементы на экране, поверх которого открыто данное окно, становятся недоступны для нажатия до закрытия окна
С кнопкой закрытия	В случае установки галочки в верхнем правом углу окна будет отображаться кнопка закрытия
Перемещаемое	В случае установки галочки открытое окно можно перемещать по экрану, зажав курсор в любом свободном месте окна

Максимальное количество пользовательских окон в проекте – **5000**.

Для изменения настроек созданного ранее экрана или окна следует нажать на нем **ПКМ** в дереве проекта, или на его фон в редакторе визуализации и использовать команду **Свойства**.

В контекстном меню, открываемом при нажатии на **ПКМ** в дереве проекта на экран/окно, также доступны следующие команды:

- **Копировать** – копирует объект в буфер обмена. Для вставки объекта следует нажать **ПКМ** на узел **Пользовательские экраны** или **Пользовательские окна**, и использовать команду **Вставить**;
- **Закрыть** – закрывает вкладку экрана/окна в редакторе визуализации;
- **Удалить** – удаляет данный экран/окно.

Объекты и команды, связанные с макросами, описаны в пункте Макросы.

Объекты библиотек описаны в пункте Библиотеки.

Узел **Устройства** описан в пункте **Настройка обмена с другими устройствами**.

8.4.1 Системные экраны и окна

Узел **Системные экраны** включает в себя экраны, используемые в экранном конфигураторе панели. Узел **Системные окна** включает в себя окна, связанные с системными экранами, а также окна авторизации пользователей, диагностических сообщений и клавиатур.

Создание и удаление системных экранов и окон не доступно для пользователя. Внешний вид существующих экранов и окон может быть изменён в проекте панели (см. в дереве проекта узел «**Системные экраны**»).

ПРИМЕЧАНИЕ
Некорректные изменения могут нарушить нормальную работу экранного конфигуратора.

Также для системных экранов и окон доступны те же команды контекстного меню, что и для пользовательских экранов и окон. Вставка скопированного в буфер обмена системного экрана/окна возможна только в узел **Пользовательские экраны/Пользовательские окна** соответственно.

Таблица 8.3 – Список системных экранов

№	Название	Описание
20001	Информация	Информация о панели – модификация, версия ПО и т. д.
20002	Системное время	Установка системного времени
20003	Сетевые настройки	Изменение сетевых настроек (только для панелей расширенных модификаций)
20004	Пароли	Изменение паролей

Таблица 8.4 – Список системных окон

№	Название	Способ открытия
25001	Авторизация	• вручную пользователем; • из окна **Ограничение доступа**; • при попытке перехода на экран, для которого не хватает уровня доступа (в случае установки галочки **Открыть окно авторизации, если не хватает уровня доступа** в настройках действия **Переход на экран** элемента **Функциональная кнопка** и **Функциональная область**)
25002	Ограничение доступа	При нажатии на элемент, для которого не хватает уровня доступа (в случае установки галочки **Открыть окно авторизации, если не хватает уровня доступа** в настройках элемента)
25003	Ошибки обмена	От системного бита **SPSB48**

Продолжение таблицы 8.4

№	Название	Способ открытия
25005	Системное предупреждение	От системного бита **SPSB3**
25006	Подтверждение выполнения	При нажатии на элемент, в настройках которого установлена галочка **Подтверждение операции**
25014	Изменение пароля	Вручную пользователем
25015	Экспорт данных на USB-накопитель	В случае экспорта, настроенного в элементе (например, элементе **Тренд**)
25017	Результат изменения сетевых настроек	В случае изменения сетевых настроек на системном экране **20003**
25018	Результат изменения системного времени	В случае изменения системного времени на системном экране **20002**
25111	Изменение конфигурации OPC UA Server	В случае изменения сетевых настроек или логинов/паролей пользователей
25007	KeyBoard_Asc_01U	Текстовая клавиатура с верхним регистром клавиш, стиль 1
25008	KeyBoard_Asc_01L	Текстовая клавиатура с нижним регистром клавиш, стиль 1
25009	KeyBoard_Num_01UL	Цифровая клавиатура с отображением минимально и максимально допустимого значения, стиль 1
25010	KeyBoard_Num_01	Цифровая клавиатура, стиль 1
25011	KeyBoard_Hex_01	Цифровая клавиатура с HEX-символами, стиль 1
25012	KeyBoard_Asc_02U	Текстовая клавиатура с верхним регистром клавиш, стиль 2
25013	KeyBoard_Asc_02L	Текстовая клавиатура с нижним регистром клавиш, стиль 2
25100	KeyBoard_Asc_03U	Текстовая клавиатура с верхним регистром клавиш, стиль 3
25101	KeyBoard_Asc_03L	Текстовая клавиатура с нижним регистром клавиш, стиль 3
25102	KeyBoard_Asc_04U	Текстовая клавиатура с верхним регистром клавиш, стиль 4
25103	KeyBoard_Asc_04L	Текстовая клавиатура с нижним регистром клавиш, стиль 4
25104	KeyBoard_Num_02UL	Цифровая клавиатура с отображением минимально и максимально допустимого значения, стиль 2
25105	KeyBoard_Hex_02	Цифровая клавиатура с HEX-символами, стиль 2
25106	KeyBoard_Num_02	Цифровая клавиатура, стиль 2
25112	KeyBoard_Asc_01Ru_U	Текстовая клавиатура с верхним регистром клавиш и поддержкой кириллицы, стиль 1
25113	KeyBoard_Asc_01Ru_L	Текстовая клавиатура с нижним регистром клавиш и поддержкой кириллицы, стиль 1
25114	KeyBoard_Asc_02Ru_U	Текстовая клавиатура с верхним регистром клавиш и поддержкой кириллицы, стиль 2
25115	KeyBoard_Asc_02Ru_L	Текстовая клавиатура с нижним регистром клавиш и поддержкой кириллицы, стиль 2
25116	KeyBoard_Asc_03Ru_U	Текстовая клавиатура с верхним регистром клавиш и поддержкой кириллицы, стиль 3
25117	KeyBoard_Asc_03Ru_L	Текстовая клавиатура с нижним регистром клавиш и поддержкой кириллицы, стиль 3
25118	KeyBoard_Asc_04Ru_U	Текстовая клавиатура с верхним регистром клавиш и поддержкой кириллицы, стиль 4
25119	KeyBoard_Asc_04Ru_L	Текстовая клавиатура с нижним регистром клавиш и поддержкой кириллицы, стиль 4

8.4.2 Библиотеки

Узел **Библиотеки** содержит ресурсы проекта:

- тексты локализации;
- переменные;
- изображения;
- аудиофайлы (только для модификаций СП210-07-Р, СП210-10-Р, СП210-15-Р).

Для перехода в библиотеку следует два раза нажать **ЛКМ** на узел нужной библиотеки в дереве проекта или использовать команду меню **Инструменты**.

8.4.2.1 Библиотека локализации

Библиотека локализации содержит тексты, привязываемые к графическим элементам проекта, с их переводом на разные языки. В проекте может быть использовано до 8 языков. Для переключения языка в процессе работы проекта используется системный регистр **SPFW260**.

Библиотека содержит две вкладки: **Метки** и **Мультиязычность**.

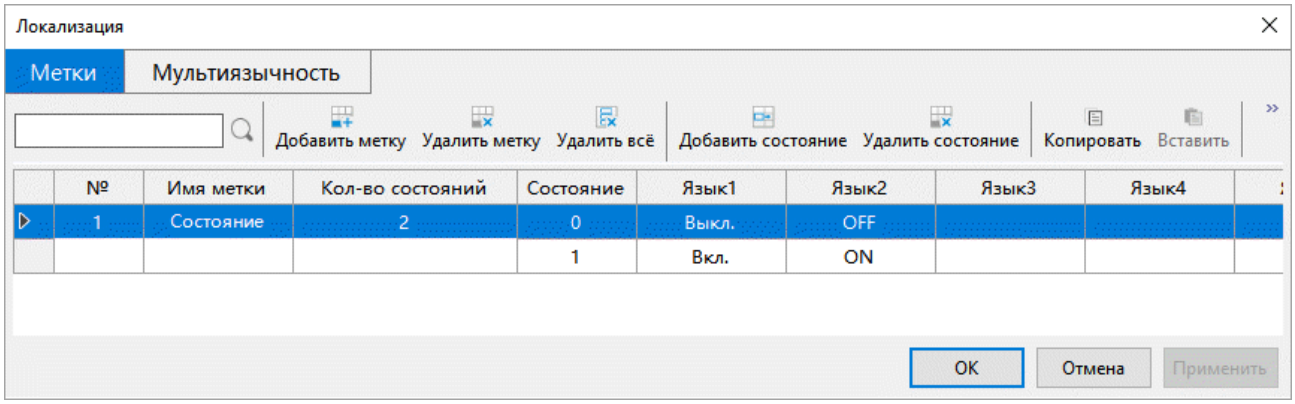


Рисунок 8.8 – Вкладка Метки

Метки используются для создания текстов элементов, которые имеют 2 и более состояния – например, **Индикатор**, **Переключатель, Переключатель с индикацией, Многопозиционный индикатор, Многопозиционный переключатель** и т. д.

Мультиязычные тексты используются для создания отдельных текстов элементов **Статический текст, Динамический текст, Таблица данных** и т. д.

Мультиязычные тексты распределяются по таблицам.

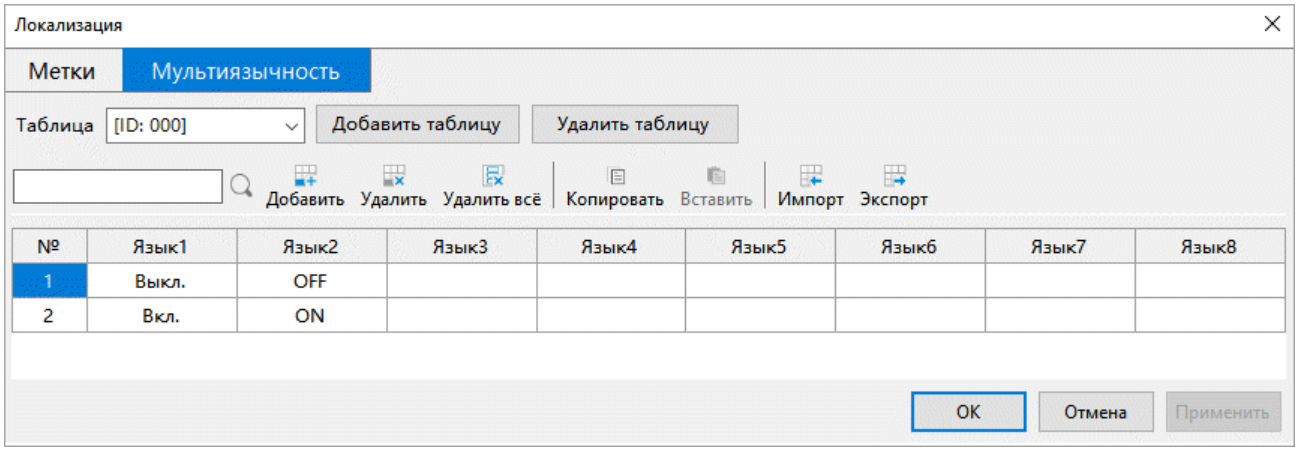


Рисунок 8.9 – Вкладка Мультиязычность

Для создания, удаления, копирования и вставки меток, их состояний и мультиязычных текстов используются соответствующие команды. Строка поиска позволяет осуществить поиск по меткам и мультиязычным текстам.

Команда **Экспорт** позволяет выгрузить метки и мультиязычные тексты в виде файла формата .xlsx с расширением .xlsx, .xls или .csv (расширение не влияет на формат файла). Команда **Импорт** позволяет загрузить файл.

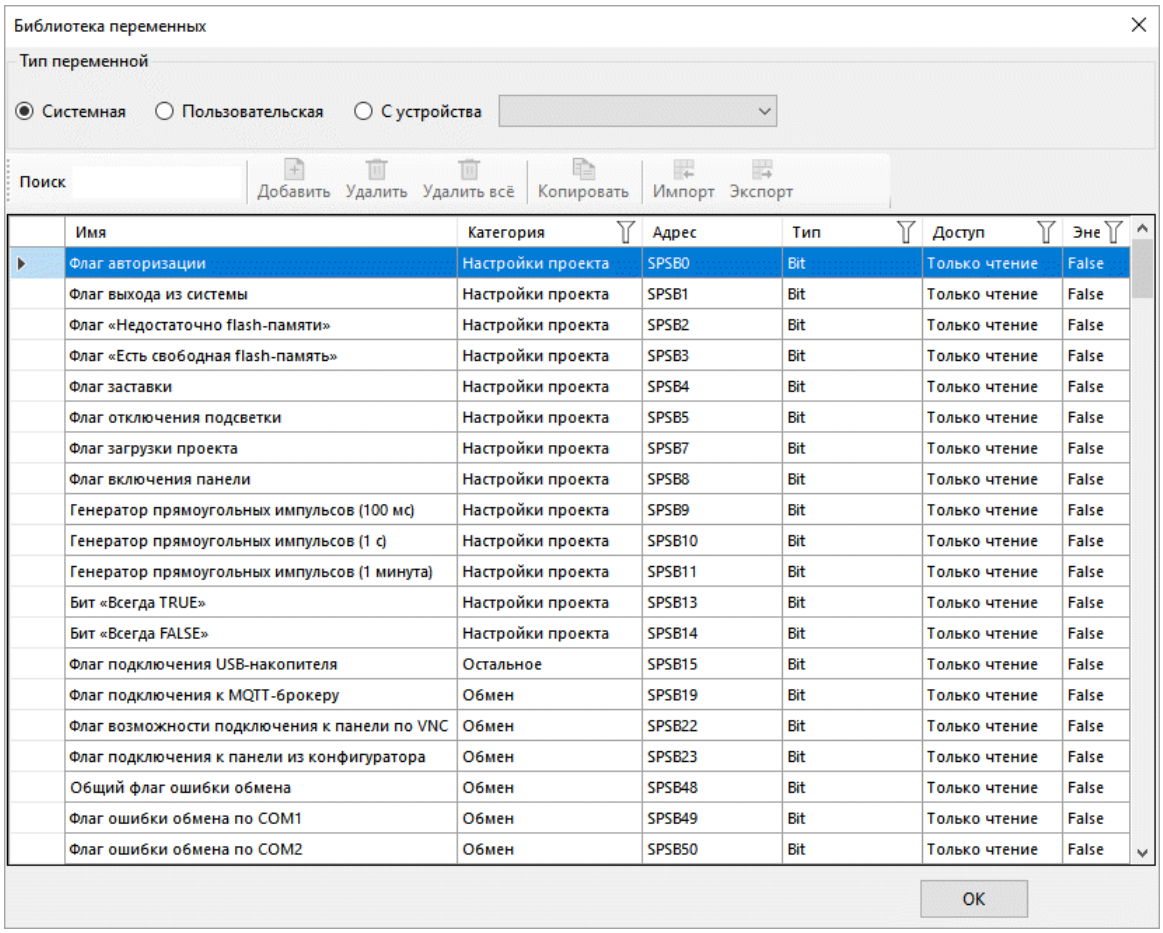
8.4.2.2 Библиотека переменных

Библиотека переменных содержит переменные, привязываемые к графическим элементам и некоторым настройкам проекта.

Библиотека содержит три вкладки: системные переменные, пользовательские переменные и переменные устройств.

Системные переменные

Системные переменные представляют собой предопределённый набор переменных, позволяющих считывать и записывать параметры панели. Такие переменные не могут быть созданы, удалены или отредактированы пользователем. Системные переменные размещаются в областях памяти **SPSB** (системные биты), **SPSW** (системные оперативные регистры) и **SPFW** (системные энергонезависимые регистры).



Имя	Категория	Адрес	Тип	Доступ	Эне
Флаг авторизации	Настройки проекта	SPSB0	Bit	Только чтение	False
Флаг выхода из системы	Настройки проекта	SPSB1	Bit	Только чтение	False
Флаг «Недостаточно flash-памяти»	Настройки проекта	SPSB2	Bit	Только чтение	False
Флаг «Есть свободная flash-память»	Настройки проекта	SPSB3	Bit	Только чтение	False
Флаг заставки	Настройки проекта	SPSB4	Bit	Только чтение	False
Флаг отключения подсветки	Настройки проекта	SPSB5	Bit	Только чтение	False
Флаг загрузки проекта	Настройки проекта	SPSB7	Bit	Только чтение	False
Флаг включения панели	Настройки проекта	SPSB8	Bit	Только чтение	False
Генератор прямоугольных импульсов (100 мс)	Настройки проекта	SPSB9	Bit	Только чтение	False
Генератор прямоугольных импульсов (1 с)	Настройки проекта	SPSB10	Bit	Только чтение	False
Генератор прямоугольных импульсов (1 минута)	Настройки проекта	SPSB11	Bit	Только чтение	False
Бит «Всегда TRUE»	Настройки проекта	SPSB13	Bit	Только чтение	False
Бит «Всегда FALSE»	Настройки проекта	SPSB14	Bit	Только чтение	False
Флаг подключения USB-накопителя	Остальное	SPSB15	Bit	Только чтение	False
Флаг подключения к MQTT-брокеру	Обмен	SPSB19	Bit	Только чтение	False
Флаг возможности подключения к панели по VNC	Обмен	SPSB22	Bit	Только чтение	False
Флаг подключения к панели из конфигуратора	Обмен	SPSB23	Bit	Только чтение	False
Общий флаг ошибки обмена	Обмен	SPSB48	Bit	Только чтение	False
Флаг ошибки обмена по COM1	Обмен	SPSB49	Bit	Только чтение	False
Флаг ошибки обмена по COM2	Обмен	SPSB50	Bit	Только чтение	False

Рисунок 8.10 – Системные переменные

Таблица системных переменных содержит следующие столбцы:

- имя переменной;
- категорию;
- адрес;

- тип данных;
- тип доступа (только чтение/только запись/чтение и запись);
- флаг энергонезависимости (принадлежности к области **SPFW**) переменной.

С помощью команды  можно осуществить фильтрацию таблицы по категории, типу данных, типу доступа и флагу энергонезависимости.

С помощью строки поиска можно осуществить поиск переменной по имени и категории.

Список системных переменных приведён в пункте 13.1.

Пользовательские переменные

Пользовательские переменные создает разработчик проекта панели.

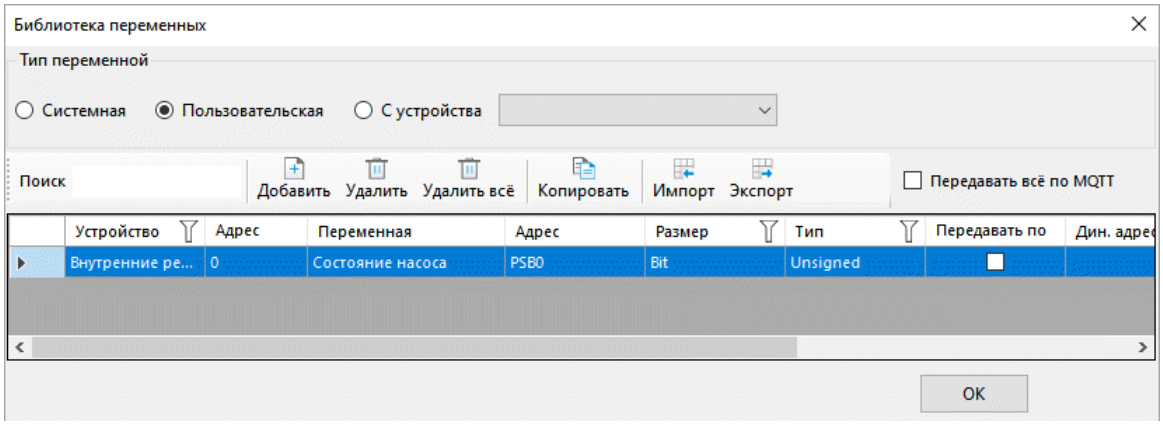


Рисунок 8.11 – Список пользовательских переменных

Галочка **Передавать по MQTT** экспортирует выбранную переменную в список параметров протокола MQTT, а галочка **Передавать всё по MQTT** осуществляет экспорт всех переменных сразу (только для расширенных модификаций панелей (-P).

Команда **Экспорт** позволяет выгрузить переменные в виде файла формата .xlsx. Команда **Импорт** позволяет загрузить файл.

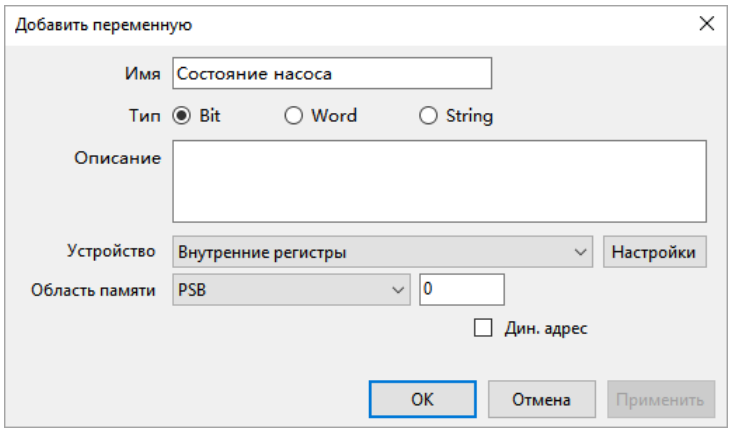


Рисунок 8.12 – Создание пользовательской переменной

При создании переменой следует указать:

- имя переменной;
- тип данных;
- описание;
- место хранения значения. Например, память одного из slave-устройств, подключенных к панели, или ее внутренние регистры;

- адрес slave-устройства (если переменная хранится в его памяти);
- формат;
- динамический адрес.

С помощью строки поиска можно осуществить поиск переменной по имени и области памяти.

Переменные устройств

В списке переменных устройств отображаются переменные, импортированные по протоколу CODESYS V3 (PLC Handler) или OPC UA Client. Для выбора переменных конкретного устройства используется выпадающий список. Пользователь может удалить из списка те переменные, которые не требуются в его проекте.

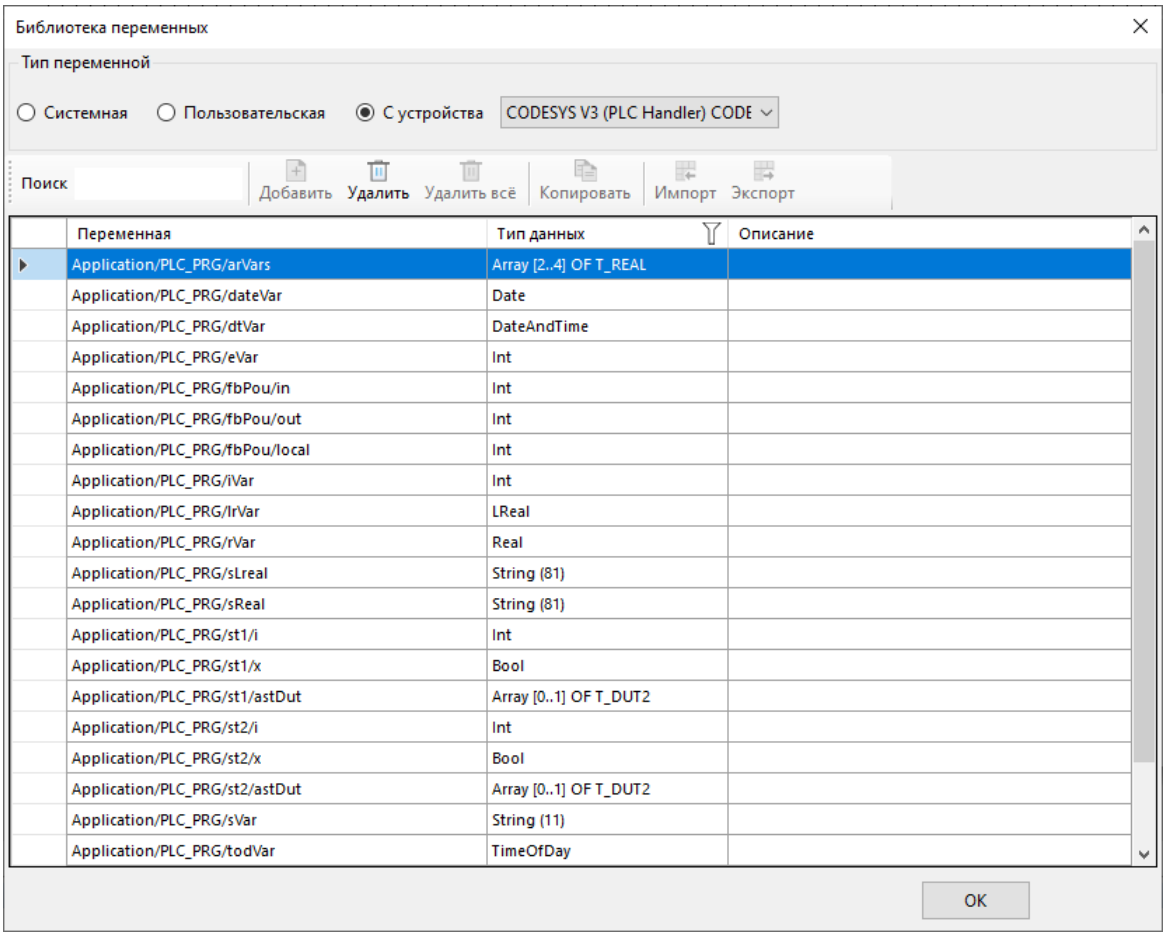


Рисунок 8.13 – Переменные устройств

Таблица переменных содержит следующие столбцы:

- путь переменной;
- тип данных;
- описание.

С помощью строки поиска можно осуществить поиск переменной по имени (имя – последний сегмент пути; по другим сегментам пути осуществить поиск нельзя).

8.4.2.3 Библиотека изображений

Библиотека изображений содержит изображения, которые могут использоваться при создании проекта. У большинства графических элементов выбор изображения осуществляется на вкладке параметров **Вид**. Кроме того, существует несколько элементов, рассчитанных на отображение графических файлов – **Изображение** и **Динамическое изображение**.

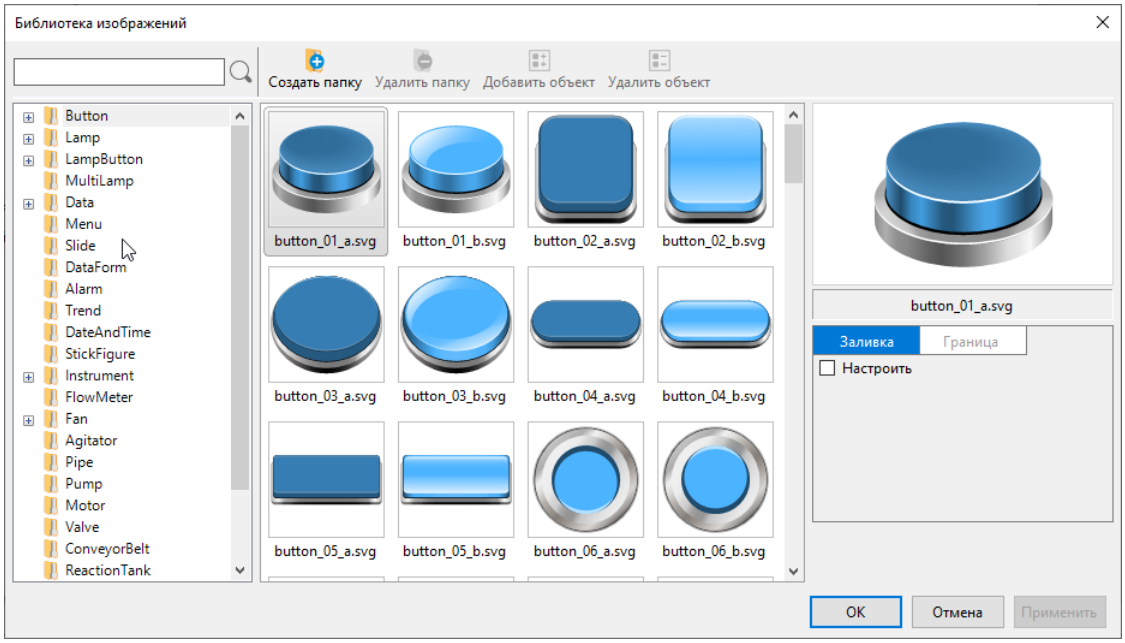


Рисунок 8.14 – Библиотека изображений

Окно библиотеки изображений состоит из трёх областей:

- дерева папок библиотеки;
- списка изображений выбранной в данный момент папки;
- списка настроек выбранного в данный момент изображения.

Создание/удаление/переименование папок и добавление/удаление изображений происходит с помощью команд контекстного меню дерева папок. Ярлыки команд также продублированы над списком изображений.

Добавление изображений возможно только в созданные пользователем папки.

Удаление папки со стандартными изображениями и самих изображений пользователю не доступно.

Поддерживаемые форматы изображений: .svg, .png, .jpg, .jpeg, .bmp.

Фактически библиотека изображений представляет собой папку **\Resource\Pic** в директории установки конфигуратора. В зависимости от настроек операционной системы, для работы с библиотекой изображений может потребоваться запуск конфигуратора с правами администратора.

Для стандартных **.svg** изображений доступны настройки типа заливки (однотонная или градиент) и цвета заливки. Для некоторых из них также доступны настройки типа и цвета заливки контура. В случае создания пользовательских изображений следует учитывать:

- цвету элемента соответствует параметр SVG с id="color1";
- цвету контура – параметр с id="color2".

Для удобного просмотра исходного кода используйте SVG Formatter или другой аналогичный редактор.

8.4.2.4 Библиотека аудиофайлов

Библиотека аудиофайлов доступна только для модификаций СП210-07-Р, СП210-10-Р, СП210-15-Р. Библиотека содержит аудиозаписи, которые могут воспроизводиться при нажатии на элемент **Переключатель**, **Переключатель с индикацией**, **Ввод символа**, **Функциональная кнопка**, а также при активации тревог.

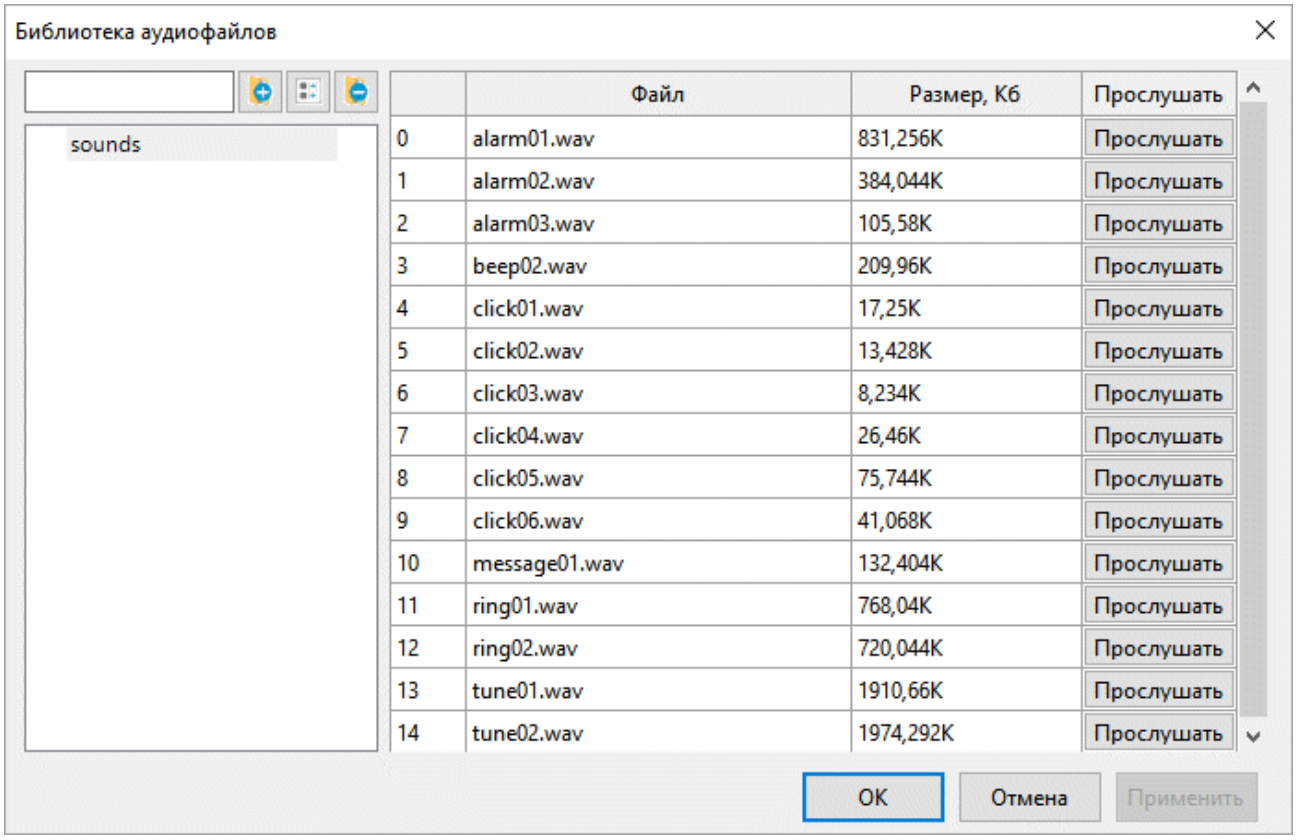


Рисунок 8.15 – Библиотека аудиофайлов

Окно библиотеки аудиофайлов состоит из двух областей:

- дерева папок библиотеки;
- списка файлов выбранной в данный момент папки.

Для создания папки используется команда контекстного меню дерева папок. Для импорта существующей папки используется ярлык . Для добавления аудиофайла нужно нажать **ПКМ** на область списка файлов и использовать команду **Добавить файл** или использовать ярлык . Для удаления папки или файла используется соответствующая команда контекстного меню или ярлык .

Поддерживаемые форматы аудиофайлов: **.mp3**, **.wav**.

Размер файла не должен превышать 3 Мб.

В списке файлов отображаются названия, размеры и кнопки воспроизведения.

Фактически библиотека аудиофайлов представляет собой папку **\Resource \Sound** в директории установки конфигуратора. В зависимости от настроек операционной системы для работы с библиотекой изображений может потребоваться запуск конфигуратора с правами администратора.

8.5 Редактор визуализации

Редактор визуализации позволяет добавлять на экраны и окна проекта графические элементы и настраивать их параметры.

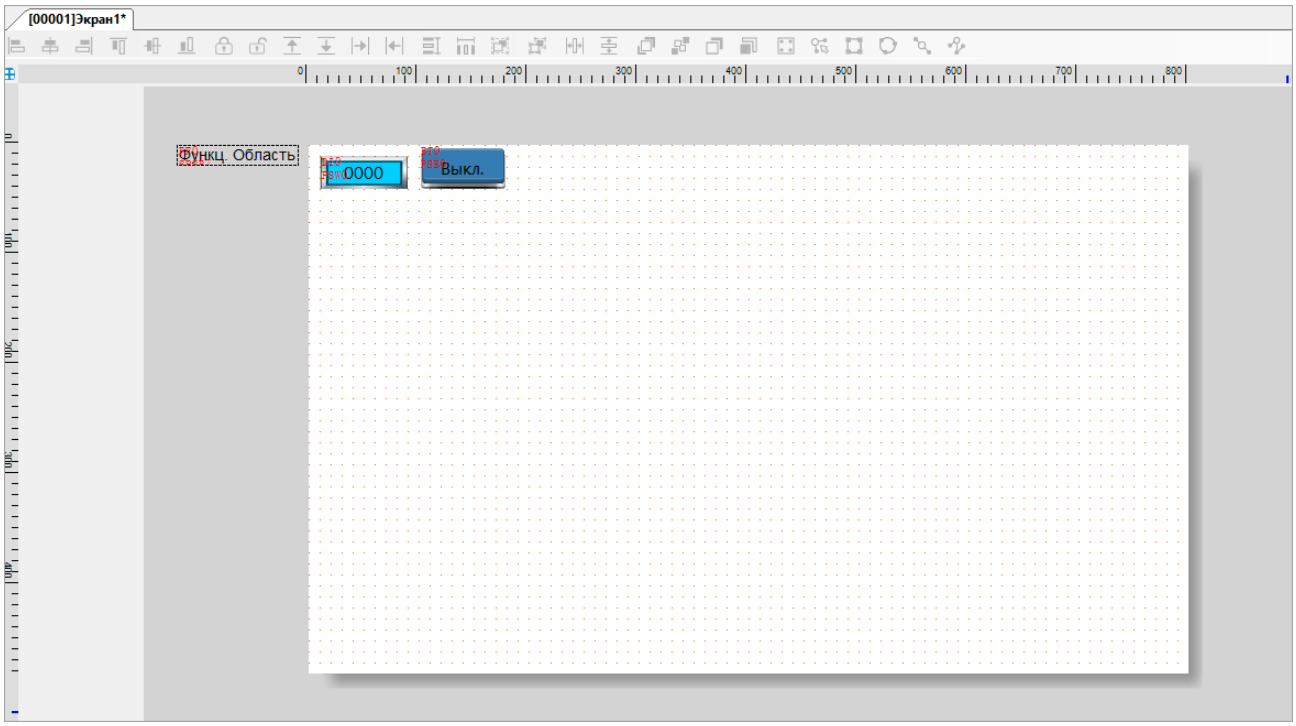


Рисунок 8.16 – Внешний вид редактора визуализации

Графические элементы доступны на панели меню в разделах **Элементы** и **Графика**, а также на панели элементов. В зависимости от типа элемента добавление производится разными способами:

- для большинства элементов размещение на экране/окне производится путём выбора элемента и однократного нажатия **ЛКМ** в нужной области экрана/окна;
- для элементов **Прямоугольник** и **Окружность** при зажатой **ЛКМ** следует отрисовать контур элемента;
- для элементов **Линия**, **Дуга**, **Ломаная/Многоульник**, **Анимация перемещения** и **Трубопровод** каждое нажатие **ЛКМ** будет приводить к созданию новой опорной точки элемента. Для завершения следует два раза нажать **ЛКМ**.

После добавления элемента автоматически открывается окно свойств.

Элементы могут быть размещены за пределами «видимой» части экрана/окна. Это удобно для размещения невизуализируемых элементов — таких, как **Функциональная область**, **Подсветка** и **Зуммер**.

Над элементом отображается идентификатор, имена привязанных переменных и адреса привязанных битов/регистров. Управление видимостью и настройками отображения информации осуществляется в меню **Инструменты – Настройки конфигуратора**.

Для перемещения выделенного элемента по экрану следует зажать **ЛКМ** и переместить курсор в нужное место. Для изменения размера следует зажать **ЛКМ** на одной из опорных точек элемента и переместить курсор на нужное расстояние. Для некоторых элементов доступна опорная точка (жёлтого цвета), позволяющая вращать элемент вокруг своей оси.

При работе с элементами можно использовать команды меню **Правка**.

8.5.1 Вкладки

В верхней части редактора размещаются вкладки открытых экранов/окон. При нажатии **ПКМ** на вкладку открывается контекстное меню, в котором доступны команды:

- закрытия текущей вкладки;
- закрытия всех вкладок, кроме текущей;
- закрытия всех вкладок.

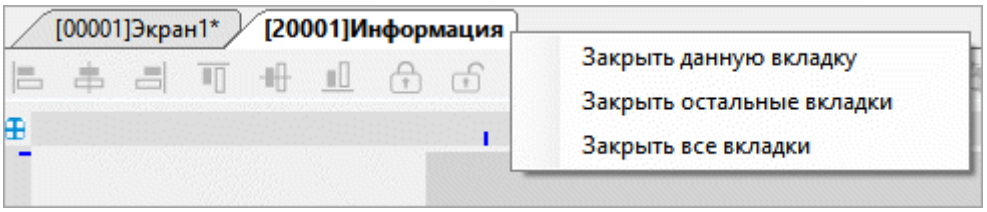


Рисунок 8.17 – Контекстное меню вкладки

Закрыть текущую открытую вкладку также можно с помощью ярлыка ✕.

С помощью ярлыка ▾ можно просмотреть список всех открытых вкладок.

8.5.2 Панель команд

Под списком вкладок отображается панель команд редактора визуализации. Часть команд доступна только при одновременном выделении нескольких элементов. Для этого нужно при зажатой **ЛКМ** обвести элементы рамкой или последовательно выделять элементы нажатием **ЛКМ** при зажатой клавише **Ctrl**.

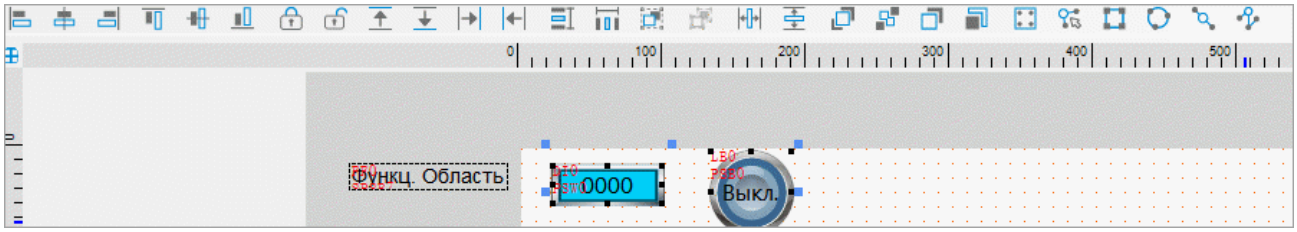


Рисунок 8.18 – Панель команд

Таблица 8.5 – Список команд редактора визуализации

Ярлык	Действие
	Выровнять группу элементов по левому краю
	Выровнять группу элементов по центру
	Выровнять группу элементов по правому краю
	Выровнять группу элементов по верхнему краю
	Выровнять группу элементов по середине (по вертикали)
	Выровнять группу элементов по нижнему краю
	Закрепить выделенные элементы. Закрепленные элементы невозможно перемещать по экрану, вращать и изменять в размерах
	Открепить элемент или группу элементов
	Переместить выделенные элементы на одну ячейку сетки вверх
	Переместить выделенные элементы на одну ячейку сетки вниз

Продолжение таблицы 8.5

Ярлык	Действие
	Переместить выделенные элементы на одну ячейку сетки вправо
	Переместить выделенные элементы на одну ячейку сетки влево
	Центрировать группу элементов по вертикали
	Центрировать группу элементов по горизонтали
	Сгруппировать элементы. Сгруппированные элементы перемещаются по экрану и масштабируются как один цельный объект
	Разгруппировать элементы
	Сделать выделенные элементы равной ширины
	Сделать выделенные элементы равной высоты
	Переместить выделенные элементы в верхний слой
	Переместить выделенные элементы в нижний слой
	Переместить выделенные элементы на слой выше
	Переместить выделенные элементы на слой ниже
	Разместить выделенные элементы: • прямоугольником (горизонтально); • по точкам; • прямоугольником (вертикально); • по окружности; • по прямой; • по ломаной. После выбора команды следует на экране/окне при зажатой **ЛКМ** начертить соответствующую геометрическую фигуру. Выделенные элементы будут расставлены в соответствии с контуром фигуры
	
	
	
	
	

8.5.3 Сетка и масштаб

По умолчанию в редакторе визуализации отображается сетка, которая помогает позиционировать элементы. При перемещении элементы «притягиваются» к точкам сетки.

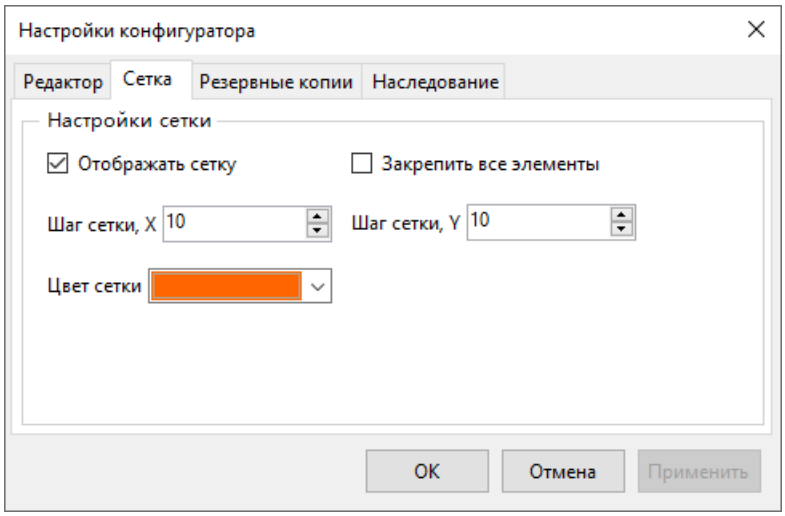


Рисунок 8.19 – Сетка

Цвет, шаг точек сетки и расстояние между ними настраиваются в меню **Инструменты – Настройки конфигуратора**.

Включение/отключение видимости сетки и изменение масштаба редактора визуализации (приближение/отдаление) производится с помощью ярлыков в строке информации.

Также изменить масштаб можно с помощью поворота колёсика мыши при зажатой клавише **Ctrl**.

8.5.4 Слои

Элементы могут располагаться в разных слоях.

Для перемещения элементов между слоями используются соответствующие ярлыки панели команд или команды контекстного меню элемента.

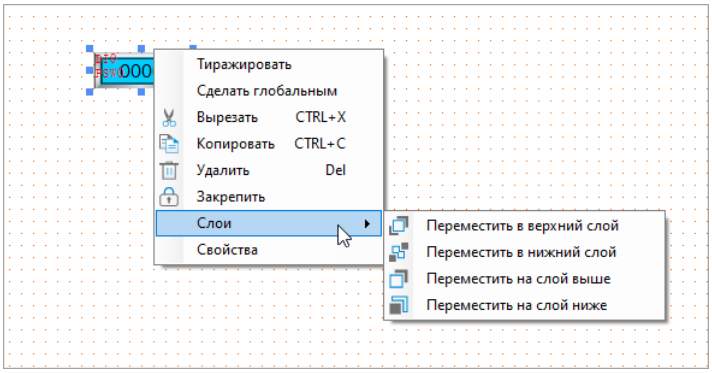


Рисунок 8.20 – Слои

Элементы, размещённые в верхних слоях, перекрывают элементы, размещённые в нижних слоях.

8.5.5 Контекстное меню элемента

При нажатии **ПКМ** в свободном пространстве экрана/окна открывается меню настроек экрана.

При нажатии **ПКМ** на элемент открывается контекстное меню.

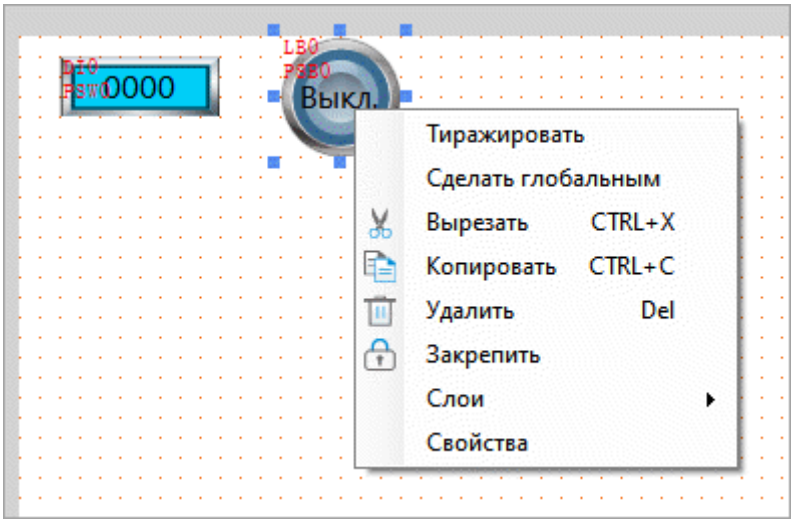


Рисунок 8.21 – Контекстное меню элемента

Таблица 8.6 – Команды контекстного меню элемента

Команды	Описание
Группировать Разгруппировать	Данные команды доступны только при выделении нескольких элементов. Дублируют соответствующие команды **панели команд**
Тиражировать	См. Тиражирование
Сделать глобальным Сделать локальным	Данные команды доступны только для элементов экранов и недоступны для элементов окон. По умолчанию элементы экрана являются локальными и отображаются только на данном экране. Глобальные элементы отображаются и функционируют на всех экранах проекта
Вырезать Копировать Вставить Удалить	Дублируют соответствующие команды меню **Правка**
Закрепить Открепить	Дублируют соответствующие команды **панели команд**
Слои	Дублируют соответствующие команды **панели команд**
Свойства	Переход к свойствам элемента

Команда **Тиражировать** позволяет создать копии выбранного элемента. При использовании команды открывается окно тиражирования, в котором необходимо указать следующие настройки:

- количество строк и столбцов для создаваемых копий. Это влияет на количество копий (например, если указать 2 столбца и 3 строки, то будет создано 5 копий для выбранного элемента);
- расстояние между создаваемыми элементами по оси X и Y;
- режим увеличения адреса «основного» параметра элемента (расположенного на вкладке свойств **Основное**) – по горизонтали или по вертикали;
- тип увеличиваемых адресов – регистры или биты регистров.

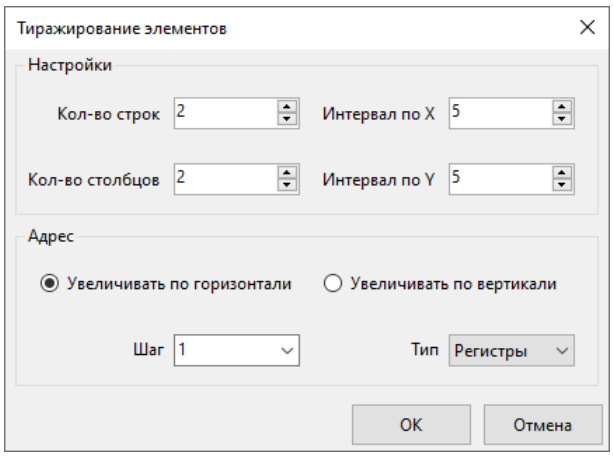


Рисунок 8.22 – Тиражирование

8.6 Карта адресов

Карта отображает адреса битов и регистров, используемых в проекте.

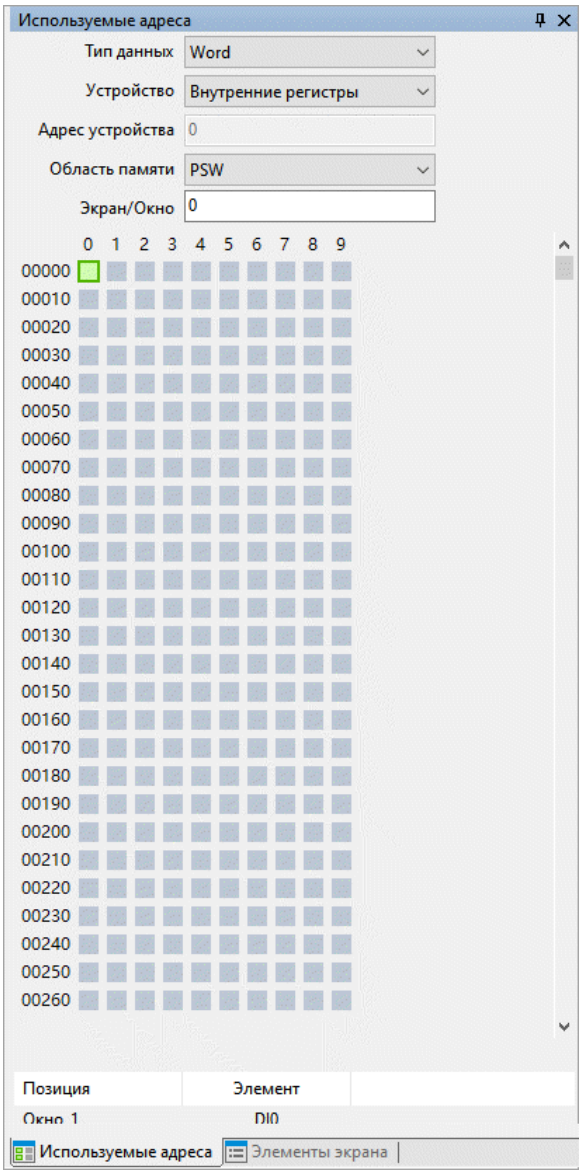


Рисунок 8.23 – Карта адресов

В верхней части окна находятся фильтры:

- по типу адреса (бит или регистр);
- по устройству (внутренние регистры или биты панели/регистры или биты одного из подключённых к панели slave-устройств);
- по области памяти.

В карту попадают только биты и регистры, привязанные к графических элементам. В карте отображаются элементы всех экранов и окон проекта (независимо от того, открыты ли они в данный момент).

Ввод адреса и нажатие **Enter** или нажатие **ЛКМ** на ячейке карты выделяет ячейку. В нижней части экрана отображаются все вхождения адреса выделенной ячейки в проекте с указанием порядкового номера экрана/окна и идентификатора элемента.

Одиночное нажатие на строку приводит к выделению элемента в редакторе визуализации, двойное – к открытию меню свойств элемента.

8.7 Список элементов

Список элементов отображает элементы открытого в редакторе визуализации экрана/окна с указанием типов и идентификаторов.

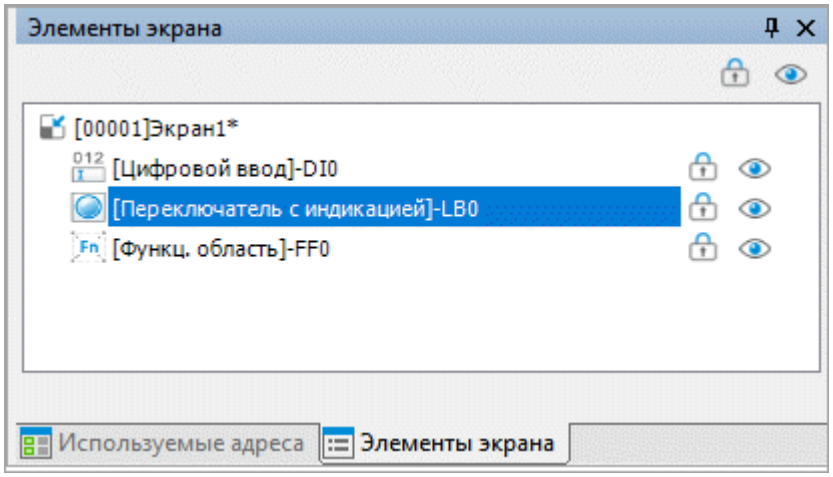


Рисунок 8.24 – Список элементов

Нажатие **ЛКМ** на элемент в списке приводит к его выделению.

Нажатие **ПКМ** открывает контекстное меню, содержащее команды **Вырезать, Копировать, Вставить, Удалить** и **Свойства**.

Для сгруппированного элемента использование команды **Свойства** приводит к открытию окна выбора меню свойств одного из элементов, входящего в состав группы.

С помощью команд / можно закрепить/открепить конкретный элемент или все элементы данного экрана/окна. С помощью команд / можно скрыть/вернуть видимость выбранного элемента или всех элементов данного экрана/окна.

8.8 Панель сообщений

Панель сообщений содержит две вкладки: **Журнал** и **Ошибки**.

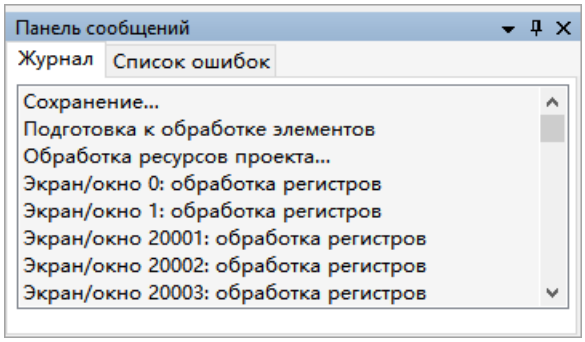


Рисунок 8.25 – Панель сообщений проекта

На вкладке **Журнал** отображаются информационные сообщения о результате компиляции, а также другие сообщения (например, сообщения об автоматическом сохранении проекта, завершении процедуры тиражирования элементов и т. д.).

На вкладке **Ошибки** отображаются сообщения об ошибках компиляции. Такие сообщения разделены на три группы: ошибки, предупреждения и информационные сообщения. Нажатие на заголовок группы приводит к установке фильтра сообщений данной группы. Сообщения данной вкладки в большинстве случаев связаны с ошибками в коде макросов. Нажатие **ЛКМ** на строку сообщения приводит к переходу к файлу, в котором были обнаружены ошибки.

8.9 Строка информации

Строка информации расположена в нижней части окна конфигуратора.

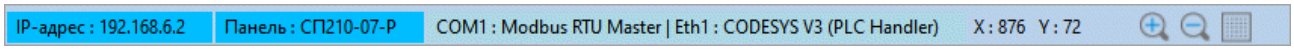
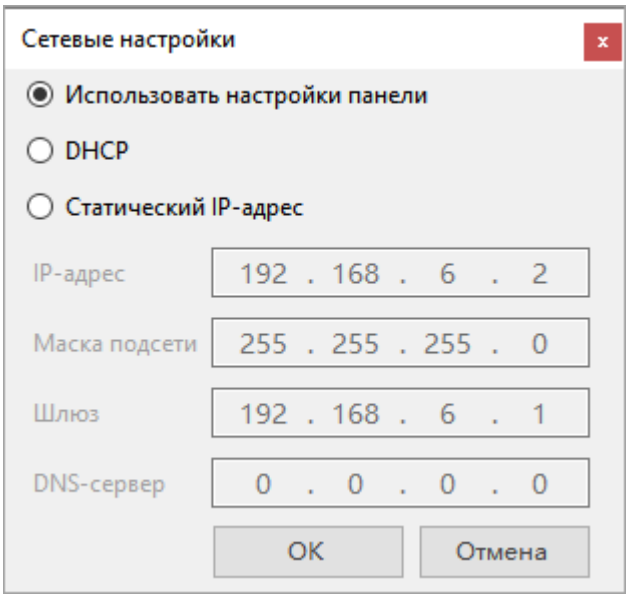
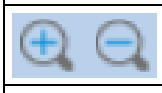


Рисунок 8.26 – Строка информации

Таблица 8.7 – Элементы строки информации

Элемент строки информации	Описание
IP-адрес : 192.168.6.2	IP-адрес панели. Двойное нажатие **ЛКМ** открывает окно сетевых настроек.  **Рисунок 8.27 – Меню сетевых настроек панели** Доступны три варианта: • Использовать настройки панели. • DHCP. • Статический IP-адрес. В двух последних вариантах заданные настройки будут записаны в панель при загрузке проекта.
Панель : СП210-07-P	Модель панели. Двойное нажатие **ЛКМ** открывает окно **Настройки проекта – Панель**.
COM1 : Modbus RTU Master | Eth1 : Modbus TCP Master	Протоколы обмена, настроенные для коммуникационных интерфейсов панели. Если для интерфейса настроено несколько протоколов, то отображается только первый из них
X : 694 Y : 468	Координаты курсора в редакторе визуализации
	Кнопки изменения масштаба редактора визуализации
	Кнопка отключения/включения видимости сетки в редакторе визуализации

8.10 Инструменты – Настройки конфигуратора

Меню **Инструменты – Настройки конфигуратора** содержит настройки редактора визуализации и автоматического сохранения файла проекта.

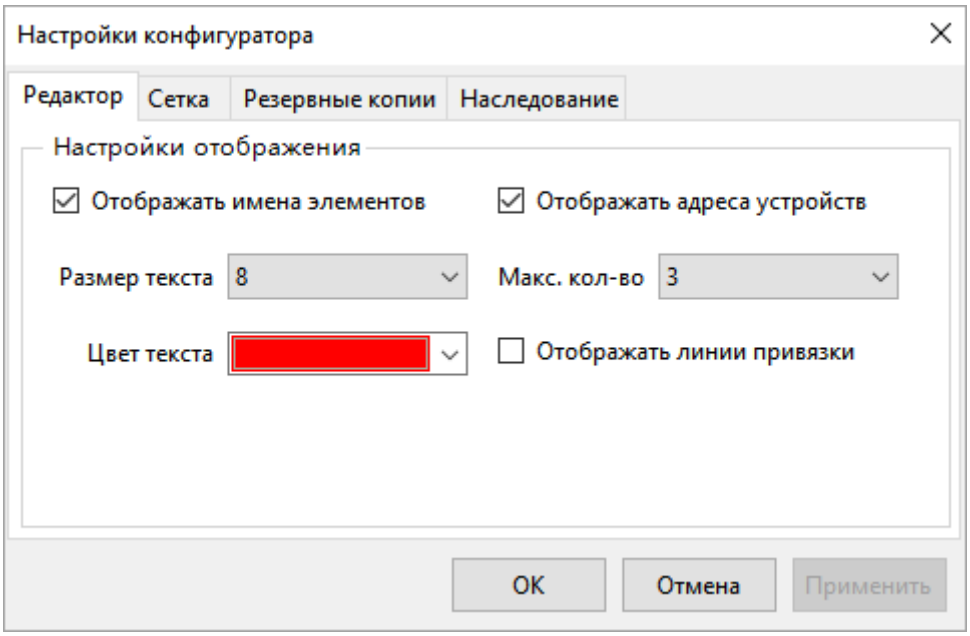


Рисунок 8.28 – Меню настроек конфигуратора

На вкладке **Редактор** доступны следующие настройки:

Таблица 8.8 – Настройки вкладки Редактор

Настройка	Описание
Отображать имена элементов	В случае установки галочки в элементе отображается его ID
Размер текста	Размер текста, используемого для отображения ID, имён переменных и адресов элемента
Цвет текста	Цвет текста, используемого для отображения ID, имён переменных и адресов элемента
Отображать адреса устройств	В случае установки галочки в элементе отображаются имена переменных и адреса битов/регистров, привязанных к элементу
Макс. кол-во	Максимальное количество имён переменных/адресов, отображаемых в элементе
Отображать линии привязки	В случае установки галочки при перемещении элемента в редакторе визуализации отображаются линии привязки (snaplines), указывающие на его положение относительно других элементов

На вкладке **Сетка** доступны следующие настройки:

Таблица 8.9 – Настройки вкладки Сетка

Настройка	Описание
Отображать сетку	В случае установки галочки в редакторе визуализации отображается сетка
Шаг сетки, X	Расстояние между точками сетки по оси X
Шаг сетки, Y	Расстояние между точками сетки по оси Y
Цвет сетки	Цвет точек сетки
Закрепить все элементы	В случае установки галочки закрепляются все элементы всех экранов и окон проекта. Закрепленные элементы невозможно перемещать по экрану, вращать и изменять в размерах

На вкладке **Резервные копии** доступны следующие настройки:

Таблица 8.10 – Настройки вкладки Резервные копии

Настройка	Описание
Макс. кол-во копий	Максимальное количество резервных копий проекта. После достижения этого количества при создании новой копии самая старая будет удалена. Изменение настройки вступает в силу после перезапуска конфигуратора
Включить автосохранение	В случае установки галочки с заданным периодом будет производиться создание резервной копии проекта
Директория	Кнопка перехода в директорию резервной копии проекта (папка **\BakDir** в директории установки конфигуратора)
Период сохранения	Период создания резервной копии проекта

Название файла резервной копии формируется следующим образом:

OWEN@<метка времени создания файла><номер резервной копии>@<название файла проекта>@BackUp.xjp

На вкладке **Наследование** доступна галочка **Наследовать настройки элементов**. При установленной галочке новый добавленный графический элемент будет иметь те же настройки, что и последний отредактированный элемент данного типа.

8.11 Меню «Файл»

Меню **Файл** содержит команды для работы с проектом и эмулятором, а также команду перехода к настройкам проекта.

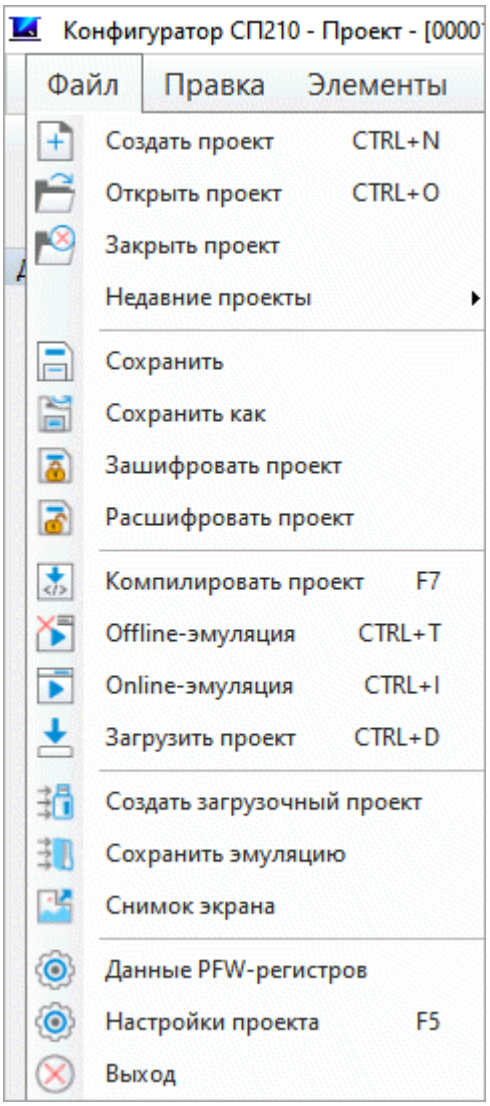


Рисунок 8.29 – Команды меню Файл

Таблица 8.11 – Список команд меню Файл

Команда	Описание
Создать проект (Ctrl+N)	Создание нового проекта. При создании проекта требуется выбрать модификацию панели, после чего откроется окно **Настройки проекта – Устройства**, в котором можно настроить коммуникационные интерфейсы панели
Открыть проект (Ctrl+O)	Открытие существующего проекта. Если в конфигураторе уже открыт проект, то нельзя открыть проект с тем же названием. С помощью фильтра указывается расширение файла проекта - **.xjp** (незашифрованный проект) или **.xep** (зашифрованный проект)
Закрыть проект	Закрытие проекта
Недавние проекты	Открытие одного из последних сохранённых проектов
Сохранить	Сохранение проекта в том же файле
Сохранить как	Сохранение проекта в виде нового файла

Продолжение таблицы 8.11

Команда	Описание
Зашифровать проект	Сохранение проекта с установкой пароля в виде файла формата **.xep**. Длина пароля должна составлять от 6 до 20 символов. Зашифрованный проект можно открыть в конфигураторе, но его содержимое недоступно для просмотра и редактирования. Единственные поддерживаемые действия – загрузка в панель и запуск в режиме эмуляции
Расшифровать проект	Расшифровка зашифрованного проекта, в ходе которой нужно ввести пароль, использованный при шифровании. Расшифрованный проект сохраняется в виде файла формата **.xjp**
Компилировать проект (F7)	Компиляция проекта, в рамках которой производится проверка на ошибки. Информация о результате компиляции и ошибках отображается на панели сообщений
Offline-эмуляция (Ctrl+T)	Запуск проекта на ПК в режиме offline-эмуляции
Online-эмуляция (Ctrl+I)	Запуск проекта на ПК в режиме online-эмуляции
Загрузить проект (Ctrl+D)	Загрузка проекта в панель или выгрузка проекта из панели.
Создать загрузочный проект	Сохранение проекта в виде файла формата **.dat**, который может быть загружен в панель с USB-накопителя
Сохранить эмуляцию	Сохранение online-эмуляции проекта в виде набора файлов. Это позволяет запустить эмуляцию на ПК, на котором не установлен конфигуратор.
Снимок экрана	Сохранение скриншотов выбранных экранов/окон проекта в виде графических файлов (формата **.png, .jpg** или **.bmp**) или одного общего файла формата **.pdf**
Данные PFW-регистров	Установка начальных значений для PFW-регистров панели.
Настройки проекта (F5)	Переход к настройкам проекта
Выход	Закрытие конфигуратора

8.12 Меню «Правка»

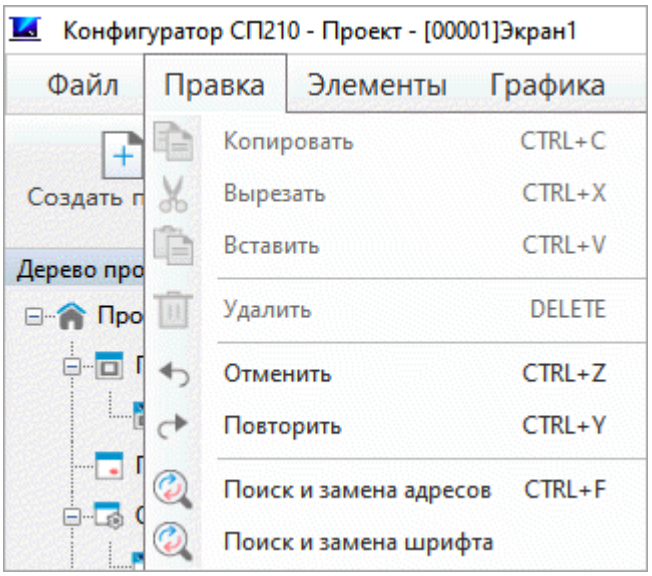


Рисунок 8.30 – Команды меню Правка

В меню **Правка** доступны команды, которые можно использовать в редакторе визуализации при работе с графическими элементами:

- копировать выделенные элементы в буфер обмена (Ctrl+C);
- «вырезать» (копировать с удалением) выделенные элементы в буфер обмена (Ctrl+X);
- вставить элементы из буфера обмена (Ctrl+V);
- удалить выделенные элементы (Delete);
- отменить последние изменения (Ctrl+Z);
- восстановить последние изменения (Ctrl+Y).

Кроме того, доступны команды **Поиск и замена адресов** (Ctrl+F) и **Поиск и замена шрифта.**

8.12.1 Поиск и замена адресов

Команда **Поиск и замена адресов** позволяет найти и заменить адреса регистров или переменные, привязанные к элементам. Например, если в процессе работы возникла необходимость использовать для размещения параметров вместо области **PSW** область **PFW** (энергонезависимую) или поменять область памяти Modbus, в которой находятся параметры опрашиваемого slave-устройства.

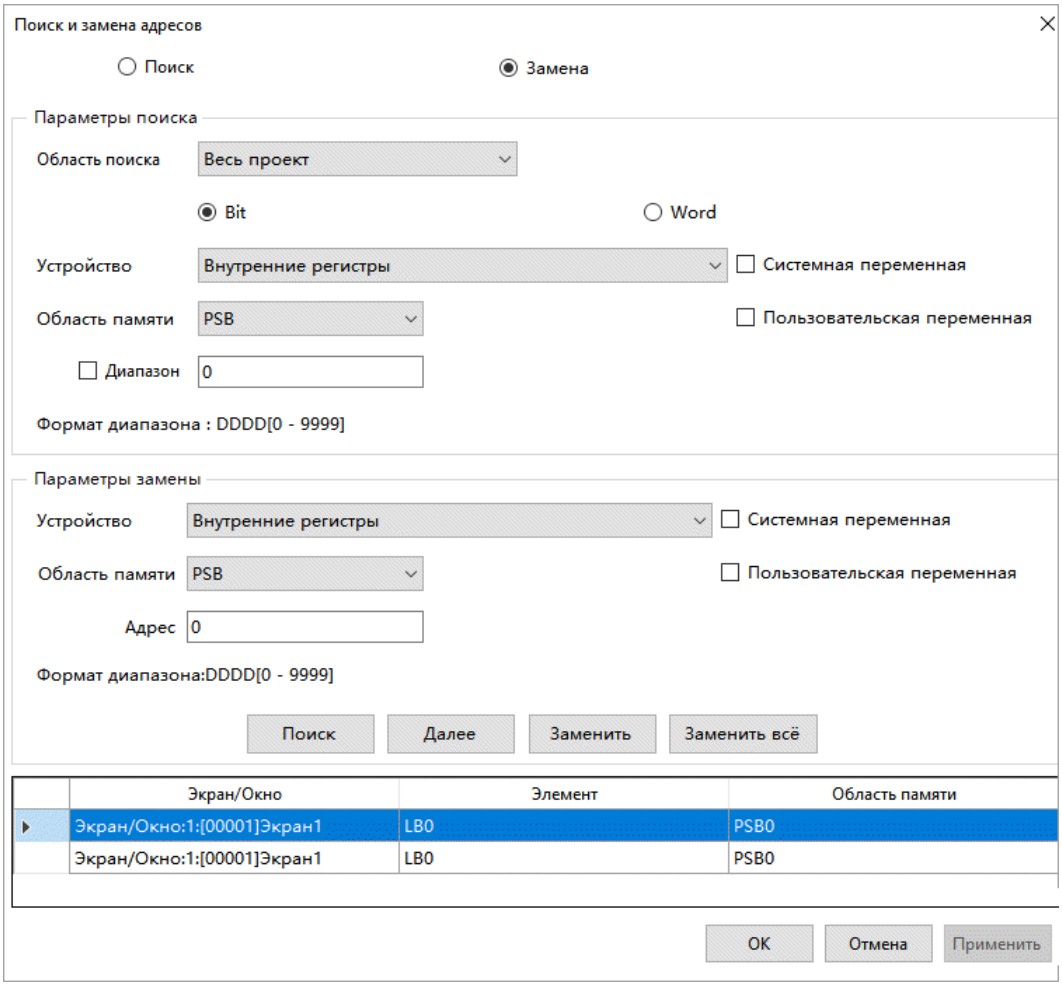


Рисунок 8.31 – Окно поиска и замены адресов

В режиме **Поиск** следует указать:

- **Область поиска** – глобальные элементы (**Весь проект**) или элементы конкретного экрана/окна;
- **Тип значения** – бит или регистр;
- **Устройство** – биты/регистры панели или одного из подключенных slave-устройств;
- **Область памяти**;
- **Адрес или диапазон адресов битов/регистров** .

Также можно осуществить поиск системной или пользовательской переменной с помощью установки соответствующей галочки.

После нажатия на кнопку **Поиск** в нижней части окна будет отображаться список всех найденных вхождений с указанием экрана/окна, идентификатора элемента и адреса привязанного объекта.

Двойное нажатие **ЛКМ** на строке приводит к выделению соответствующего элемента в редакторе визуализации.

Нажатие на кнопку **Поиск** приводит к выбору следующей строки списка.

В режиме **Замена** указываются те же параметры, что и для режима **Поиск**, а также устройство, область памяти и значение адреса, которые будут использоваться в процессе замены. Сначала следует нажать кнопку **Поиск**, а затем использовать кнопку **Заменить** для изменения настроек выбранной строки или **Заменить всё** для выполнения глобальной замены в рамках всего списка.

Если для поиска используется диапазон адресов, то в режиме замены доступна галочка **Смещение адресов**. В случае установки этой галочки к каждому из заменяемых адресов добавляется значение смещения. В случае отсутствия галочки каждый заменяемый адрес будет заменён на указанный адрес.

Таблица 8.12 – Пример использования функции замены с разными вариантами настроек

Настройки поиска	Настройки замены	Результат замены
PSW, 0~2	PFW, 0, галочка **Смещение адресов** не установлена	PSW0->PFW0 PSW1->PFW0 PSW2->PFW0
	PFW, 0, галочка **Смещение адресов** установлена	PSW0->PFW1 PSW1->PFW2 PSW2->PFW3
	PFW, 1, галочка **Смещение адресов** установлена	PSW0->PFW2 PSW1->PFW3 PSW2->PFW3

8.12.2 Поиск и замена шрифта

Команда **Поиск и замена шрифта** позволяет найти элементы, в настройках которых указан выбранный шрифт, и заменить его на другой.

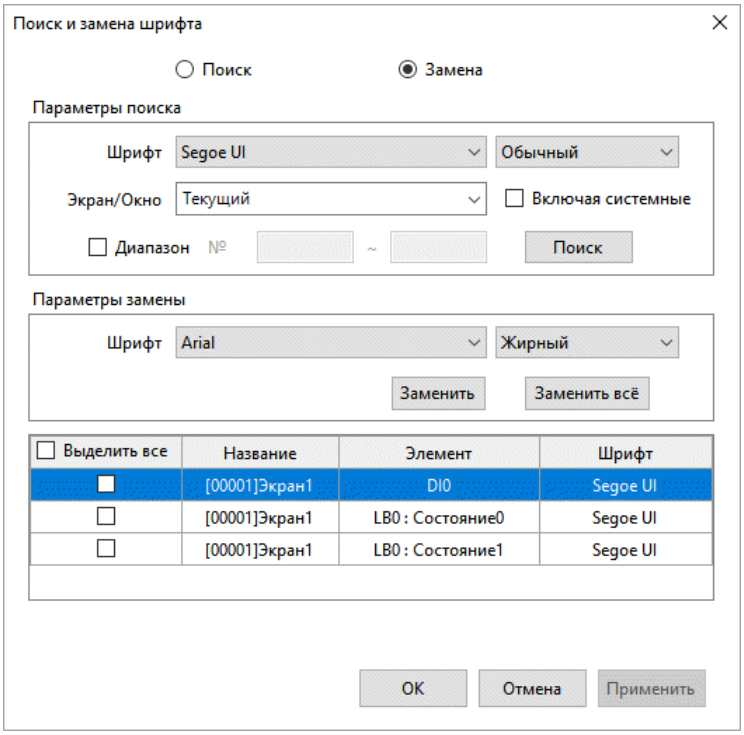


Рисунок 8.32 – Окно поиска и замены шрифта

В режиме **Поиск** следует указать:

- **Шрифт**;
- **Стиль шрифта** – обычный, жирный, курсив или жирный курсив;
- **Область поиска** – текущий экран/окно, выбранный экран/окно, все экраны/окна проекта или заданный диапазон экранов/окон. В случае установки галочки **Включая системные** в окне

выбора будут доступны системные экраны/окна, а в случае выбора варианта **Все** – поиск будет производиться по всем экранам/окнам (в т. ч. и по системным).

После нажатия на кнопку **Поиск** в нижней части окна будет отображаться список всех найденных вхождений с указанием экрана/окна, идентификатора элемента, шрифта и его стиля. Двойное нажатие **ЛКМ** на строке приводит к выделению соответствующего элемента в редакторе визуализации.

В режиме **Замена** указываются те же параметры, что и для режима **Поиск**, а также шрифт и стиль, который будет использоваться для замены. Сначала следует нажать кнопку **Поиск**, а затем использовать кнопку **Заменить** для изменения настроек выделенных галочками элементов или **Заменить всё** для выполнения замены в рамках всего списка.

8.13 Настройки проекта

Меню настроек проекта содержит 8 вкладок:

- **Общие** – содержит общие настройки панели;
- **Панель** – позволяет изменить модификацию панели, выбранную в проекте;
- **Управление** – позволяет указать регистры/переменные, которые будут использоваться для чтения номера текущего экрана панели и его переключения;
- **Права доступа** – позволяет создавать пользователей;
- **Время** – позволяет настроить способ получения системного времени панели;
- **Устройства** – позволяет настроить коммуникационные интерфейсы панели;
- **Сканер штрихкодов** – позволяет задать настройки интерфейса при подключении сканера штрихкодов;
- **Проект** – позволяет указать автора проекта и комментарии.

8.13.1 Вкладка «Общие»

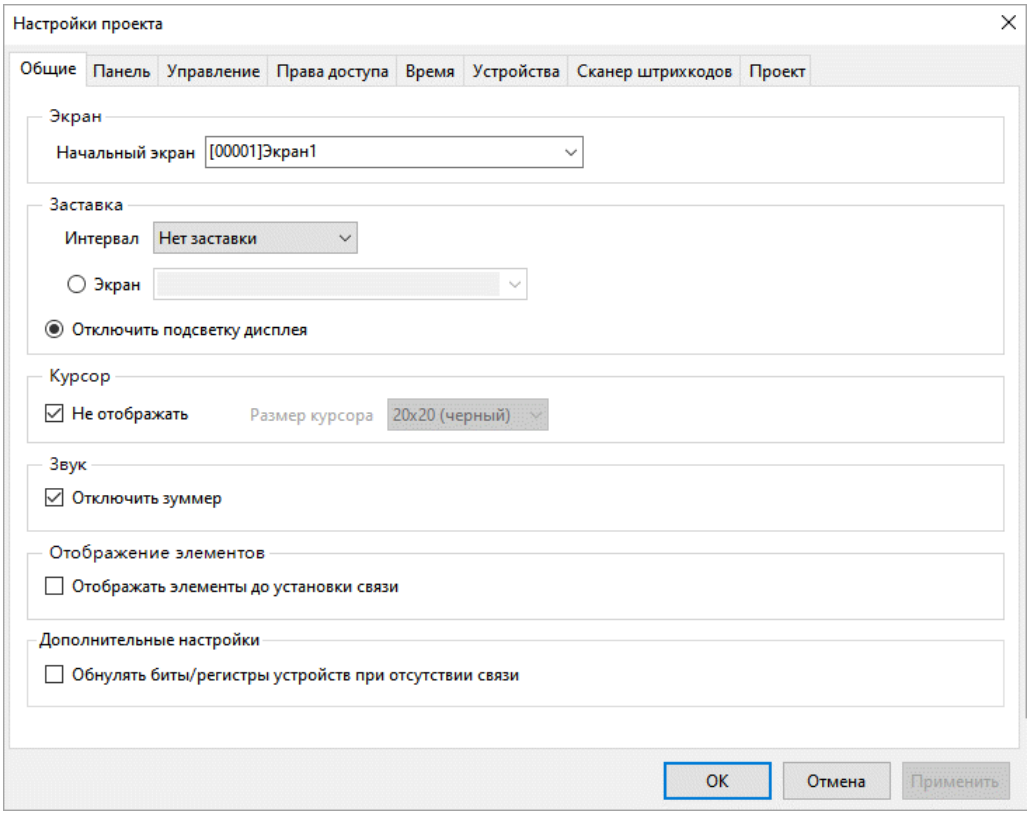


Рисунок 8.33 – Настройки проекта, вкладка «Общие»

Таблица 8.13 – Настройки вкладки «Общие»

Параметр	Описание	Системный бит/регистр
Начальный экран	Начальный экран, который будет открыт после запуска проекта	SPFW257
Заставка	Экран заставки, который будет открыт в случае заданного периода неактивности пользователя (задаётся в минутах). В случае установки галочки **Отключить подсветку дисплея** вместо перехода на экран заставки происходит отключение подсветки. В случае нажатия на экран заставки (или на экран с отключённой подсветкой) происходит переход на экран, который отображался до перехода на экран заставки или отключения подсветки	Период неактивности для заставки: SPFW258 Отключить подсветку: SPFW452 Флаг заставки: SPSB4 Флаг отключения подсветки: SPSB5
Курсор	Размеры и цвет пиктограммы курсора, отображаемого в месте нажатия на дисплей. В случае установки галочки **Не отображать** курсор отображаться не будет	SPFW449 (1 – отключить, 0 – включить)
Звук	В случае установки галочки происходит отключение зуммера панели. В этом случае: • отсутствует звук нажатий на элементы панели; • не работает элемент Зуммер; • не работает функция Веер.	SPFW448 (1 – отключить, 0 – включить)
Отображение элементов	В случае установки галочки элементы, к которым привязаны биты/регистры slave-устройства, отображаются сразу после запуска проекта до получения первого ответа от устройства (отображаются значения **FALSE/0**). В случае снятия галочки данные элементы будут отображены только после получения первого ответа от slave-устройства (отображаются считанные значения)	
Дополнительные настройки	В случае установки галочки **Обнулять биты/ регистры устройств при отсутствии связи** – при отсутствии связи со slave-устройством в элементах, к которым привязаны его биты/ регистры, отображаются значения **FALSE/0**. В случае снятия галочки отображаются последние считанные значения	

8.13.2 Вкладка «Панель»

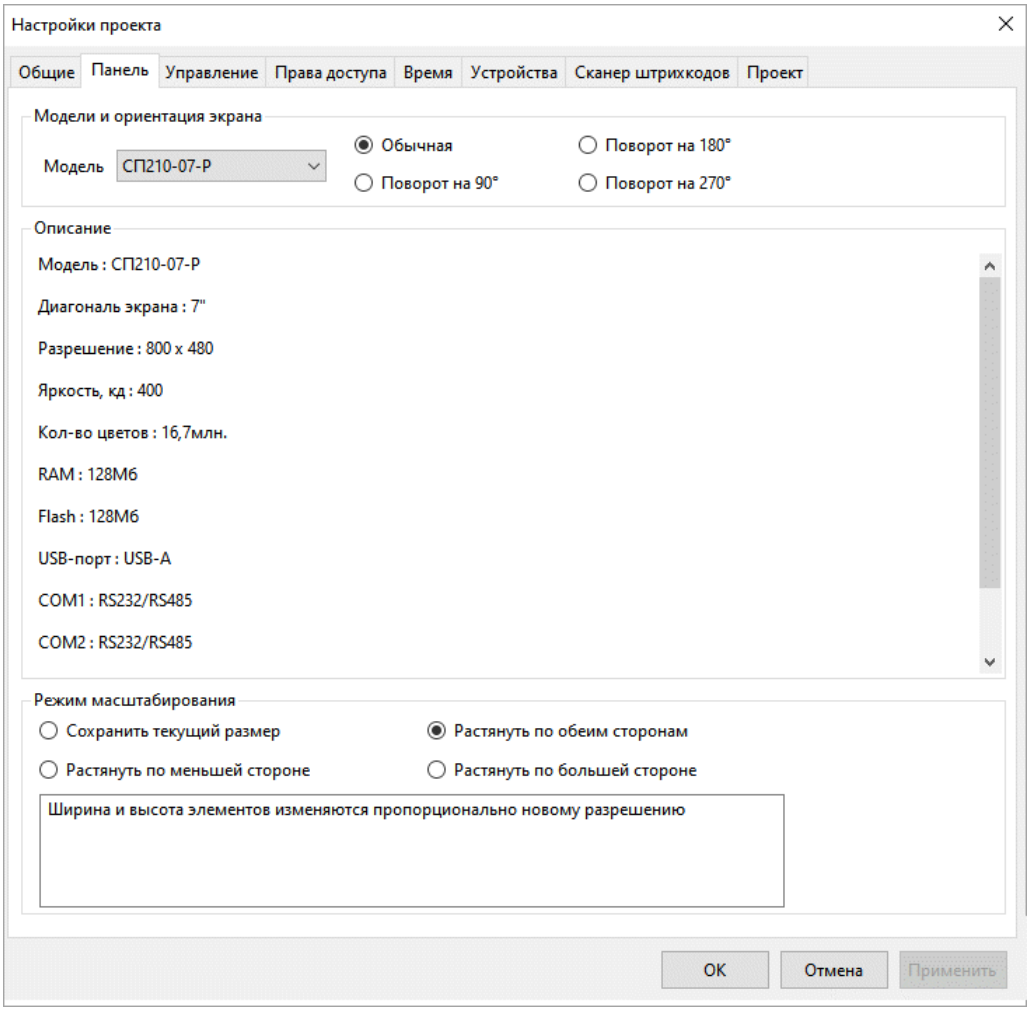


Рисунок 8.34 – Настройки проекта, вкладка «Панель»

Таблица 8.14 – Настройки вкладки «Панель»

Параметр	Описание
Модель	Выбор модификации панели
Описание	Технические характеристики выбранной модификации
Режим масштабирования	Тип масштабирования элементов при выборе модификации, разрешение которой отличается от текущей модификации. Возможные варианты изменения ширины и высоты элементов: • пропорционально новому разрешению; • пропорционально разрешению большей стороны дисплея; • пропорционально разрешению меньшей стороны дисплея; • не изменяются.

8.13.3 Вкладка «Управление»

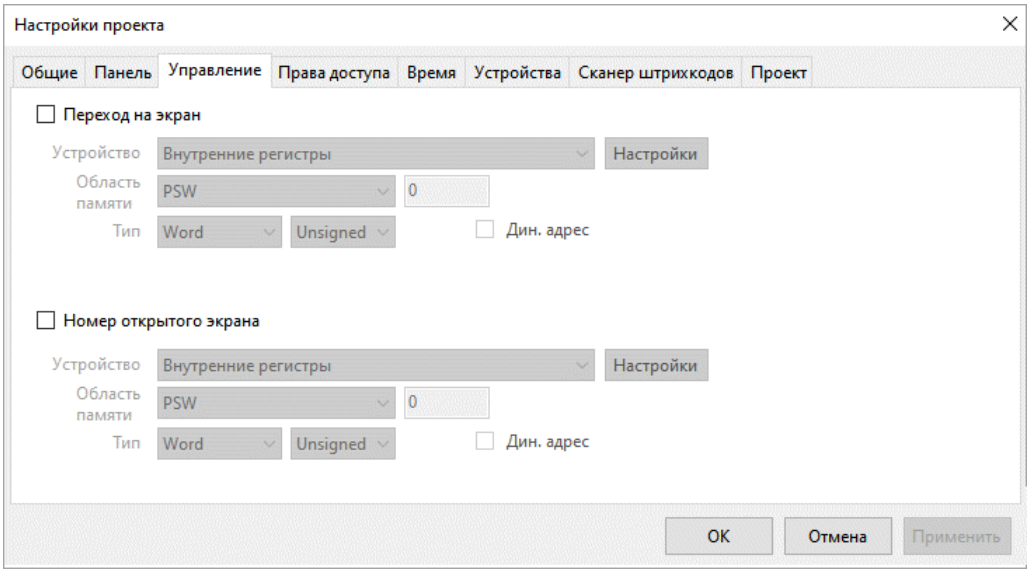


Рисунок 8.35 – Настройки проекта, вкладка Управление

Таблица 8.15 – Настройки вкладки Управление

Параметр	Описание
Переход на экран	Выбор адреса регистра или переменной, которая будет использоваться для переключения на экран с заданным номером. Номера экранов отображаются в дереве проекта. После переключения в параметр автоматически записывается значение 0
Номер открытого экрана	Выбор адреса регистра или переменной, в которую будет записываться номер открытого в данный момент экрана

8.13.4 Вкладка «Права доступа»

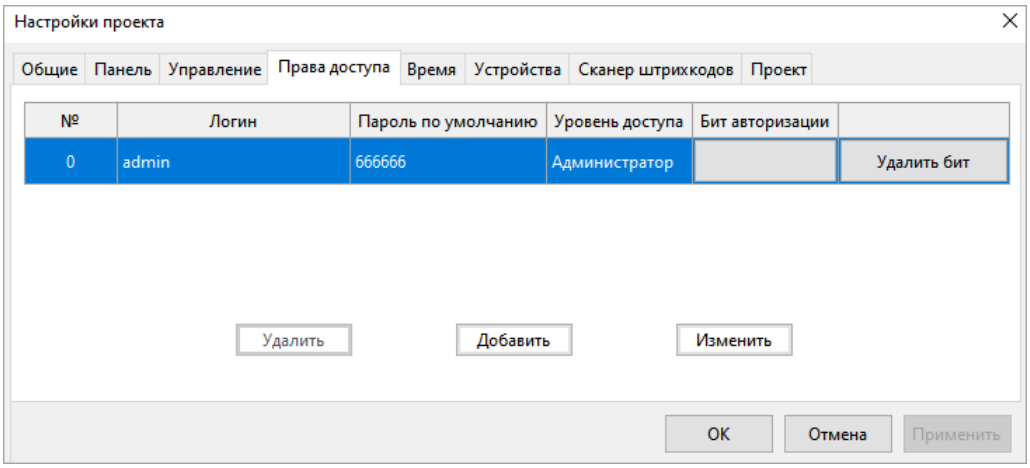


Рисунок 8.36 – Настройки проекта, вкладка Права доступа

На данной вкладке производится создание пользователей проекта. Для каждого пользователя нужно указать:

- логин (не более 16 символов);
- пароль (не более 8 символов, символы кириллицы не поддерживаются);
- уровни доступа.

Уровни доступа позволяют ограничить:

- возможность перехода на экран (см. параметр Уровень доступа);
- возможность открытия окна;
- видимость и активность графических элементов (см. настройки вкладки **Безопасность**).

Реализовано 30 независимых уровней доступа. Пользователю одновременно может быть назначено несколько. Уровни доступа не являются иерархическими: пользователю с уровнем 2 недоступны экраны, окна и элементы, для которых требуется уровень 1.

По умолчанию в проекте предусмотрен пользователь-администратор с логином **admin** и паролем по умолчанию **666666**, которому предоставлены все уровни доступа. Администратора нельзя удалить. Дополнительно можно создать до 19 пользователей.

Для каждого пользователя может быть назначен бит авторизации. Этот бит принимает значение **TRUE** после авторизации данного пользователя. Если записать в этот бит значение **FALSE** – то будет выполнен выход из системы (логаут) пользователя. Для удаления этого бита используется кнопка **Удалить бит**.

Системный бит **SPSB0**:

- принимает значение **TRUE** после авторизации любого из пользователей;
- принимает значение **FALSE** после выхода пользователя из системы.

Системный бит **SPSB1**:

- принимает значение **TRUE** после успешного выхода из системы любого пользователя;
- принимает значение **FALSE** после успешной авторизации любого пользователя.

Для авторизации пользователя используется системное окно **25001**, для изменения пароля – **25014**. Пароли пользователей (за исключением администратора) сохраняются в системных регистрах (см. таблица 13.3) и могут быть изменены средствами проекта (например, с помощью макроса).

При переходе на экран заставки или отключении подсветки выход пользователя из системы не выполняется. При нажатии на экран заставки выполняется возврат на ранее открытый экран, даже если текущий уровень доступа оператора формально недостаточен для его отображения.

8.13.5 Вкладка «Время»

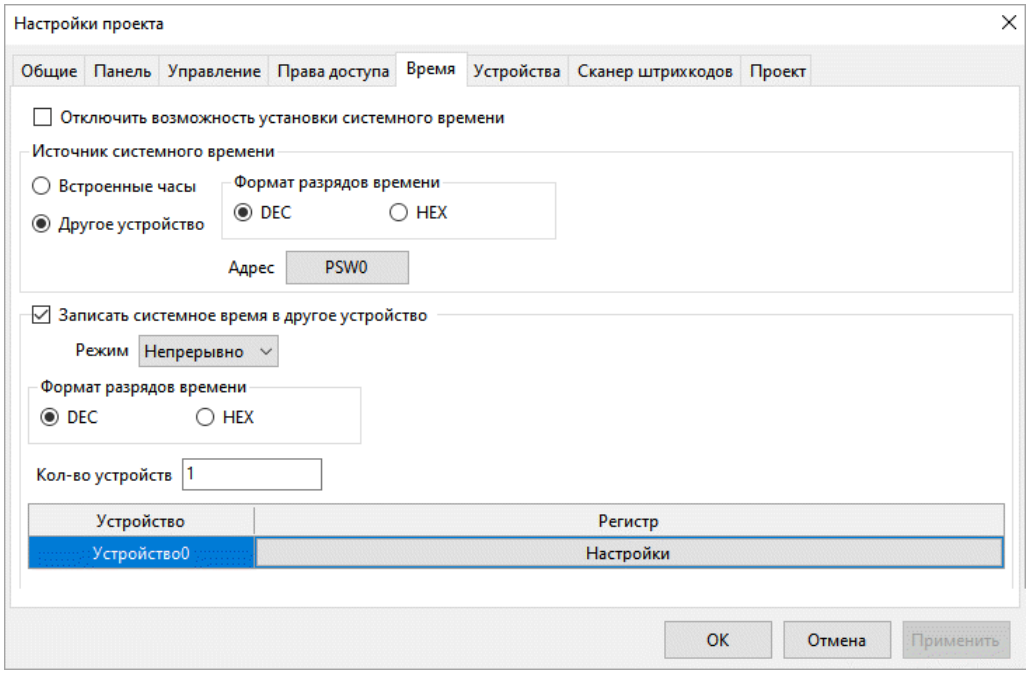


Рисунок 8.37 – Настройки проекта, вкладка «Время»

Таблица 8.16 – Настройки вкладки Время

Параметр	Описание
Отключить возможность установки системного времени	В случае установки галочки у пользователя не будет возможности изменить системное время панели ни одним из доступных способов. Галочка автоматически устанавливается при активации функции Контроль оплаты
Источник системного времени	Выбор источника системного времени. Возможные варианты: внутренние часы панели или регистр (в т. ч. можно указать регистр slave-устройства), начиная с которого размещаются значения разрядов системного времени в порядке: год, месяц, день, часы, минуты, секунды. Пользователь указывает формат разрядов системного времени – DEC или HEX
Записать системное время в другое устройство	В случае установки галочки панель записывает значения разрядов системного времени во внутренние регистры или регистры одного или нескольких подключённых slave-устройств в порядке: год, месяц, день, часы, минуты, секунды. Пользователь указывает формат разрядов системного времени (DEC или HEX) и режим записи: • непрерывно (с минимально возможным интервалом); • однократно по переднему или заднему фронту указанного бита; • циклически (интервал задаётся вручную или через выбранный регистр в рамках указанной размерности).

ПРИМЕЧАНИЕ
Чтение и установка времени может быть произведена с помощью системных регистров (см. пункт 13.5).

8.14 Данные PFW-регистров

С помощью команды **Файл – Данные PFW-регистров** пользователь может установить начальные значения, которые будут однократно записаны в PFW-регистры панели после первого запуска проекта.

В данном окне пользователь добавляет группы регистров, указывая для каждой начальный и конечный адрес. Диапазоны адресов разных групп не могут пересекаться. Для изменения диапазона следует выделить группу, ввести новые значения начального и конечного адреса, после чего нажать кнопку **Применить**.

Для ввода значений регистров следует два раза нажать **ЛКМ** на группу или выделить группу и нажать кнопку **Изменить**. Откроется окно, в котором можно ввести значения для каждого из регистров группы в выбранном формате (DEX или HEX), нажав на ячейку регистра. С помощью кнопок **Установить в 0** и **Установить в 0xFFFF** можно записать во все регистры соответствующее значение.

Для ввода значений типа Float потребуется представить их в виде двух HEX-значений. Для этого удобно использовать онлайн-конвертер: https://www.h-schmidt.net/FloatConverter/IEEE754.html

Для ввода строковых значений потребуется представить их в виде набора байт: https://crypt-online.ru/crypts/text2hex/

В примерах ниже показана инициализация PFW-регистров с размещением значений типа **Float** и **String** (кодировка **Win1251**). Обратите внимание на отличие в порядке байт между вводимым и отображаемым в конвертере значениями.

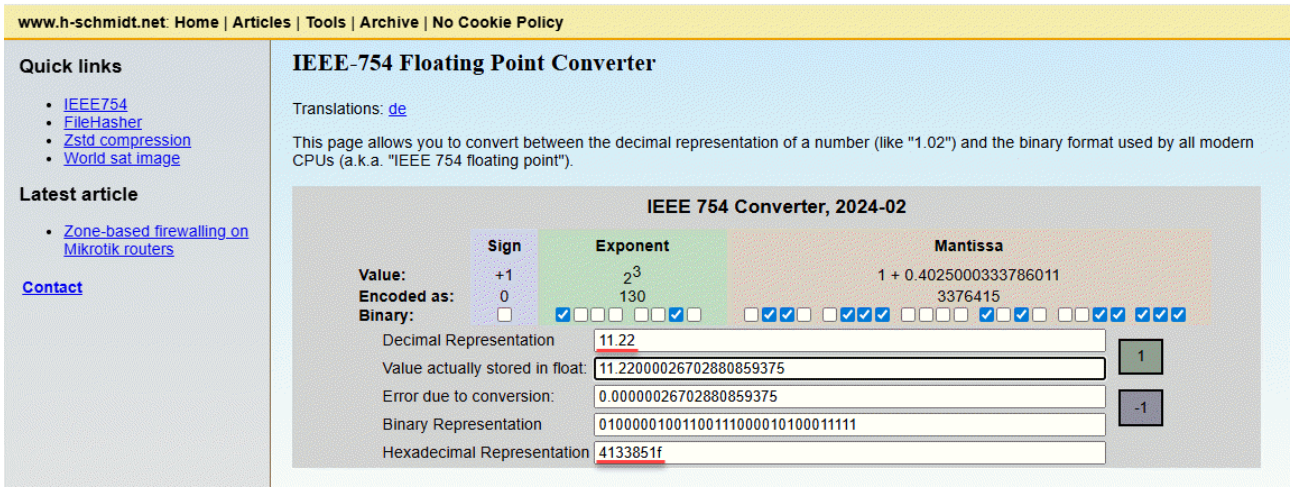


Рисунок 8.38 – Отображение Float в HEX-представлении

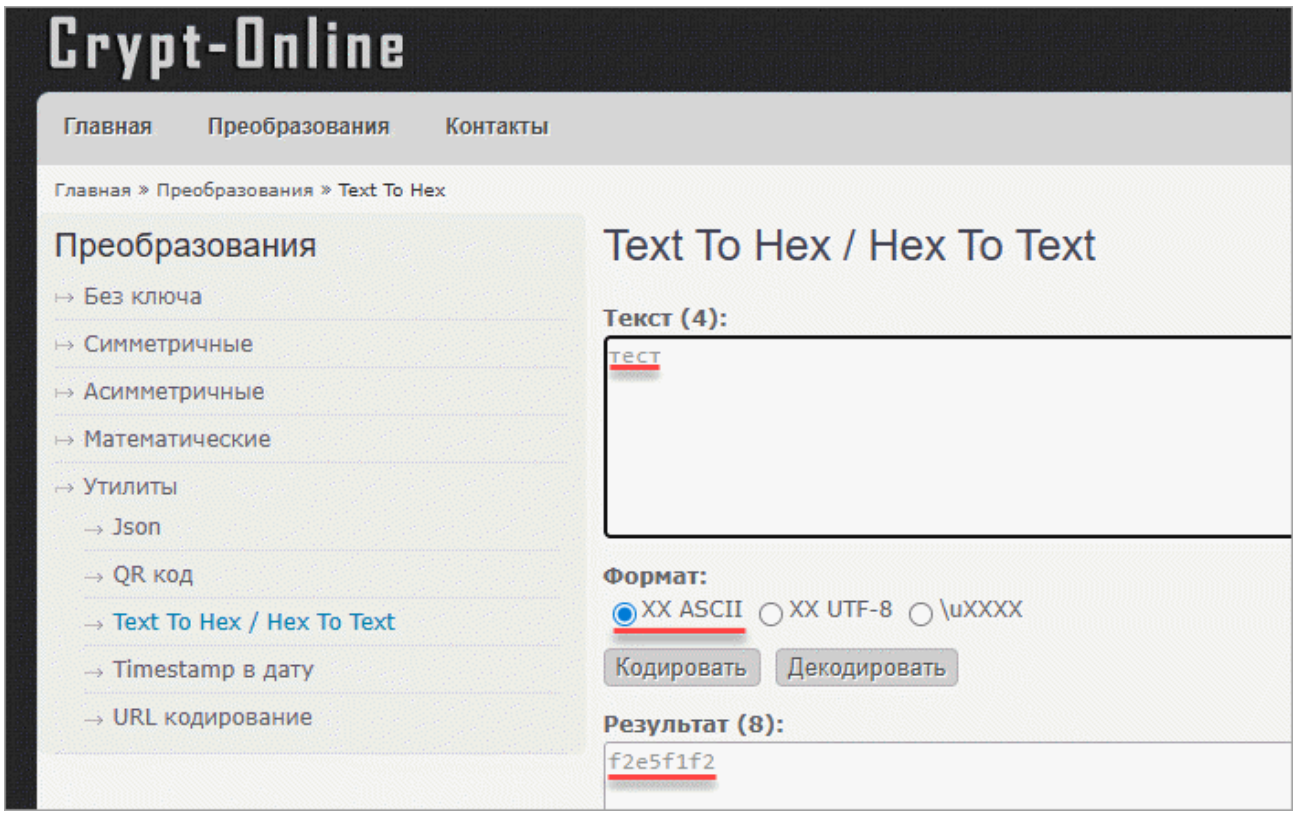


Рисунок 8.39 – Отображение String (Win1251) в HEX-представлении

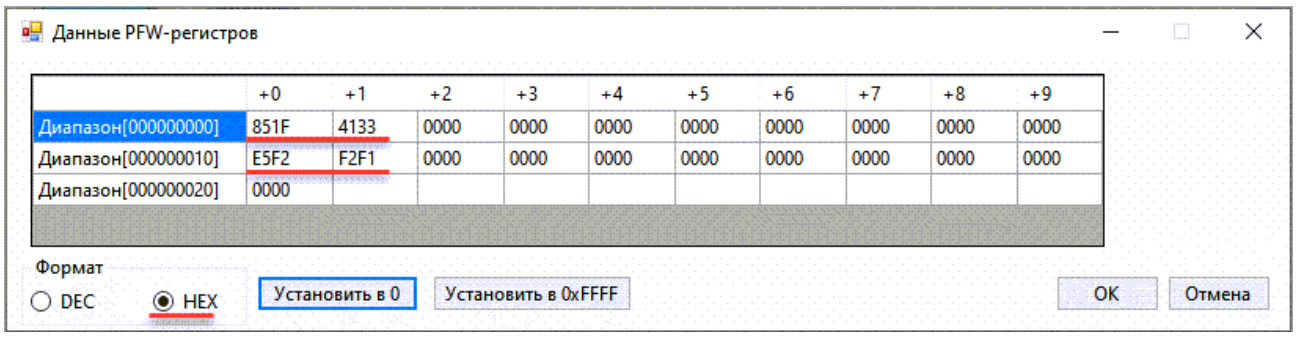
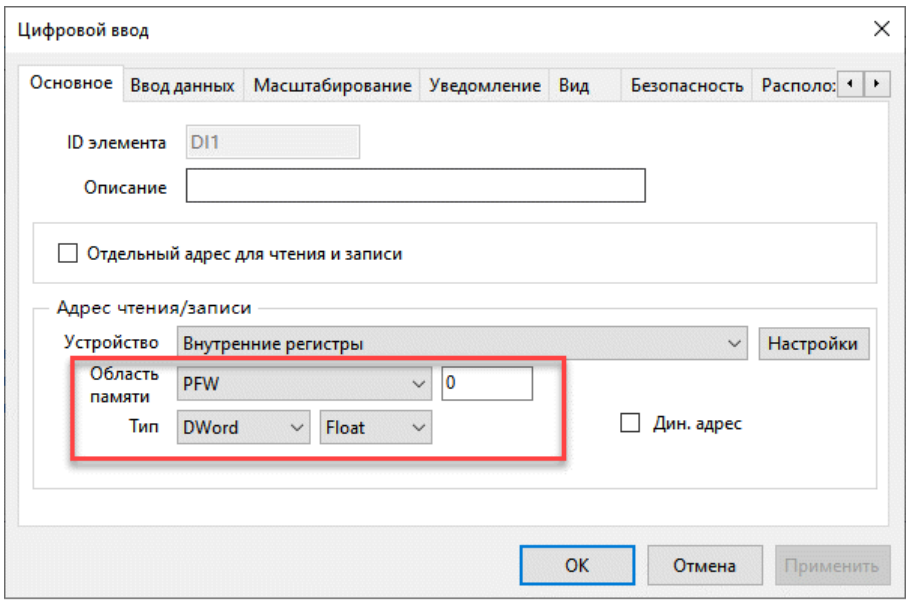


Рисунок 8.40 – Инициализация PFW для типов Float и String



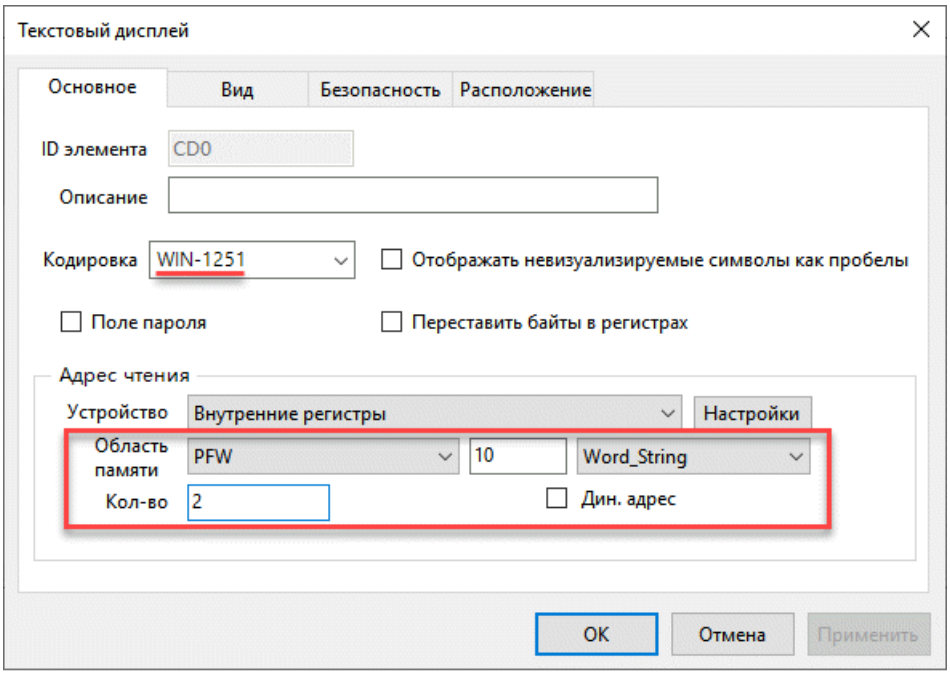


Рисунок 8.41 – Настройки элементов визуализации



Рисунок 8.42 – Отображение размещенных в области PFW значений в элементах визуализации

9 Загрузка проекта и режим эмуляции

9.1 Загрузка проекта с ПК

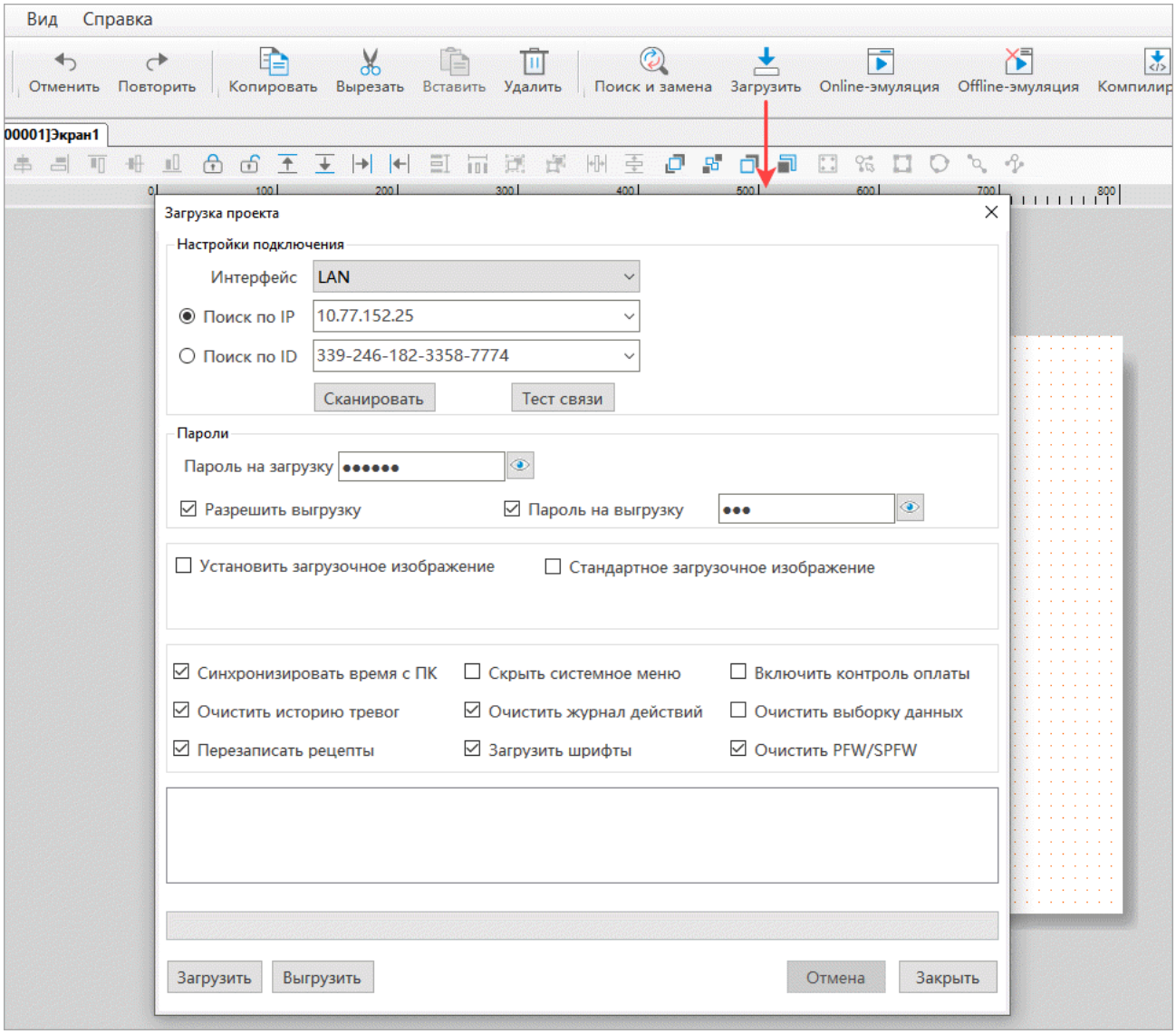


Рисунок 9.1 – Окно загрузки проекта

Для панелей базовых модификаций (-Б) загрузка проекта выполняется через интерфейс USB B. Перед началом загрузки нужно подключить панель к ПК с помощью кабеля USB B – USB A.

Для панелей расширенных модификаций (-Р) загрузка проекта выполняется через интерфейс Ethernet. Перед началом загрузки:

- панель должна быть подключена к той же локальной сети, что и ПК (или напрямую к сетевому адаптеру ПК);
- для панели должны быть заданы сетевые настройки, соответствующие локальной сети (с помощью системного меню или экранного конфигуратора).

Для загрузки проекта следует использовать команду **Файл – Загрузить проект** или соответствующий ярлык на панели ярлыков. Доступны следующие кнопки:

- **Загрузить** – выполняет загрузку проекта.
- **Выгрузить** – выполняет попытку выгрузки.
- **Закрыть** – закрывает окно загрузки.

ПРИМЕЧАНИЕ
Для панелей базовой модификации (-Б) загрузка проекта возможна только в том случае, если к порту USB A не подключен USB-накопитель.

Таблица 9.1 – Настройки меню Загрузка проекта

Параметр	Описание
Тип соединения	USB или Ethernet
Поиск по IP	IP-адрес панели, в которую будет загружен проект
Поиск по ID	Идентификатор панели (отображается в системном меню или экранном конфигураторе), в которую будет загружен проект
Пароль на загрузку	Пароль, который требуется для загрузки проекта. Задаётся в системном меню или экранном конфигураторе. Значение по умолчанию – **123456**
Сканировать	Данная кнопка доступна только при подключении по Ethernet. При нажатии кнопки ПК отправляет UDP-пакет на **широковещательный адрес сети** (порт **3333**), в результате чего его получают все панели. Результаты сканирования отображаются в виде таблицы с указанием имени панели, IP-адреса, идентификатора и модификации
Тест связи	При нажатии на кнопку происходит проверка связи с панелью (отправка ping-запроса), после чего отображается соответствующее информационное сообщение
Разрешить выгрузку	В случае установки галочки загружаемый проект будет доступен для выгрузки
Пароль на выгрузку	Пароль, который требуется для выгрузки проекта. Может быть изменен в системном меню или экранном конфигураторе
Установить загрузочное изображение	В случае установки галочки можно выбрать изображение формата **.jpg** или **.bmp**, которое будет отображаться на дисплее панели в процессе её загрузки
Стандартное загрузочное изображение	В случае установки галочки в процессе загрузки панели на её дисплее будет отображаться стандартное изображение с логотипом ОВЕН
Синхронизировать время с ПК	В случае установки галочки при загрузке проекта в часы панели будет установлено системное время ПК
Очистить историю тревог	В случае установки галочки при загрузке проекта будут удалены файлы истории тревог
Перезаписать рецепты	В случае установки галочки при загрузке проекта значения в файлах рецептов будут заменены на значения из загружаемого проекта
Скрыть системное меню	В случае установки галочки после загрузки проекта у пользователя не будет возможности перейти в системное меню (кнопка **Настройки** не будет отображаться)
Очистить журнал действий	В случае установки галочки при загрузке проекта будет очищен журнал действий оператора
Загрузить шрифты	В случае установки галочки при загрузке проекта в память панели будут загружены файлы шрифтов, использованных в настройках графических элементах, но отсутствующих в панели
Включить контроль оплаты	В случае установки галочки при загрузке проекта будет включена функция контроля оплаты
Очистить выборку данных	В случае установки галочки при загрузке проекта будут удалены файлы выборки данных
Очистить PFW/SPFW	В случае установки галочки при загрузке проекта значения всех регистров **PFW** будут обнулены и установлены значения, заданные в меню Данные PFW-регистров, а значения регистров **SPFW** – установлены в значения по умолчанию

9.2 Загрузка проекта с USB-накопителя

Для загрузки проекта с USB-накопителя следует:

- выполнить команду **Файл – Создать загрузочный проект**. Установить нужные галочки и указать путь к создаваемому файлу. Название файла не должно содержать символов кириллицы, пробелов и спецсимволов;

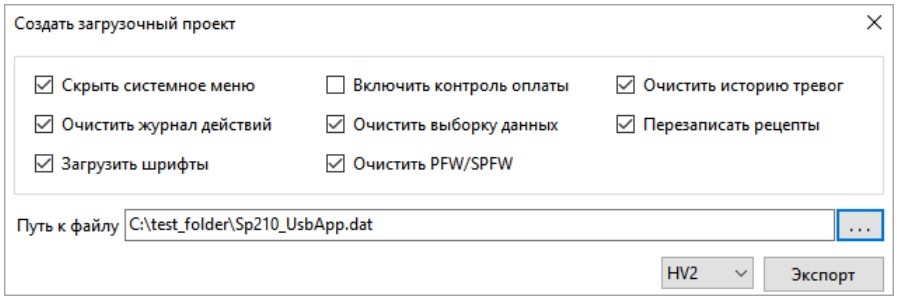


Рисунок 9.2 – Окно создания загрузочного проекта

- полученный файл с расширением .dat скопировать на USB-накопитель. Требования к USB-накопителю: файловая система FAT32, объём – не более 32 Гб;
- подключить USB-накопитель к включенной панели;
- в открывшемся окне нажать на кнопку **Обновление проекта** и указать путь к файлу **.dat**;
- после загрузки проект будет автоматически запущен.

9.3 Эмуляция

Режим эмуляции позволяет проверить работу проекта без реальной панели. Существует два варианта эмуляции: offline-эмуляция и online-эмуляция.

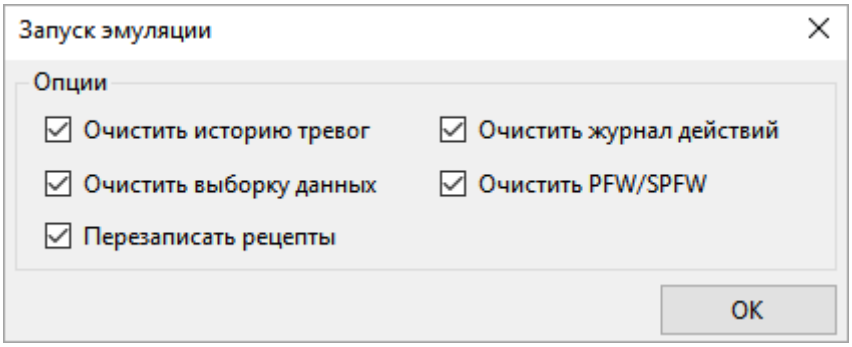


Рисунок 9.3 – Окно запуска эмуляции

Таблица 9.2 – Сравнение возможностей offline-эмуляции и online-эмуляции

Параметр	Offline-эмуляция	Online-эмуляция
Проверка обмена с другими устройствами	Не поддерживается	Поддерживается, в режимах Master и Slave
Время непрерывной работы	Без ограничений	2 часа (после этого эмулятор можно перезапустить)

В режиме эмуляции не поддерживается:

- работа со сканерами штрихкодов;
- управление зуммером;
- управление яркостью подсветки;
- воспроизведение аудиофайлов;
- элемент **Воспроизведение видеофайлов**;

- элемент **Отображение видеопотока (RTSP)**;
- функционал системного меню или экранного конфигуратора;
- функция **Контроль оплаты**;
- функции меню **MQTT** и **VNC**.

Рабочая директория эмулятора в директории установки конфигуратора: **\Temp**

Директория, соответствующая USB-накопителю: **\Temp\Run\Storage\udisk**

Для запуска эмуляции используются команды меню **Файл** или соответствующие ярлыки панели ярлыков.

При запуске online-эмуляции последовательно появляются два диалоговых окна:

- В первом производится маппинг COM-портов. Пользователь должен указать, какие COM-порты ПК будут соответствовать COM-портам, использованным в проекте панели (COM1 и COM2).

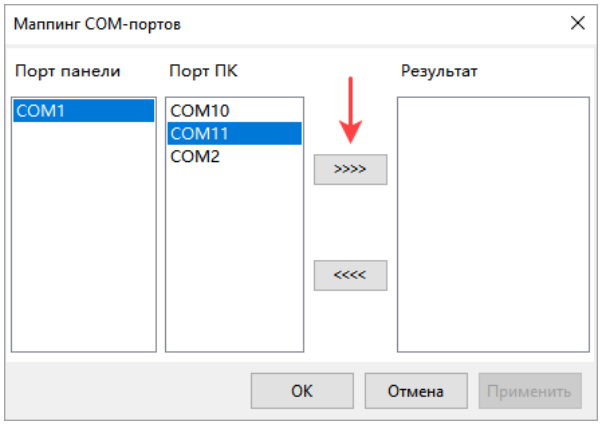


Рисунок 9.4 – Маппинг COM-портов

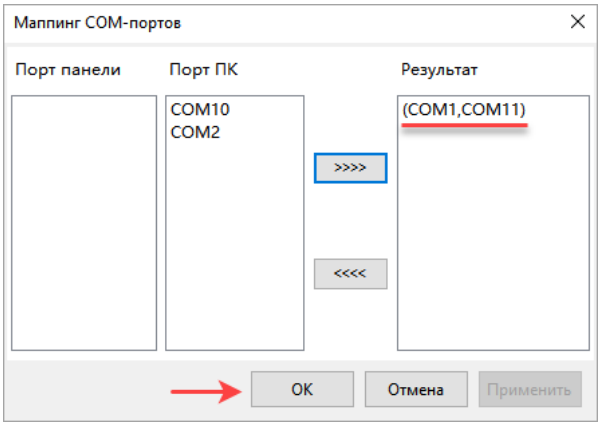


Рисунок 9.5 – Маппинг COM-портов

- Во втором окне доступны галочки, описанные в таблице 9.1:
  - очистить историю тревог;
  - очистить выборку данных;
  - перезаписать рецепты;
  - очистить журнал действий;
  - очистить PFW/SPFW.

9.4 Сохранение эмуляции

С помощью команды **Файл – Сохранить эмуляцию** окружение online-эмуляции сохраняется в виде набора файлов, которые можно перенести на другой ПК. Это позволяет запустить эмуляцию на компьютере, на котором не установлен конфигуратор (например, для демонстрации функционала заказчику).

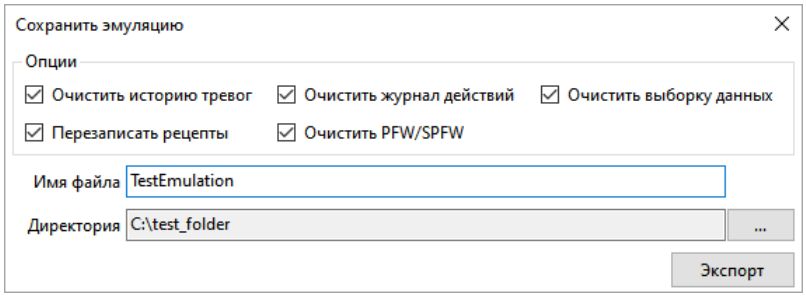


Рисунок 9.6 – Окно сохранения эмуляции

В открывшемся окне следует указать директорию ПК, в которой будут сохранены файлы конфигурации, и название исполняемого файла. Описание доступных галочек приведено в таблице 9.1.

Для запуска эмуляции на другом ПК следует перенести на него содержимое указанной выше директории и запустить файл с расширением **.exe**. При запуске появится окно маппинга COM-портов (см. рисунок 9.4, рисунок 9.5).

Время работы online-эмуляции ограничено 2 часами. После этого эмулятор можно перезапустить.

10 Графические элементы

10.1 Типовые настройки

В данном пункте описаны типовые настройки, присутствующие у большинства графических элементов. Уникальные настройки элементов описаны в следующих пунктах.

10.1.1 Вкладка «Основное»

Два параметра вкладки **Основное** являются типовыми для всех элементов:

- **ID элемента** – уникальный идентификатор элемента. Не доступен для изменения. Отображается в списке элементов, карте адресов и журнале действий оператора;
- **Описание** – описание элемента. Отображается в журнале действий оператора;.

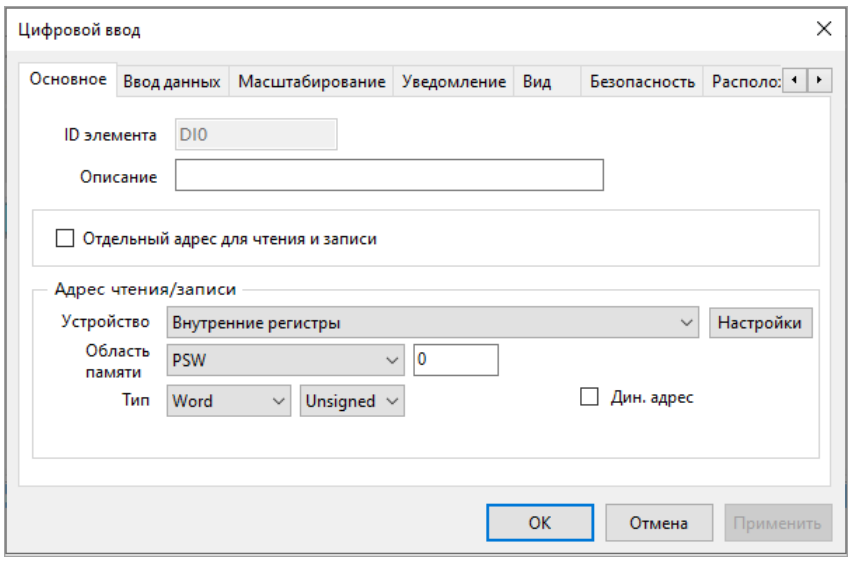


Рисунок 10.1 – Вкладка Основное

Кроме того, для многих элементов выбирается адрес основного отображаемого или изменяемого параметра элемента.

10.1.2 Вкладка «Безопасность»

На вкладке **Безопасность** доступны настройки, связанные с ограничением видимости/активности элемента для отдельных групп пользователей. У некоторых элементов присутствуют не все параметры, приведенные в таблице (например, у элементов, не поддерживающих возможность нажатия, нет настроек управления активностью).

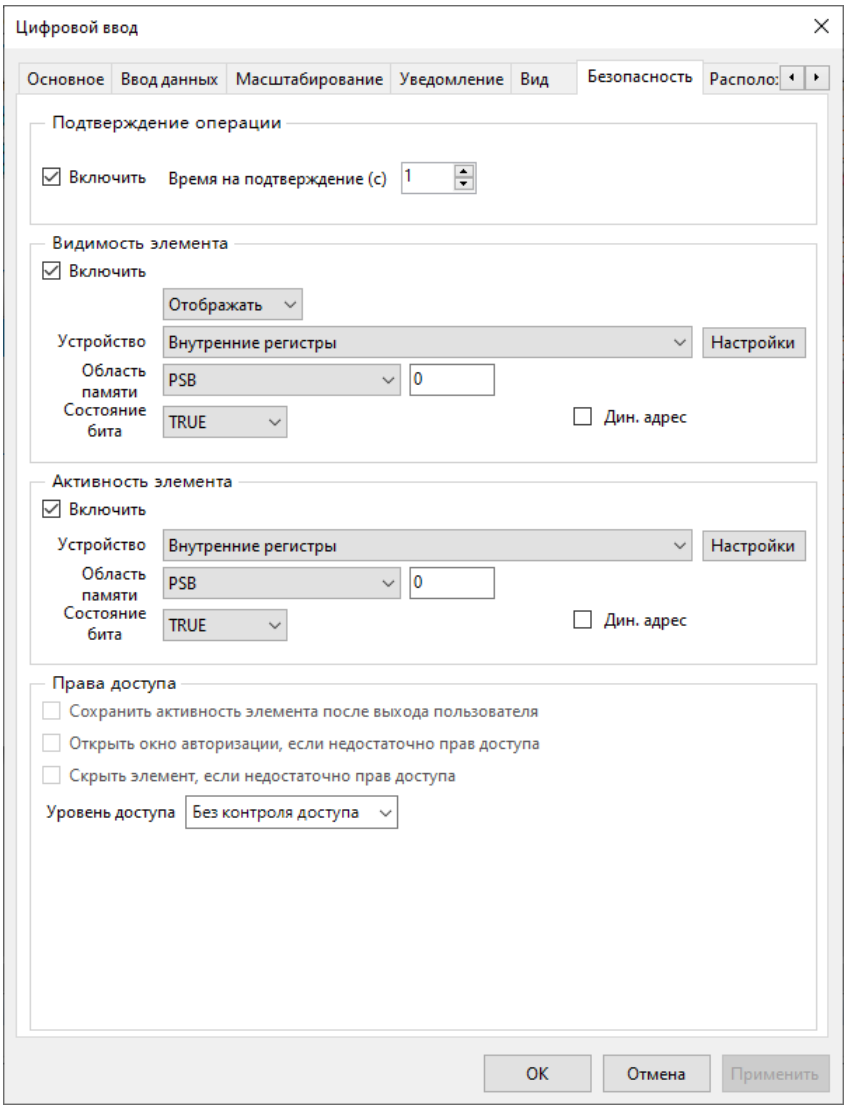


Рисунок 10.2 – Вкладка Безопасность

Таблица 10.1 – Настройки вкладки Безопасность

Настройка	Описание
Подтверждение операции	В случае установки галочки после нажатия на элемент на заданное время открывается системное окно **Подтверждение выполнения** (25006). Доступные кнопки: **ОК** — операция будет выполнена (например, произойдёт ввод нового значения). **Отмена** (или если окно закроется по истечению времени) — операция не будет выполнена
Видимость элемента	В случае установки галочки пользователь указывает бит, который определяет видимость элемента, и значение бита, соответствующее видимому состоянию (**TRUE** или **FALSE**)

Продолжение таблицы 10.1

Настройка	Описание
Активность элемента	В случае установки галочки пользователь указывает бит, который определяет активность элемента, и значение бита, соответствующее активному состоянию (**TRUE** или **FALSE**). Значения неактивных элементов выделены серым цветом
Уровень доступа	Уровень доступа пользователя, который требуется для взаимодействия с данным элементом
Сохранить активность элемента после выхода пользователя	В случае установки галочки – если после авторизации пользователь с достаточным уровнем доступа выйдет из системы, то элемент останется видимым и активным
Открыть окно авторизации, если не хватает уровня доступа	В случае установки галочки при нажатии на данный элемент пользователем, не имеющим требуемого уровня доступа, будет открыто системное окно **Авторизация** (25001)
Скрыть элемент, если недостаточно прав доступа	В случае установки галочки данный элемент не будет отображаться для пользователей, не имеющих требуемого уровня доступа

10.1.3 Вкладка «Расположение»

На вкладке **Расположение** доступны настройки, связанные с координатами и размерами элемента.

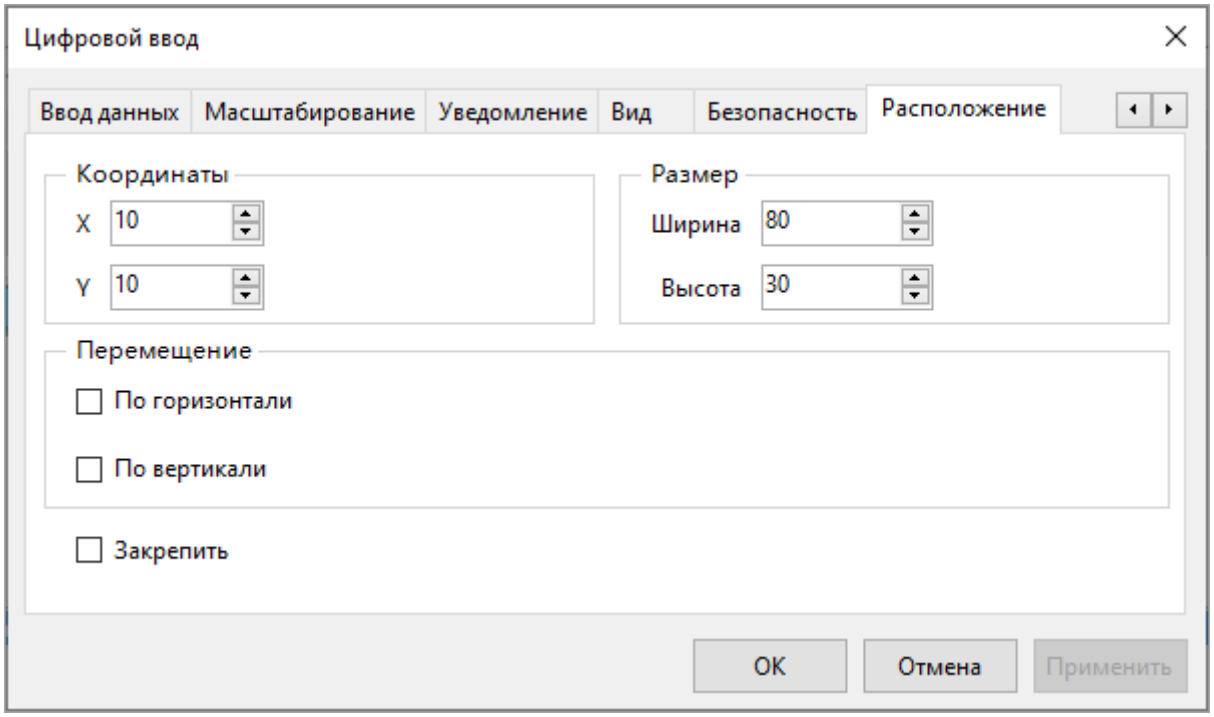


Рисунок 10.3 – Вкладка Расположение

Таблица 10.2 – Настройки вкладки Расположение

Настройка	Описание
Координаты (X, Y)	Координаты размещения элемента на экране визуализации. Соответствуют верхней левой точке элемента относительно верхней левой точки экрана визуализации
Размер	Размеры элемента в пикселях. Ограничения: • X + ширина ≤ ширина экрана панели · 1.1875; • Y + высота ≤ высота экрана панели · 1.1875; Т. е. чтобы создать элемент с размерами, совпадающими с разрешением экрана панели, следует сначала для X и Y установить значение 0. Указанные ограничения не влияют на изменение размеров элемента с помощью опорных точек, отображаемых при выделении элемента в редакторе визуализации
Перемещение	В случае установки галочки можно указать адрес регистра целочисленного знакового значения/переменной, который будет использоваться для относительного перемещения элемента по оси X/Y (значение 100 соответствует перемещению на 100 пикселей вправо/вниз, а значение -100 – перемещению на 100 пикселей влево/вверх)
Закрепить	В случае установки галочки элемент нельзя будет перемещать по экрану (аналогично одноименному ярлыку редактора визуализации и соответствующей команде контекстного меню элемента)

10.2 Графика

10.2.1 Линия

Элемент **Линия** используется для отображения на экране панели прямой линии.

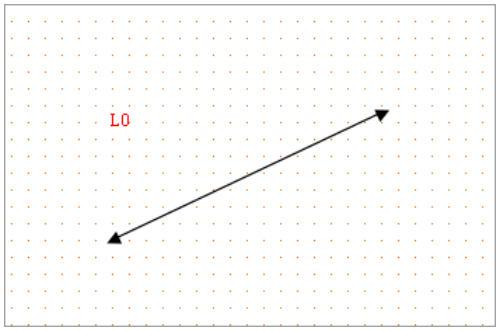
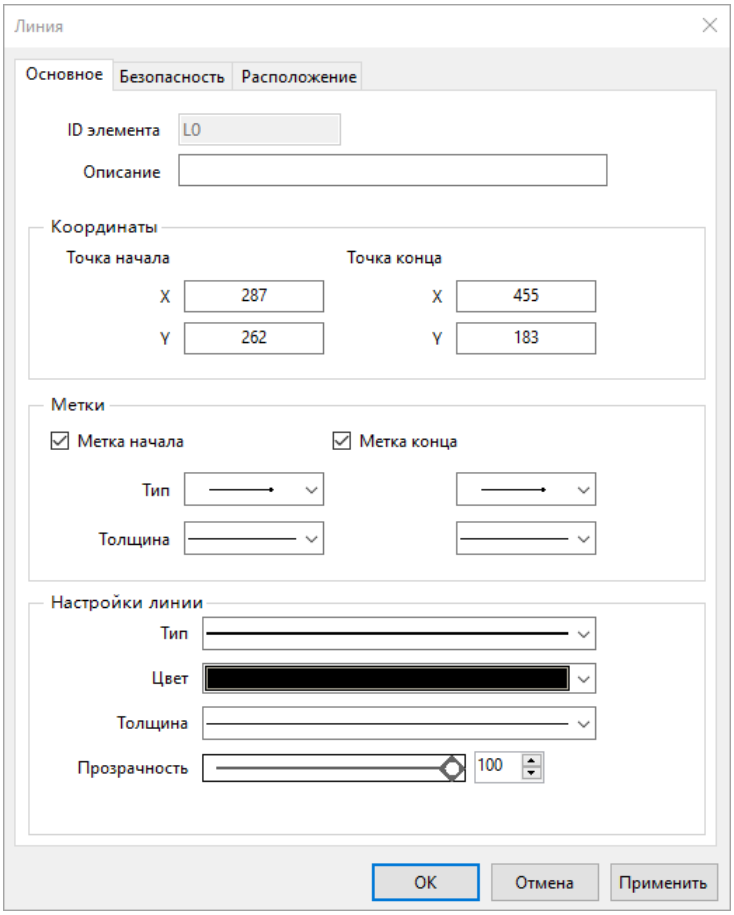


Рисунок 10.4 – Линия

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.3 – Настройки элемента Линия

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.5 – Вкладка Основное**	
Координаты	Координаты точек начала и конца линии
Метки	Настройки пиктограмм меток, отображаемых на концах линии, их толщины
Настройки линии	Настройки типа, цвета, толщины и степени прозрачности линии

10.2.2 Окружность

Элемент **Окружность** используется для отображения на экране панели окружности.

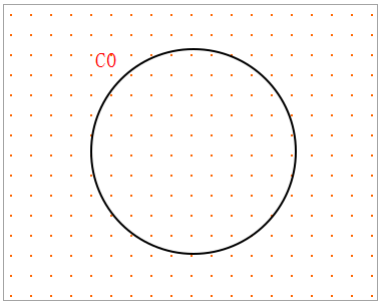
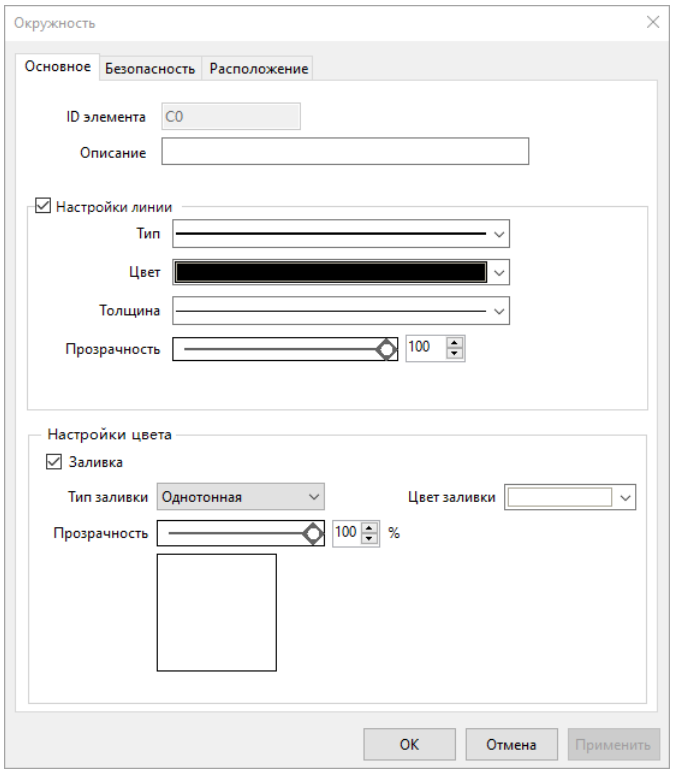


Рисунок 10.6 – Окружность

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.4 – Настройки элемента Окружность

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.7 – Вкладка Основное**	
Настройки линии	Настройки типа, цвета, толщины и степени прозрачности контура окружности
Заливка	Тип, цвет и степень прозрачности заливки окружности. Поддерживаемые типы: • однотонная; • двухцветный градиент; • узор. Для типа **Узор** задаётся тип и цвет линий узора

10.2.3 Прямоугольник

Элемент **Прямоугольник** используется для отображения на экране панели прямоугольника.

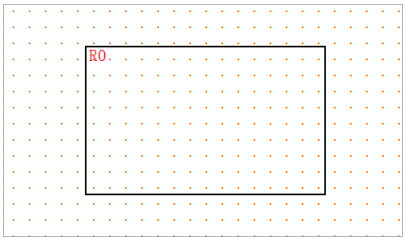
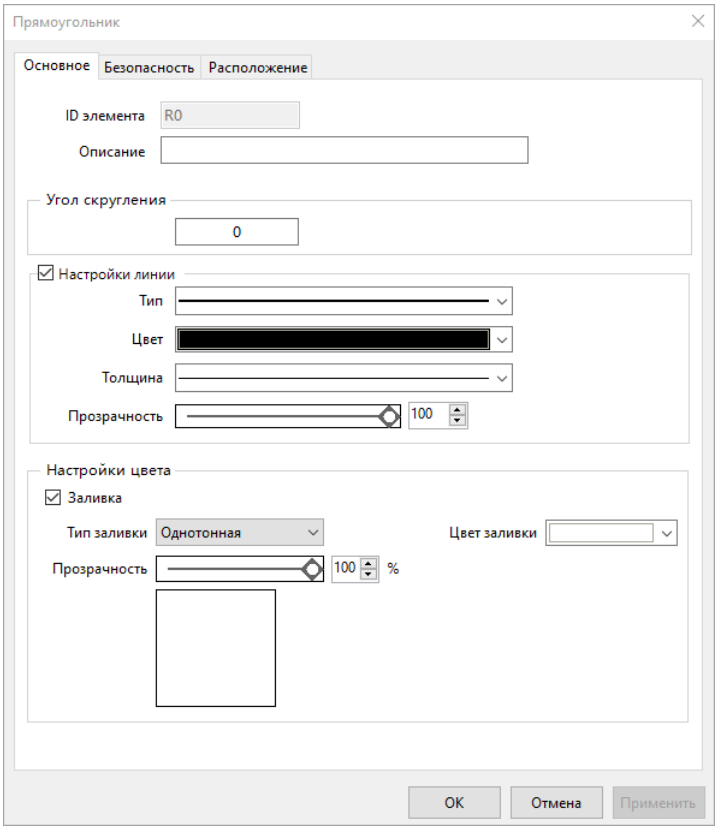


Рисунок 10.8 – Прямоугольник

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.5 – Настройки элемента Прямоугольник

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.9 – Вкладка Основное**	
Угол скругления	Угол скругления углов прямоугольника. Диапазон возможных значений зависит от ширины и высоты элемента (0…85)
Настройки линии	Настройки типа, цвета, толщины и степени прозрачности контура прямоугольника
Заливка	Тип, цвет и степень прозрачности заливки прямоугольника. Поддерживаемые типы: • однотонная; • двухцветный градиент; • узор. Для типа **Узор** задаётся тип и цвет линий узора

10.2.4 Дуга

Элемент **Дуга** используется для отображения на экране панели кривой.

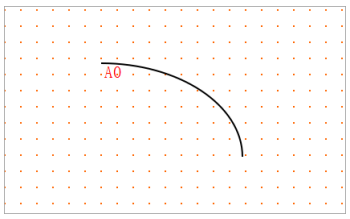
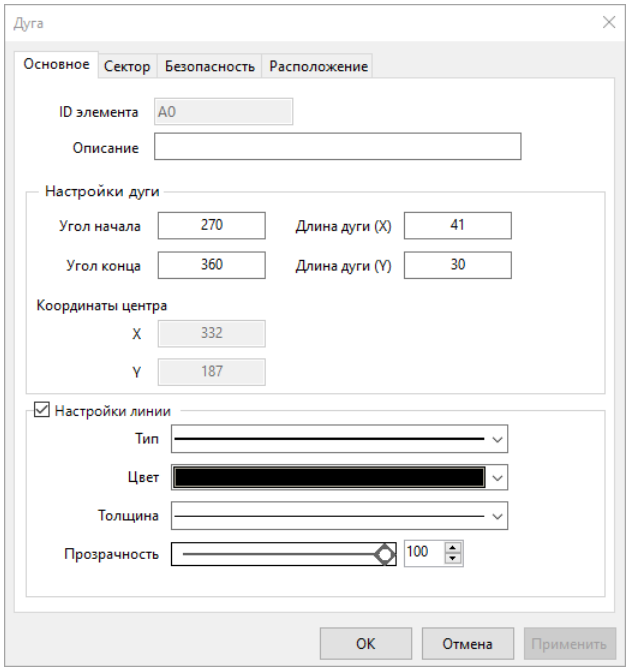


Рисунок 10.10 – Дуга

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.6 – Настройки элемента Дуга

Настройка	Описание
Вкладка **Основное**	
	
Рисунок 10.11 – Вкладка Основное	
Настройки дуги	Длина дуги по осям X и Y, а также угол между нормалью и начальной/ конечной точкой дуги. Также отображаются координаты центра дуги (недоступны для редактирования)
Настройка линии	Настройки типа, цвета, толщины и степени прозрачности линии
Вкладка **Сектор**	
Отрисовать сектор	В случае установки галочки дуга дорисовывается до сектора
Заливка	Тип, цвет и степень прозрачности заливки сектора. Поддерживаемые типы: • однотонная; • двухцветный градиент; • узор. Для типа **Узор** задаётся тип и цвет линий узора

10.2.5 Ломаная/Многоугольник

Элемент **Ломаная/Многоуольник** используется для отображения на экране соответствующих геометрических фигур.

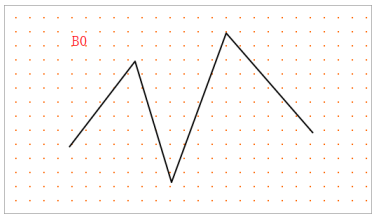
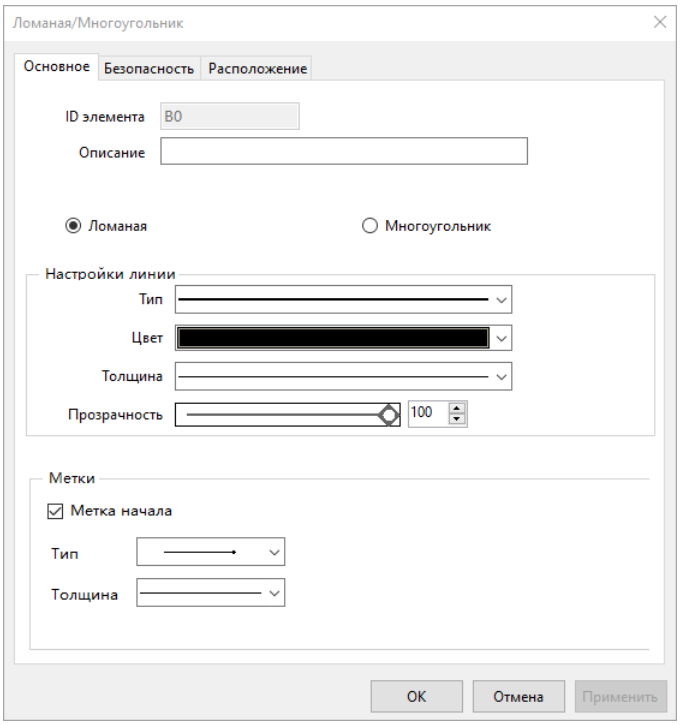


Рисунок 10.12 – Ломаная/Многоугольник

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.13 – Вкладка Основное**	
Тип фигуры	Ломаная или Многоугольник
Настройка линии	Настройки типа, цвета, толщины и степени прозрачности линии
Метки (только для типа фигуры **Ломаная**)	Настройки пиктограмм меток, отображаемых на концах линии, и их толщины
Заливка (только для типа фигуры **Многоугольник**)	Тип, цвет и степень прозрачности заливки многоугольника. Поддерживаемые типы: • однотонная; • двухцветный градиент; • узор. Для типа **Узор** задаётся тип и цвет линий узора

10.2.6 Таблица

Элемент **Таблица** используется для отображения на экране шаблона таблицы.

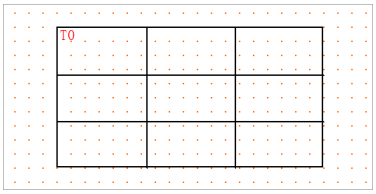
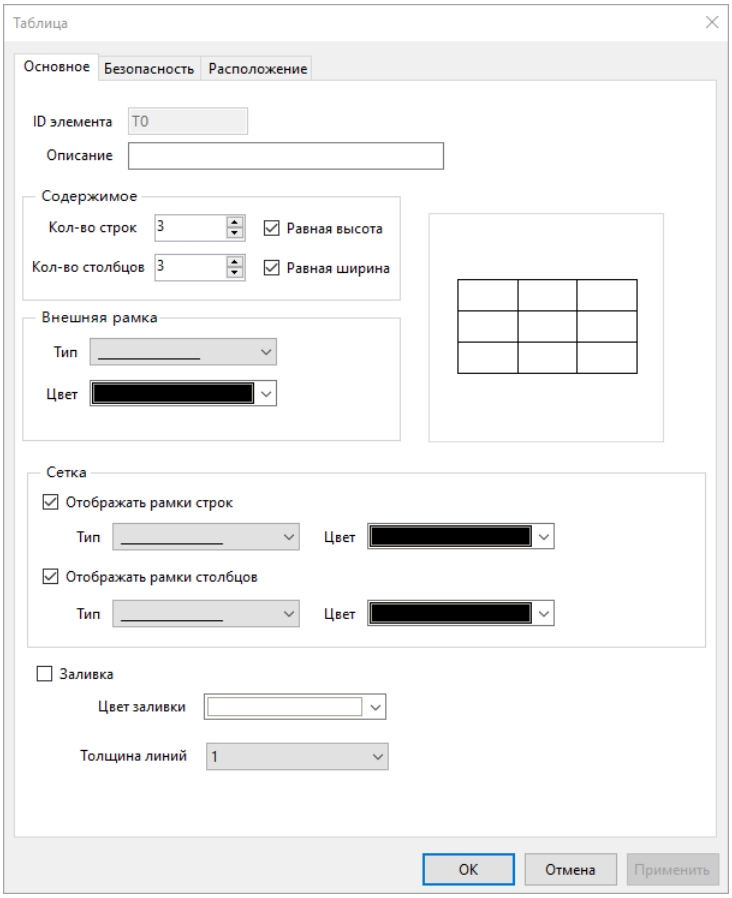


Рисунок 10.14 – Таблица

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.7 – Настройки элемента Таблица

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.15 – Вкладка Основное**	
Содержимое	Количество столбцов (не более **190**) и строк таблицы (не более **100**). Галочки **Равная высота** и **Равная ширина** позволяют выровнять ячейки таблицы
Внешняя рамка	Тип линии и цвет внешней рамки таблицы
Сетка	Настройка отображения рамок строк и столбцов (тип и цвет линии рамок)
Цвет заливки	Цвет фона таблицы
Толщина линий	Толщина линий таблицы (1…8)

10.2.7 Шкала

Элемент **Шкала** используется для отображения на экране шкалы.

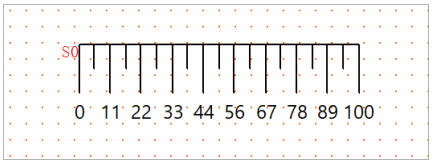
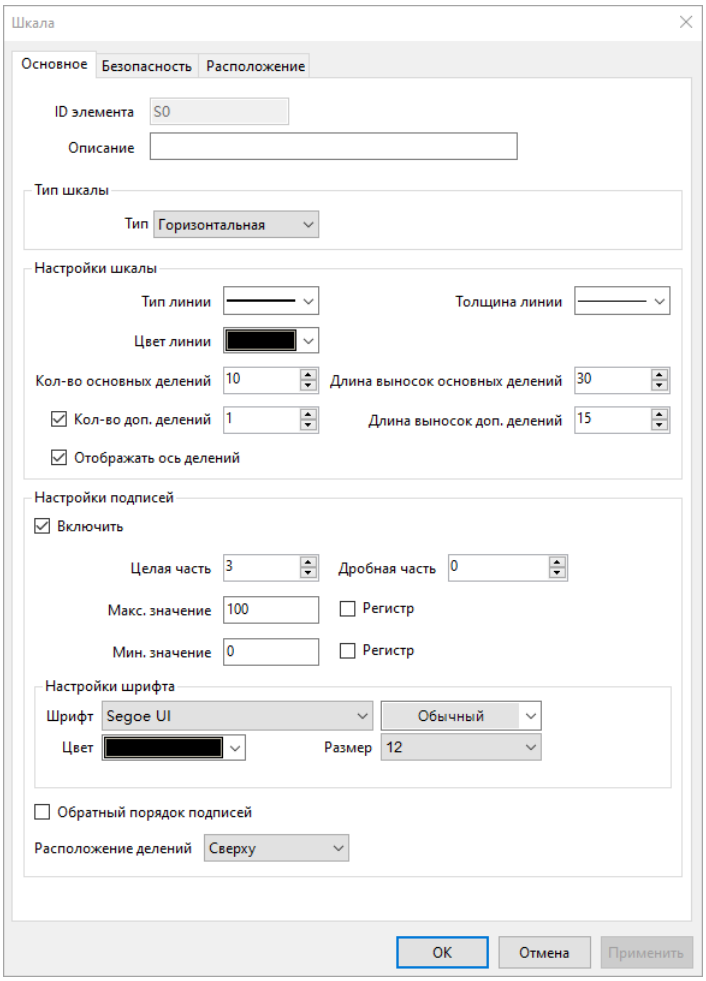


Рисунок 10.16 – Шкала

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.8 – Настройки элемента Шкала

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.17 – Вкладка Основное**	
Тип шкалы	Возможные варианты: • горизонтальная; • вертикальная; • верхний полукруг; • нижний полукруг; • круглая; • настраиваемая (устанавливается угол начала и конца).

Продолжение таблицы 10.8

Настройка	Описание
Настройки шкалы	Доступны следующие настройки: • тип, толщина и цвет линий шкалы; • количество основных делений шкалы и длина их выносок; • количество дополнительных делений шкалы (в случае установки галочки) и длина их выносок; • отображение оси шкалы.
Настройки подписей	В случае установки галочки рядом с делениями шкалы отображаются подписи. Для подписей настраивается: • максимальное количество знаков целой и дробной части; • минимальное и максимальное значение (задается константами или выбором адресов регистров); • шрифт, стиль шрифта, цвет и размер текста.
Обратный порядок подписей	В случае установки галочки подписи отображаются в обратном порядке
Расположение делений	Для типа шкалы **Горизонтальная**: сверху, снизу или по центру. Для типа шкалы **Вертикальная**: слева, справа или по центру. Для остальных типов шкалы: внутри, снаружи или по центру.

10.2.8 QR-код

Элемент **QR-код** используется для отображения на экране QR-кода или штрихкода.

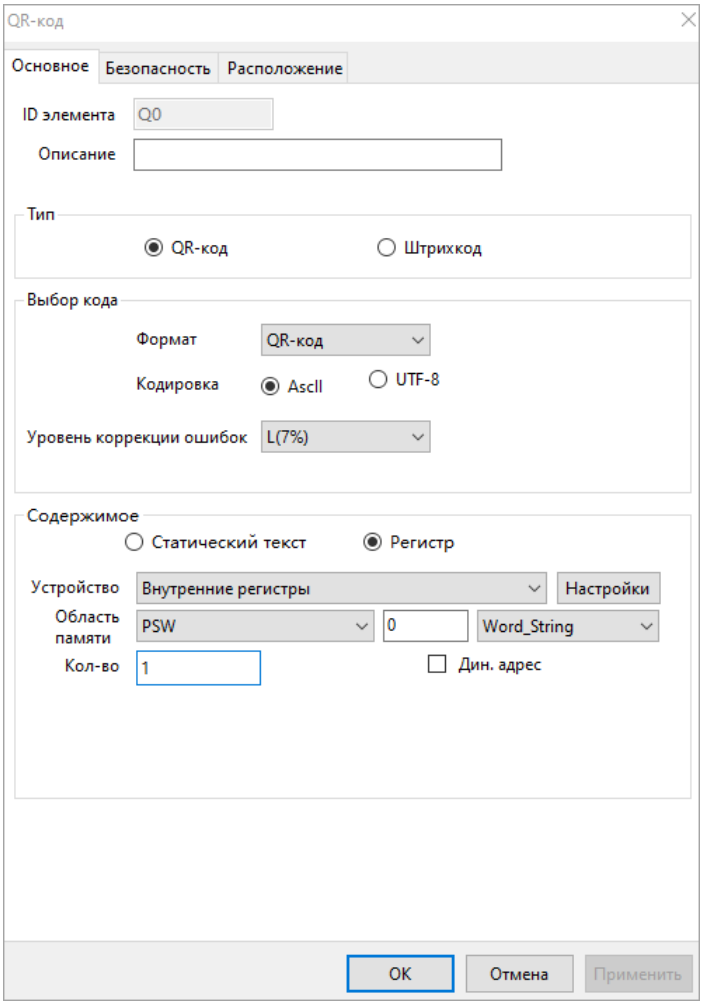


Рисунок 10.18 – QR-код

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.9 – Настройки элемента QR-код

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.19 – Вкладка Основное**	
Тип	QR-код или штрихкод
Формат	Для QR-кодов: QR, DataMatrix или PDF417 Для штрихкодов: Code128
Кодировка	Для QR-кодов: ASCII или UTF-8 Для остальных типов – только ASCII
Уровень коррекции ошибок	Только для QR-кодов. Возможные значения: L (7%), M (15%), Q (25%), H (30%)
Содержимое	Статический текст или адрес регистра, начиная с которого хранятся данные текста. Не более 200 символов

10.2.9 Изображение

Элемент **Изображение** используется для отображения на экране графического файла.

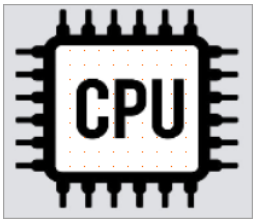
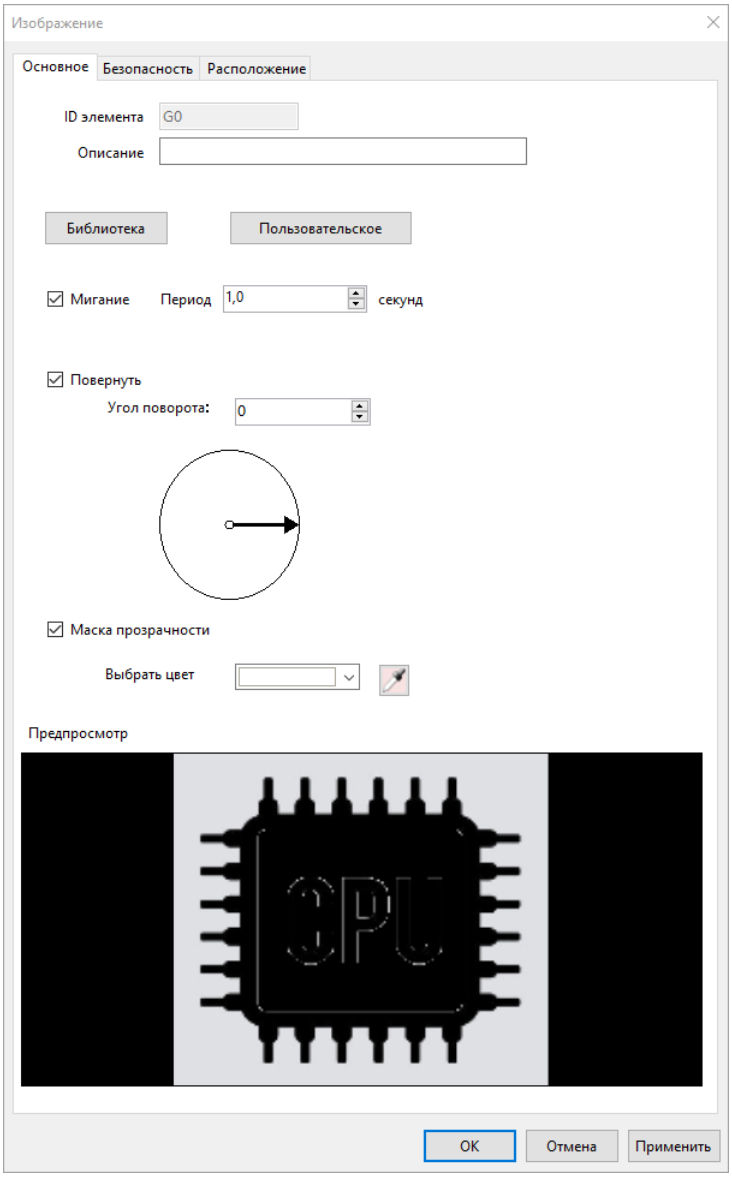


Рисунок 10.20 – Изображение

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.10 – Настройки элемента Изображение

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.21 – Вкладка Основное**	
Библиотека	Выбор изображения из библиотеки изображений

Продолжение таблицы 10.10

Настройка	Описание
Пользовательское	Выбор пользовательского изображения. Поддерживаемые форматы: .png, .jpg, .bmp, .gif Поддерживаются анимированные .gif Анимированные .png (APNG) не поддерживаются Для использования изображения формата .svg необходимо добавить его в библиотеку изображений и выбрать через кнопку **Библиотека**
Мигание	В случае установки галочки изображение будет мигать с заданным периодом (в секундах)
Циклическое воспроизведение	Данная настройка доступна только для анимированных .gif вместо настройки **Мигание**. В случае установки галочки анимация воспроизводится по кругу
Повернуть	В случае установки галочки можно повернуть изображение на заданный угол, указав значение или с помощью поворота стрелки
Маска прозрачности	В случае установки галочки можно выбрать цвет, который при отрисовке изображения будет обрабатываться как прозрачный

10.2.10 Динамическое изображение

Элемент **Динамическое изображение** используется для отображения на экране последовательно переключаемых графических файлов.

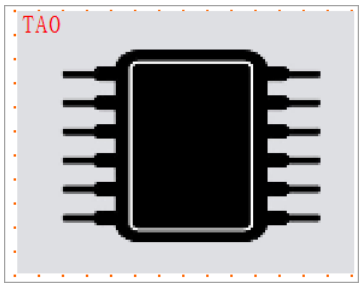
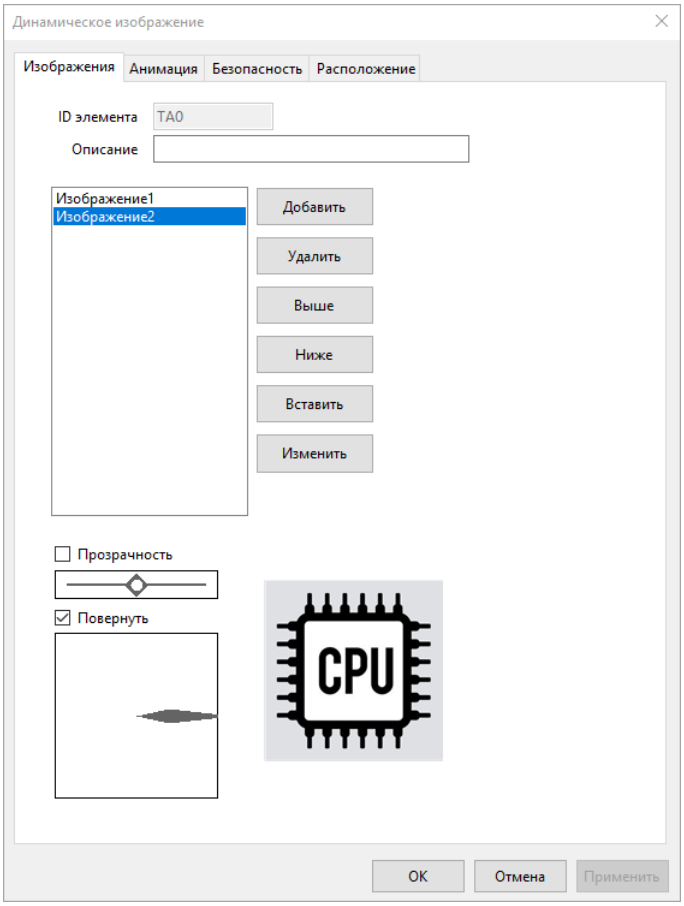


Рисунок 10.22 – Динамическое изображение

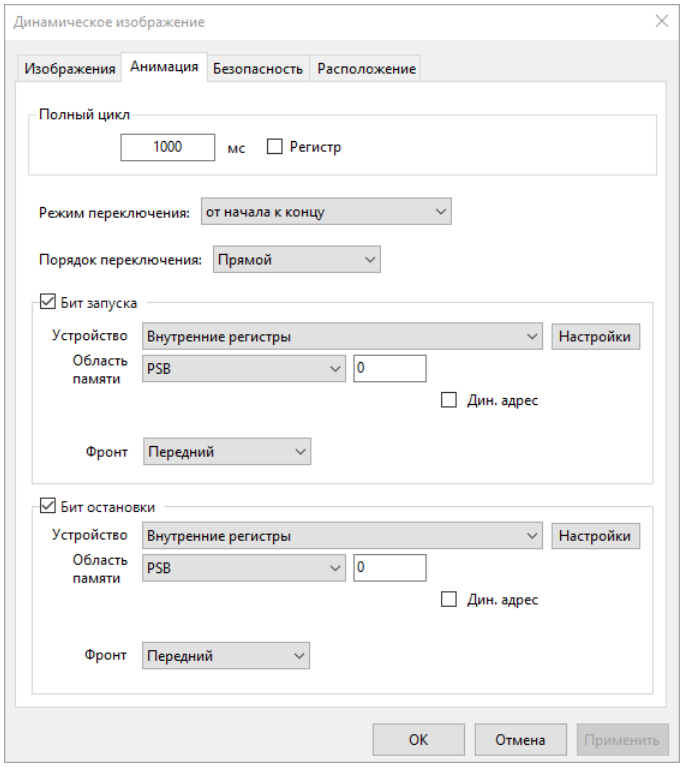
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.11 – Настройки элемента Динамическое изображение

Настройка	Описание
Вкладка **Изображения**  **Рисунок 10.23 – Вкладка Изображение**	
Кнопки	Команды добавления, удаления, настройки порядка, вставки и замены изображений. Поддерживаемые форматы: .png, .jpg, .bmp
Прозрачность	В случае установки галочки можно настроить прозрачность изображения
Повернуть	В случае установки галочки можно повернуть изображение на заданный угол путём ввода значения или с помощью поворота стрелки

Продолжение таблицы 10.11

Настройка	Описание
Вкладка **Анимация**  **Рисунок 10.24 – Вкладка Анимация**	
Полный цикл	Время (в миллисекундах), за которое переключаются все изображения. Задается константой или выбором адреса регистра
Режим переключения	Возможные варианты: • от начала к концу (1-2-3-4-1-2-3-4-…); • от начала к концу, от конца к началу (1-2-3-4-3-2-1-2-3-4…)
Порядок переключения	Возможные варианты: • прямой – от первого изображения в списке к последнему; • обратный – от последнего изображения в списке к первому; • в случайном порядке.
Бит запуска	В случае установки галочки запуск переключения изображений начинается по фронту (переднему или заднему) выбранного бита
Бит остановки	В случае установки галочки по фронту (переднему или заднему) выбранного бита происходит остановка переключения изображений. После фронта бита запуска переключение продолжается с остановленного изображения

10.2.11 Анимация перемещения

Элемент **Анимация перемещения** является невизуализируемым и используется для создания траектории, по которой будет перемещаться другой графический элемент (или элементы), сгруппированный с данным.

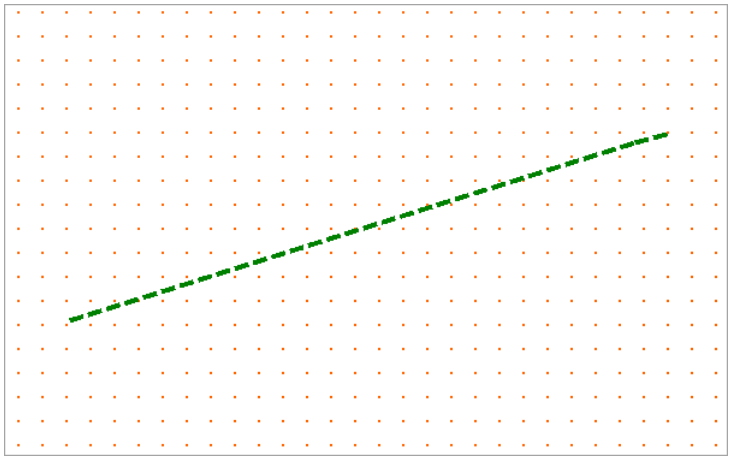
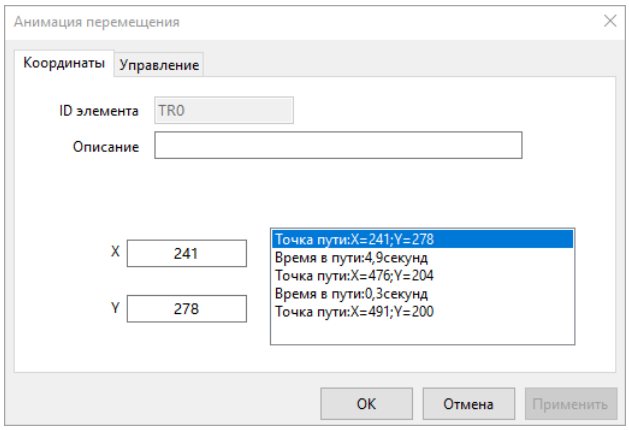


Рисунок 10.25 – Анимация перемещения

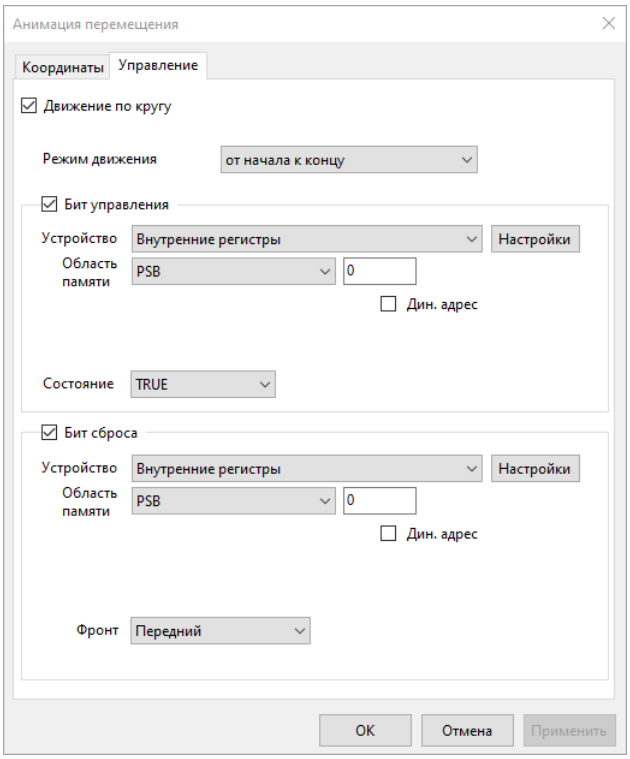
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.12 – Настройки элемента Анимация перемещения

Настройка	Описание
Вкладка **Координаты**  **Рисунок 10.26 – Вкладка Координаты**	
X, Y	Координаты точки пути
Время в пути	Время перемещения между соседними точками (в секундах)

Продолжение таблицы 10.12

Настройка	Описание
Вкладка **Управление**  **Рисунок 10.27 – Вкладка Управление**	
Движение по кругу	В случае установки галочки перемещение циклически повторяется. В случае отсутствия галочки после достижения конечной точки перемещение прекращается
Режим движения	Данная настройка обрабатывается только в случае установки галочки **Движение по кругу**. Возможные варианты: • от начала к концу – после достижения конечной точки элемент мгновенно перемещается в начальную; • от начала к концу, от конца к началу – после достижения конечной точки элемент начинает движение к начальной.
Бит управления	Бит управления перемещением. Пока бит имеет заданное значение (**FALSE** или **TRUE**) – выполняется перемещение. Если бит принимает противоположное значение, перемещение останавливается. После повторного переключения бита перемещение продолжится с того же места, а не начнётся с начальной точки
Бит сброса	В случае установки галочки по фронту (переднему или заднему) выбранного бита происходит перемещение элемента в начальную точку

10.2.12 Холст

Элемент **Холст** используется для отрисовки произвольных геометрических фигур с помощью набора функций, доступных в макросах.

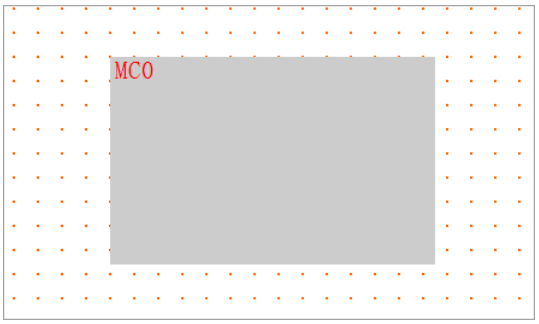
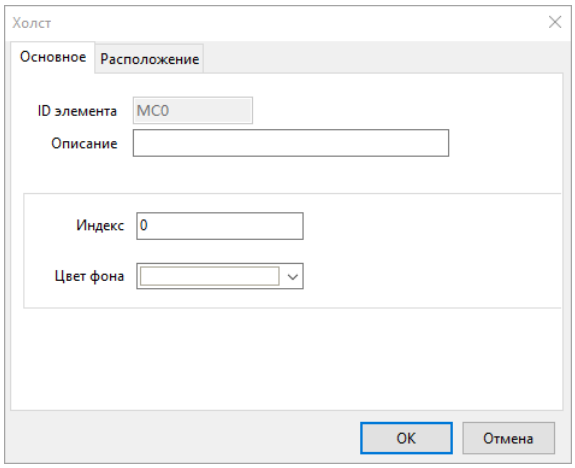


Рисунок 10.28 – Холст

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.13 – Настройки элемента Холст

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.29 – Вкладка Основное**	
Индекс	Идентификатор, используемый для обращения к элементу в коде макроса
Цвет фона	Цвет фона элемента

10.3 Текст

10.3.1 Статический текст

Элемент **Статический текст** используется для отображения на экране панели статического текста.

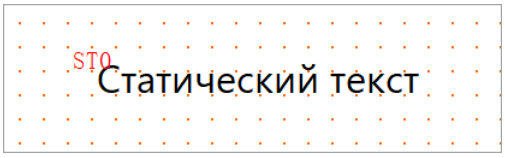
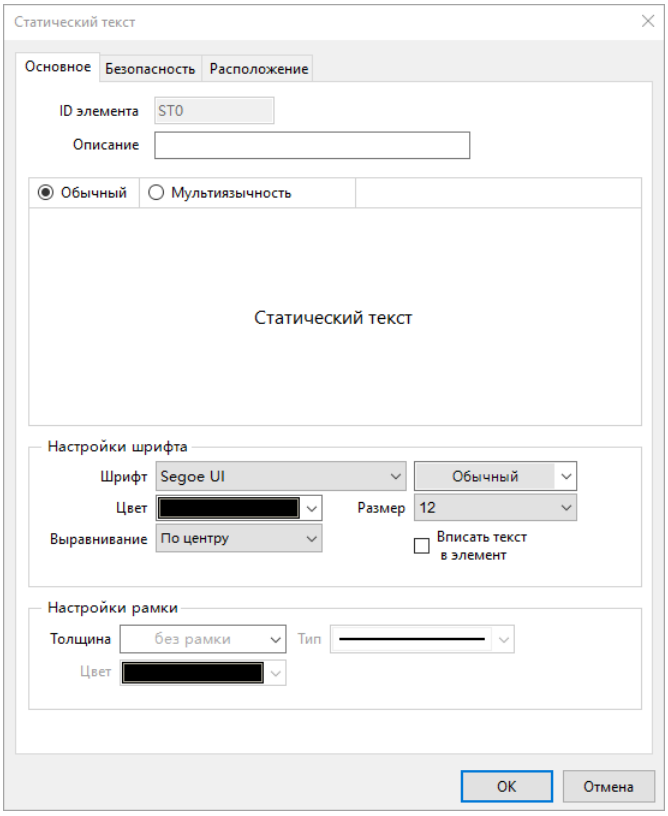


Рисунок 10.30 – Статический текст

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.14 – Настройки элемента Статический текст

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.31 – Вкладка Основное**	
Текст	Тип текста: • статический текст, вводимый пользователем в настройках элемента; • мультиязычный текст, выбираемый пользователем из библиотеки локализации
Шрифт	Выбор шрифта, типа, цвета, размера и выравнивания текста. В случае установки галочки **Вписать текст в элемент** размер шрифта будет установлен автоматически таким образом, чтобы текст не выходил за границы элемента
Настройки рамки	Настройки толщины, типа и цвета рамки элемента

10.3.2 Динамический текст

Элемент **Динамический текст** используется для отображения на экране панели динамического текста, содержимое которого определяется значением регистра, привязанного к элементу .

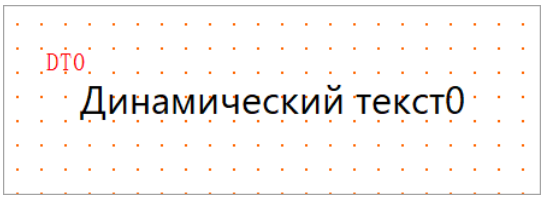
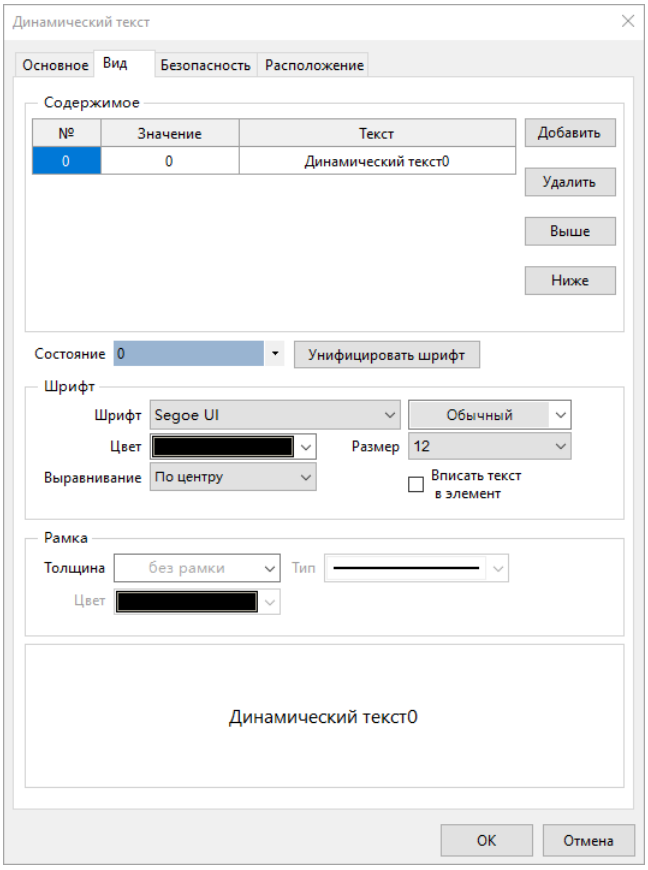


Рисунок 10.32 – Динамический текст

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.15 – Настройки элемента Динамический текст

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.33 – Вкладка Вид**	
Содержимое	Таблица возможных значений регистра элемента и соответствующих текстов (статических или из библиотеки локализации). Для выбора текста локализации нужно нажать кнопку «…».
Состояние	Выбор состояния элемента (значения регистра), к которому будут применены текущие настройки шрифта и рамки
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента

Продолжение таблицы 10.15

Настройка	Описание
Шрифт	Выбор шрифта, типа, цвета, размера и выравнивания текста. В случае установки галочки **Вписать текст в элемент** размер шрифта будет установлен автоматически таким образом, чтобы текст не выходил за границы элемента
Настройки рамки	Настройки толщины, типа и цвета рамки элемента

10.3.3 Бегущая строка

Элемент **Бегущая строка** используется для отображения на экране панели бегущей строки, содержимое которой может включать в себя статические тексты, динамические тексты и числовые значения регистров.

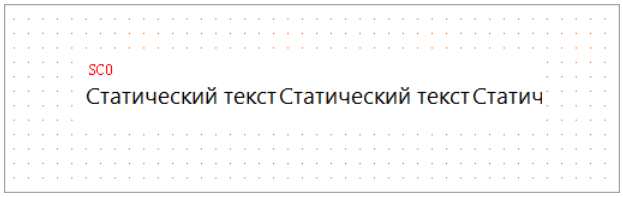
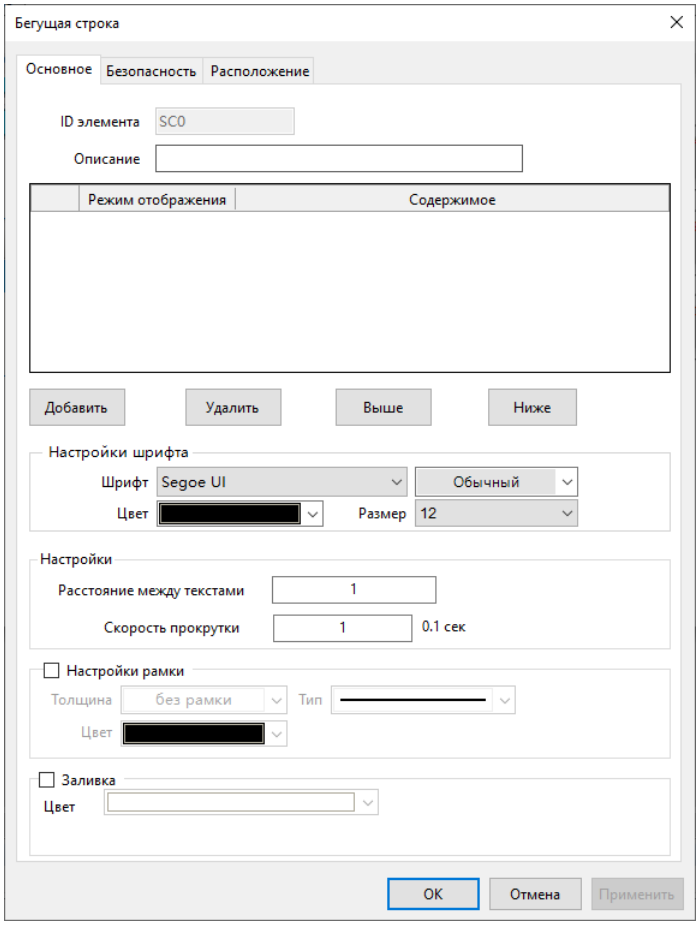


Рисунок 10.34 – Бегущая строка

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.16 – Настройки элемента Бегущая строка

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.35 – Вкладка Основное**	
Содержимое	Таблица текстов, составляющих содержимое бегущей строки. В столбце **Режим отображения** настраивается видимость конкретного текста: • Показывать всегда; • Бит видимости (с выбором значения бита, соответствующего видимости текста – **TRUE** или **FALSE**). С помощью нажатия **ПКМ** на ячейку столбца **Содержимое** происходит добавление, копирование или удаление текста. Доступные типы текстов: • Статический текст; • Динамический текст; • Значение регистра. Для варианта **Значение регистра** помимо адреса, типа и формата указывается количество отображаемых знаков целой и дробной части значения. В случае установки галочки **С ведущим 0** разряды целой части дополнятся ведущими нулями. Для редактирования текста следует нажать **ЛКМ** на ячейку
Шрифт	Выбор шрифта, типа, цвета и размера текста
Настройки	Расстояние между текстами строки в пикселях и скорость прокрутки строки, выраженная в пикселях/0.1 с
Настройки рамки	Настройки толщины, типа и цвета рамки элемента
Заливка	В случае установки галочки можно выбрать цвет фона сообщения

10.4 Ввод

10.4.1 Цифровой ввод

Элемент **Цифровой ввод** используется для отображения числового значения с возможностью ввода с помощью цифровой клавиатуры.

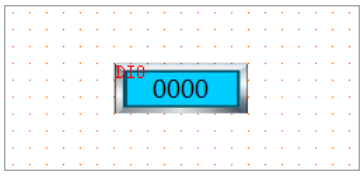
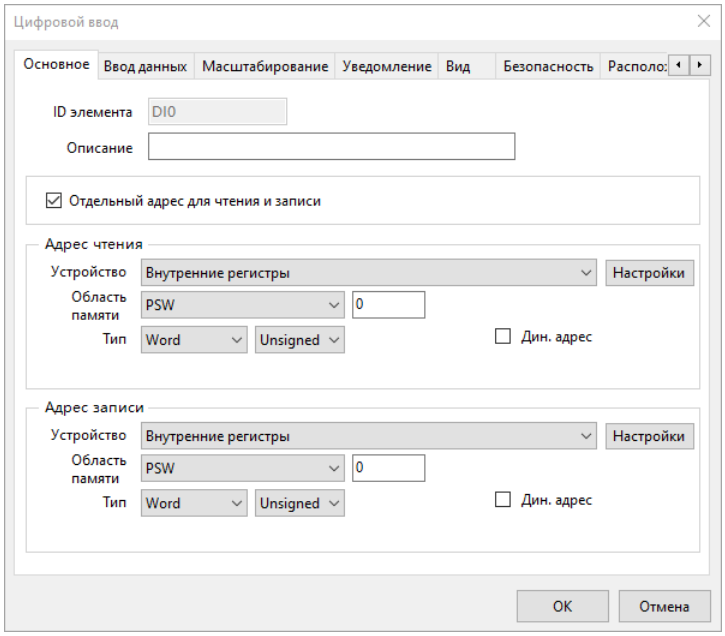


Рисунок 10.36 – Цифровой ввод

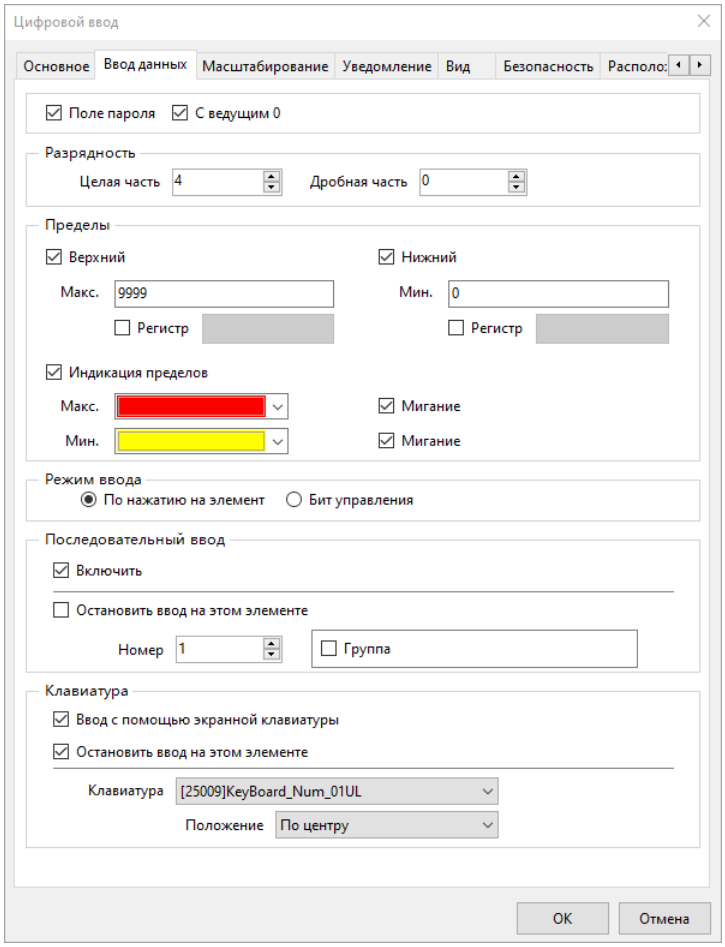
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.17 – Настройки элемента Цифровой ввод

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.37 – Вкладка Основное**	
Отдельный адрес для чтения и записи	В случае установки галочки элемент будет отображать значение одного регистра или переменной, а ввод осуществлять в другой

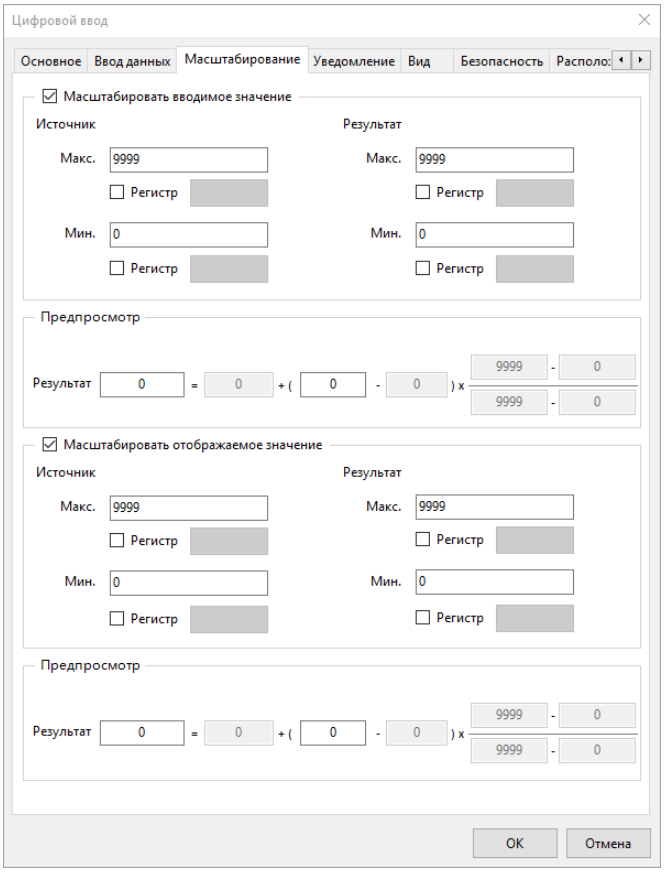
Продолжение таблицы 10.17

Настройка	Описание
Вкладка **Ввод данных**  **Рисунок 10.38 – Вкладка Ввод данных**	
Поле пароля	В случае установки галочки вместо цифр значения элемент отображает соответствующее количество символов “*”
С ведущим 0	В случае установки галочки значение дополняется ведущими нулями до указанного количества разрядов целой части
Разрядность	Количество цифр целой и дробной части значения. Для значений целочисленного типа дробная часть определяет позицию отображения десятичной точки
Пределы	В случае установки галочек можно указать минимальное и максимальное значение, которое будет доступно для ввода в элемент (константой или вводом адреса регистра). В случае установки галочки **Индикация пределов** можно указать цвет, которым будет отображаться значение при выходе за нижний или верхний предел (значение может выйти за пределы, если считывается из slave-устройства или его регистры изменяются каким-то другим элементом или master-устройством). В случае установки галочки **Мигание** вышедшее за границу значение будет мигать

Продолжение таблицы 10.17

Настройка	Описание
Режим ввода	Способ открытия клавиатуры ввода: • По нажатию на элемент; • По переднему фронту выбранного бита.
Последовательный ввод	Для режима ввода **По нажатию на элемент**: в случае установки галочки **Включить**, после ввода в элементе нового значения и нажатия на данный элемент будет открыта клавиатура ввода. После ввода в элементе нового значения и нажатия на клавишу **ENT** будет открыта клавиатура ввода для элемента **Цифровой ввод**/**Текстовый ввод** со следующим порядковым номером данной группы с установленной галочкой **Включить** или, если данный элемент не принадлежит группе, клавиатура ввода для элемента **Цифровой ввод**/**Текстовый ввод** со следующим порядковым номером с установленной галочкой **Включить**. В случае установки галочки **Остановить ввод на этом элементе** – после нажатия на клавишу **ENT** в клавиатуре ввода данного элемента последовательный ввод прекращается. Для режима ввода **По переднему фронту** выбранного бита: если к нескольким элементам цифрового ввода привязан один и тот же бит, то в случае его срабатывания клавиатуры ввода будут открываться в порядке, определяемом порядковыми номерами. Примечание для обоих режимов ввода: если элементы цифрового ввода имеют одинаковый порядковый номер, то порядок открытия клавиатур определяется номером в идентификаторе элемента

Продолжение таблицы 10.17

Клавиатура	В случае установки галочки **Ввод с помощью экранной клавиатуры** – после нажатия на элемент будет открыта выбранная пользователем клавиатура ввода с указанным относительно элемента местоположением на экране. В случае отсутствия галочки – ввод значения будет производиться с помощью клавиатуры, подключенной к порту USB A В случае установки галочки **Отображать мин. и макс. предел** будет использована клавиатура ввода с отображением нижнего и верхнего пределов
Вкладка **Масштабирование**	
	
Рисунок 10.39 – Вкладка Масштабирование	
Настройки	Настройки линейного масштабирования для вводимого и отображаемого значения **ПРИМЕЧАНИЕ** Для обновления результата в предпросмотре после ввода нового значения следует закрыть и заново открыть окно свойств элемента или изменить состояние любой из галочек в настройках.
Вкладка **Уведомление**	
Перед записью	Действие, выполняемое перед записью значения в регистр элемента – в момент открытия клавиатуры ввода
После записи	Действие, выполняемое после записи значения в регистр элемента – после нажатия на **ENT**
Действие	Доступные действия: • запись значения **TRUE** или **FALSE** в выбранный бит; • запись заданного значения в выбранный регистр.
Вкладка **Вид**	

Продолжение таблицы 10.17

Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Шрифт	Выбор шрифта, типа, цвета, размера и выравнивания текста. В случае установки галочки **Вписать текст в элемент** размер шрифта будет установлен автоматически таким образом, чтобы текст элемента не выходил за границы

10.4.2 Текстовый ввод

Элемент **Текстовый ввод** используется для отображения текстового значения с возможностью ввода с помощью символьной клавиатуры.

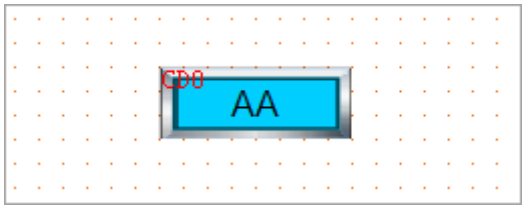
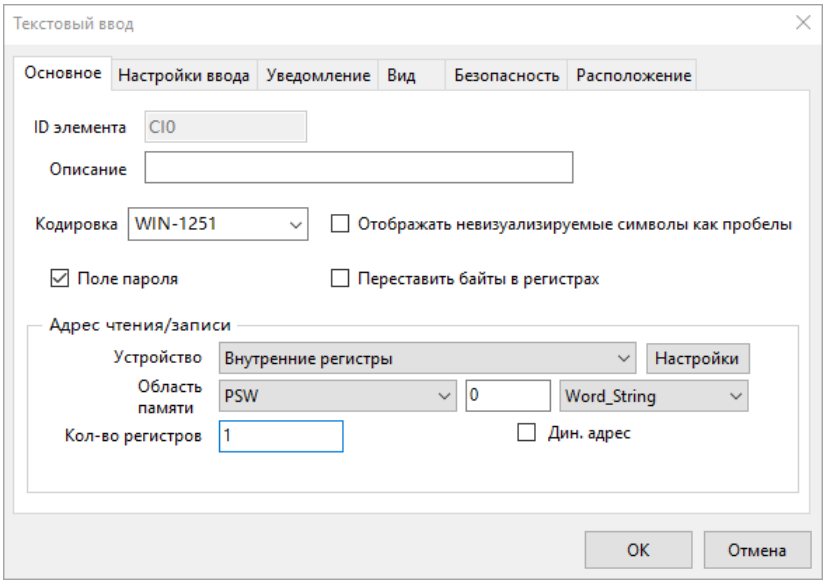


Рисунок 10.40 – Текстовый ввод

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.18 – Настройки элемента Текстовый ввод

Настройка	Описание
Вкладка **Основное**	
	
Рисунок 10.41 – Вкладка Основное	
Кодировка	Возможные значения: WIN-1251, UTF-8, UTF-16

Продолжение таблицы 10.18

Настройка	Описание
Отображать невизуализируемые символы как пробелы	Данная настройка доступна только для кодировки WIN-1251. В случае установки галочки невизуализируемые ASCII-символы отображаются как пробелы. В случае отсутствия галочки – такие символы отображаются в «**каретной нотации**»
Поле пароля	В случае установки галочки вместо символов значения элемент отображает соответствующее количество символов “*”
Переставить байты в регистрах	В случае установки галочки при размещении текста в регистрах выполняется перестановка байт. Порядок байт по умолчанию описан в таблице 7.6

Вкладка **Настройки ввода**

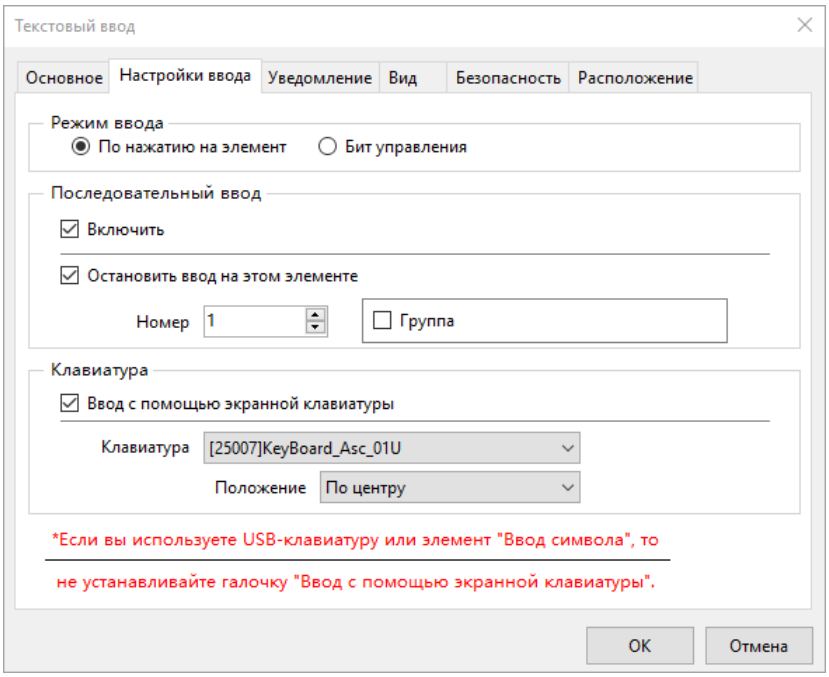


Рисунок 10.42 – Вкладка Настройки ввода

Режим ввода	Способ открытия клавиатуры ввода: • По нажатию на элемент; • По переднему фронту выбранного бита.

Продолжение таблицы 10.18

Настройка	Описание
Последовательный ввод	Для режима ввода **По нажатию на элемент**: в случае установки галочки **Включить**, после ввода в элементе нового значения и нажатия на данный элемент будет открыта клавиатура ввода. После ввода в элементе нового значения и нажатия на клавишу **ENT** будет открыта клавиатура ввода для элемента **Цифровой ввод**/**Текстовый ввод** со следующим порядковым номером данной группы с установленной галочкой **Включить** или, если данный элемент не принадлежит группе, клавиатура ввода для элемента **Цифровой ввод**/**Текстовый ввод** со следующим порядковым номером с установленной галочкой **Включить**. В случае установки галочки **Остановить ввод на этом элементе** – после нажатия на клавишу **ENT** в клавиатуре ввода данного элемента последовательный ввод прекращается. Для режима ввода **По переднему фронту** выбранного бита: если к нескольким элементам цифрового ввода привязан один и тот же бит, то в случае его срабатывания клавиатуры ввода будут открываться в порядке, определяемом порядковыми номерами. Примечание для обоих режимов ввода: если элементы цифрового ввода имеют одинаковый порядковый номер, то порядок открытия клавиатур определяется номером в идентификаторе элемента
Клавиатура	В случае установки галочки **Ввод с помощью экранной клавиатуры** после нажатия на элемент будет открыта выбранная пользователем клавиатура ввода с указанным относительно элемента местоположением на экране. В случае отсутствия галочки – ввод значения будет производиться с помощью аппаратной клавиатуры или с помощью элементов Ввод символа
Вкладка **Уведомление**	
Перед записью	Действие, выполняемое перед записью значения в регистр элемента – в момент открытия клавиатуры ввода
После записи	Действие, выполняемое после записи значения в регистр элемента – после нажатия на **ENT**
Действие	Доступные действия: • запись значения **TRUE** или **FALSE** в выбранный бит; • запись заданного значения в выбранный регистр
Вкладка **Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста. В случае установки галочки **Вписать текст в элемент** размер шрифта будет установлен автоматически таким образом, чтобы текст элемента не выходил за его границы

10.4.3 Ползунок

Элемент **Ползунок** используется для отображения числового значения с возможностью его изменения с помощью перемещения указателя.

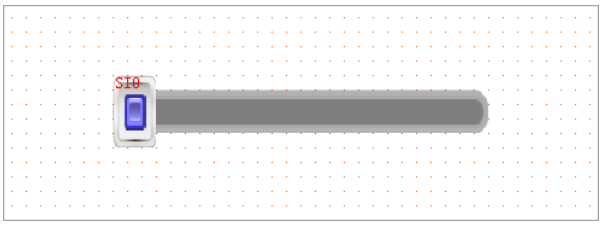
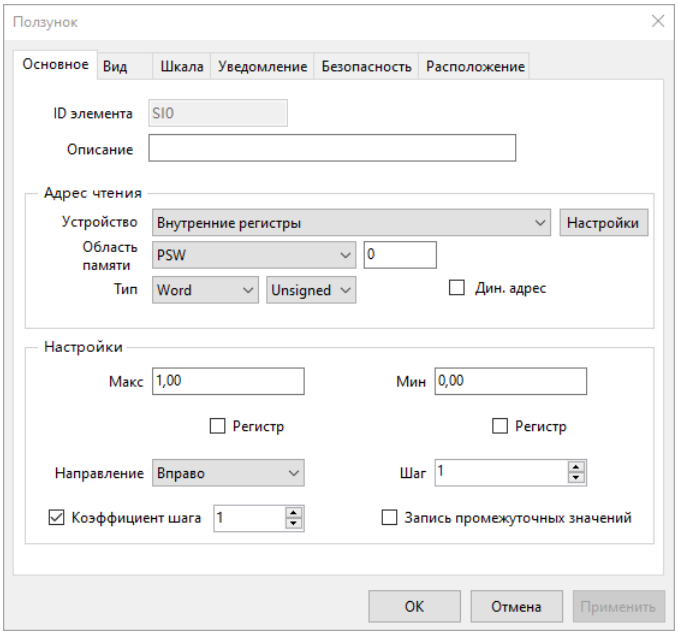


Рисунок 10.43 – Ползунок

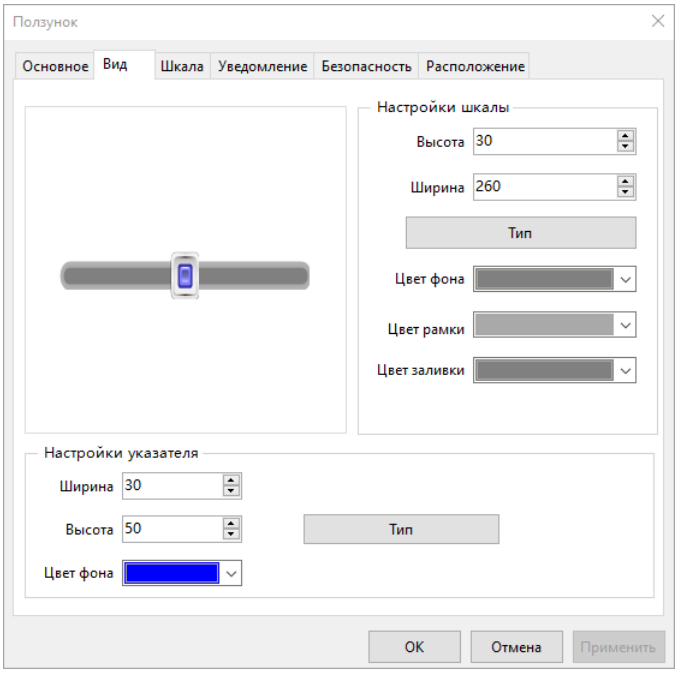
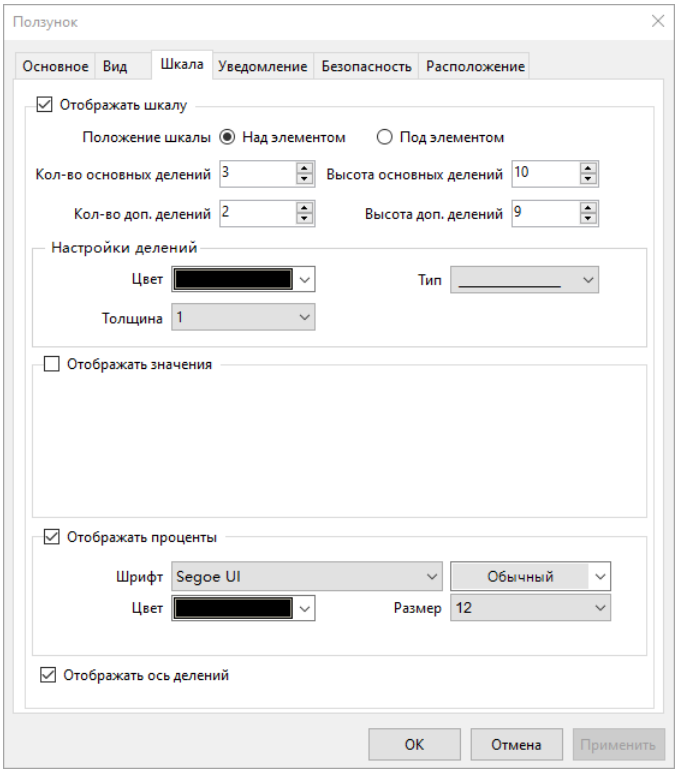
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.19 – Настройки элемента Ползунок

Настройка	Описание
Вкладка **Основное** 	
Настройки	**Мин./Макс.** – диапазон значения, которое отображается и изменяется ползунком (из интервала 0…99). Границы диапазона могут быть заданы константами или путём указания адресов регистров. **Направление** – определяет ориентацию элемента и начальное положение ползунка. Возможные значения: • вверх; • вниз; • влево; • вправо. **Шаг** – определяет дискретность изменения значения. Значение параметра домножается на коэффициент шага (в случае установки одноименной галочки) **Запись промежуточных значений** – в случае установки галочки значение регистра, привязанного к элементу, меняется по мере перемещения указателя. В случае отсутствия галочки – запись значения в регистр происходит в момент отпускания указателя

Продолжение таблицы 10.19

Настройка	Описание
Вкладка **Вид**	
	
Рисунок 10.44 – Вкладка Вид	
Настройки шкалы	Размеры, вид и цвета шкалы
Настройки указателя	Размеры, вид и цвет указателя
Вкладка **Шкала**	
	
Рисунок 10.45 – Вкладка Шкала	

Продолжение таблицы 10.19

Настройка	Описание
Отображать шкалу	В случае установки галочки отображаются настройки шкалы элемента: • Положение шкалы – над элементом или под элементом; • Количество и высота основных делений шкалы; • Количество и высота дополнительных делений шкалы.
Отображать значения	В случае установки галочки рядом с делениями отображаются соответствующие абсолютные значения. Для них можно задать количество знаков целой и дробной части, шрифт, его стиль, цвет и размер текста
Отображать проценты	Данная галочка может быть установлена только при отсутствии галочки **Отображать значения**. В случае установки галочки рядом с делениями отображаются соответствующие относительные значения в процентах. Для них можно задать шрифт, стиль, цвет и размер текста
Отображать ось делений	В случае установки галочки отображается ось делений
Вкладка **Уведомление**	
Перед записью	Действие, выполняемое перед записью значения в регистр элемента – в момент нажатия на указатель
После записи	Действие, выполняемое после записи значения в регистр элемента – после отпускания указателя
Действие	Доступные действия: • запись значения **TRUE** или **FALSE** в выбранный бит; • запись заданного значения в выбранный регистр.

10.5 Дисплей

10.5.1 Цифровой дисплей

Элемент **Цифровой дисплей** используется для отображения числового значения без возможности его ввода.

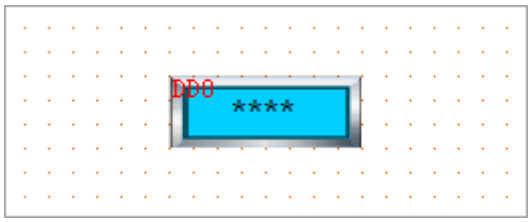


Рисунок 10.46 – Цифровой дисплей

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице 10.17.

10.5.2 Текстовый дисплей

Элемент **Текстовый дисплей** используется для отображения текстового значения без возможности его ввода.

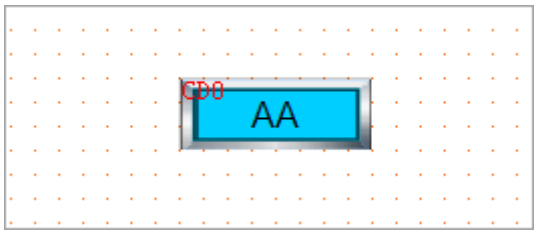


Рисунок 10.47 – Текстовый дисплей

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице 10.18.

10.6 Переключатели/индикаторы

10.6.1 Ввод символа

Элемент **Ввод символа** используется в системных окнах клавиатур для ввода ASCII-символов. В случае размещения на пользовательском экране или окне данный элемент используется совместно с элементом **Текстовый ввод** со снятой галочкой **Ввод с помощью экранной клавиатуры**.

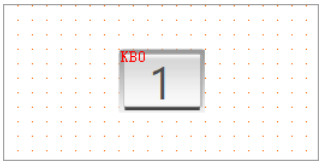
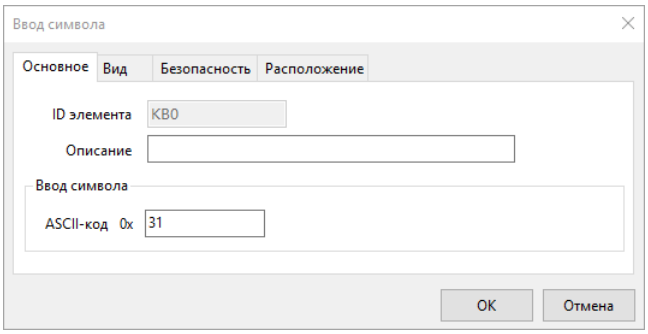


Рисунок 10.48 – Ввод символа

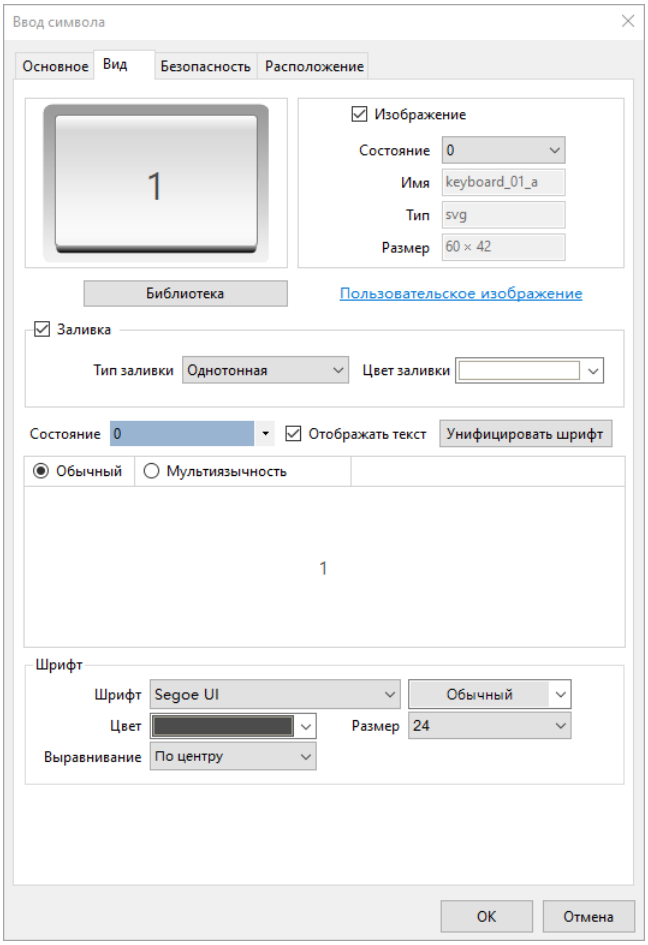
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.20 – Настройки элемента Ввод символа

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.49 – Вкладка Основное**	
ASCII-код	ASCII-код символа (в HEX), который будет введен при нажатии на кнопку
Аудиофайл	Только для модификаций СП210-07-Р, СП210-10-Р и СП210-15-Р. Выбор аудиофайла, который будет воспроизводиться при нажатии на элемент

Продолжение таблицы 10.20

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.50 – Вкладка Основное**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния кнопки: **0** – отпущена, **1** – зажата Все остальные настройки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента
Шрифт	Выбор шрифта, типа, цвета, размера и выравнивания текста

10.6.2 Индикатор

Элемент **Индикатор** используется для отображения состояния бита.

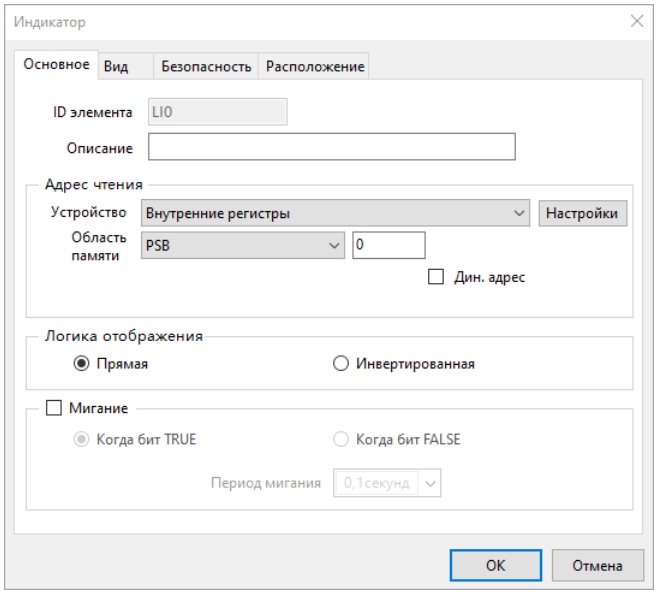


Рисунок 10.51 – Индикатор

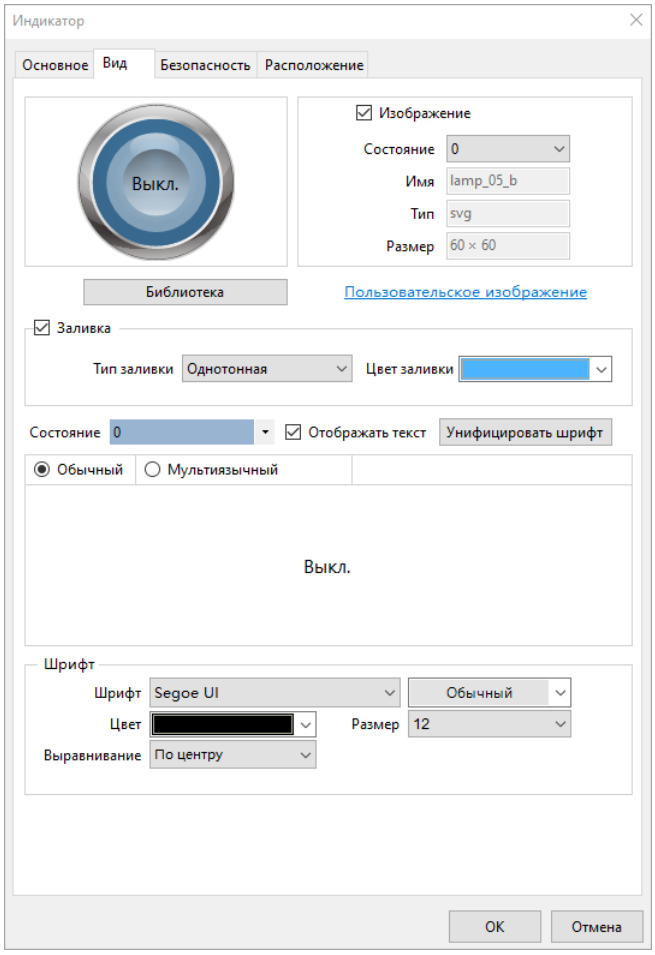
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.21 – Настройки элемента Индикатор

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.52 – Вкладка Основное**	
Логика отображения	Возможные варианты: • Прямая – включенному состоянию элемента соответствует значение бита **TRUE**, отключенному – **FALSE**; • Инвертированная – включенному состоянию элемента соответствует значение бита **FALSE**, отключенному – **TRUE**.
Мигание	В случае установки галочки элемент будет мигать с заданным периодом, когда его бит перейдёт в заданное состояние. Диапазон периода мигания: 0.1…10.0 секунд (с дискретностью в 0.1 секунды)

Продолжение таблицы 10.21

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.53 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния индикатора: **0** – отключен, **1** – включен Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.6.3 Переключатель

Элемент **Переключатель** используется для изменения значения бита.

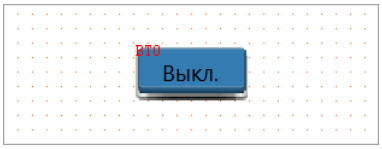
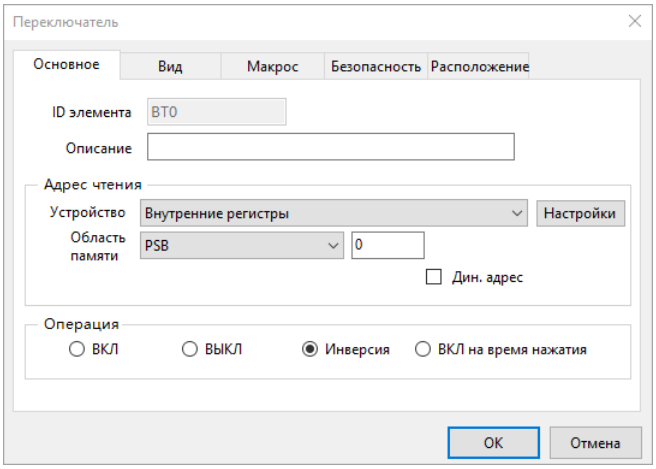


Рисунок 10.54 – Переключатель

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.22 – Настройки элемента Переключатель

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.55 – Вкладка Основное**	
Операция	Действие с битом, выполняемое при нажатии на элемент. Возможные варианты: • ВКЛ – переключение бита в значение **TRUE**; • ВЫКЛ – переключение бита в значение **FALSE**; • Инверсия – изменение значения бита на противоположное; • ВКЛ на время нажатия – установка бита в значение **TRUE** на время удержания элемента. После отпускания бит будет переключен в значение **FALSE**
Аудиофайл	Только для модификаций СП210-07-Р, СП210-10-Р и СП210-15-Р. Выбор аудиофайла, который будет воспроизводиться при нажатии на элемент
Вкладка **Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

Продолжение таблицы 10.22

Настройка	Описание
Состояние	Выбор состояния переключателя: **0** – отключен, **1** – включен Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента
Шрифт	Выбор шрифта, типа, цвета, размера и выравнивания текста
Вкладка **Макрос**	
Кнопка	Событие, при котором будут выполнены добавленные макросы – нажатие или отпускание кнопки. Для нажатия и отпускания может быть настроен вызов разных макросов
Макрос	Добавленные макросы выполняются в порядке размещения. Информация о режиме выполнения (синхронно/асинхронно) приведена в пункте 12.6.10

10.6.4 Переключатель с индикацией

Элемент **Переключатель с индикацией** используется для отображения состояния бита с возможностью его изменения. Элемент объединяет функционал элементов **Индикатор** и **Переключатель**.



Рисунок 10.56 – Переключатель с индикацией

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице 10.21 и таблице 10.22.

Таблица 10.23 – Настройки элемента Переключатель с индикацией

Настройка	Описание
Вкладка **Основное**	
Отдельный адрес для чтения и записи	В случае установки галочки элемент будет отображать значение одного бита, а изменять значение другого

10.6.5 Многопозиционный индикатор

Элемент **Многопозиционный индикатор** используется для отображения состояния регистра или набора битов. Для каждого состояния настраивается своё изображение, текст и другие параметры.

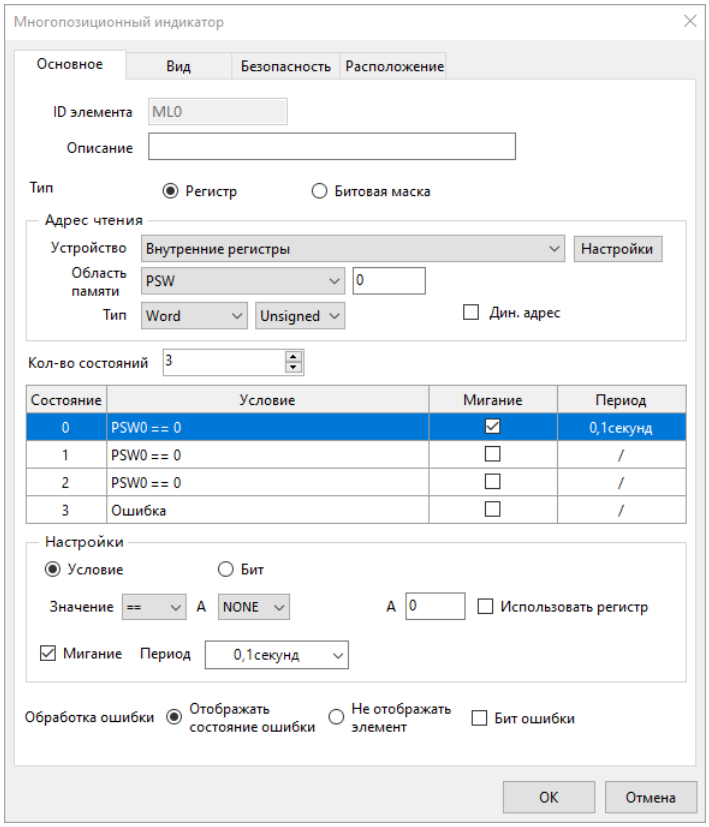


Рисунок 10.57 – Многопозиционный индикатор

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.24 – Настройки элемента Многопозиционный индикатор

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.58 – Вкладка Основное**	
Тип	Возможные варианты: • Регистр – состояние элемента определяется значением указанного регистра или его отдельных битов; • Битовая маска – состояние элемента определяется значением набора битов. Для этого режима указывается адрес начального бита и количество битов, используемое элементом. Этот набор битов интерпретируется как целочисленное значение

Продолжение таблицы 10.24

Настройка	Описание
Кол-во состояний	Количество состояний элемента. Дополнительно к ним всегда присутствует ненастраиваемое состояние **Ошибка**, соответствующее всем значениям регистра/набора битов, не указанным в элементе
Настройки	Для каждого состояния пользователь указывает условие перехода. Условие включает в себя один или три оператора, которым соответствует один или два операнда. Операнды являются целочисленными значениями, которые могут быть заданы константами или путём указания адресов их регистров. Доступные операторы: • < (меньше); • >(больше); • <= (меньше или равно); • >= (больше или равно); • == (равенство); • != (неравенство); • AND (логическое И); • OR (логическое ИЛИ). В условии может использоваться только один логический оператор. Для режима **Регистр** вместо условия можно указать требуемое состояние (**TRUE** или **FALSE**) одного из битов регистра В случае установки галочки **Мигание** в момент отображения данного состояния элемент будет мигать с заданным периодом. Диапазон периода мигания: 0.1…10.0 секунд (с дискретностью в 0.1 секунды)
Обработка ошибки	Возможные варианты: • Отображать состояние ошибки – в состоянии ошибки (см. параметр **Кол-во состояний**) элемент отображается; • Не отображать элемент – в состоянии ошибки элемент не отображается
Бит ошибки	В случае установки галочки в момент отображения состояния ошибки заданный бит имеет значение **TRUE**
Вкладка **Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния элемента. Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.6.6 Многопозиционный переключатель

Элемент **Многопозиционный переключатель** используется для отображения состояния регистра или набора битов с возможностью его изменения. Каждое нажатие на элемент приводит к переключению на следующее состояние (по кругу).

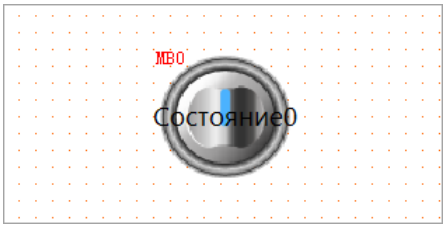
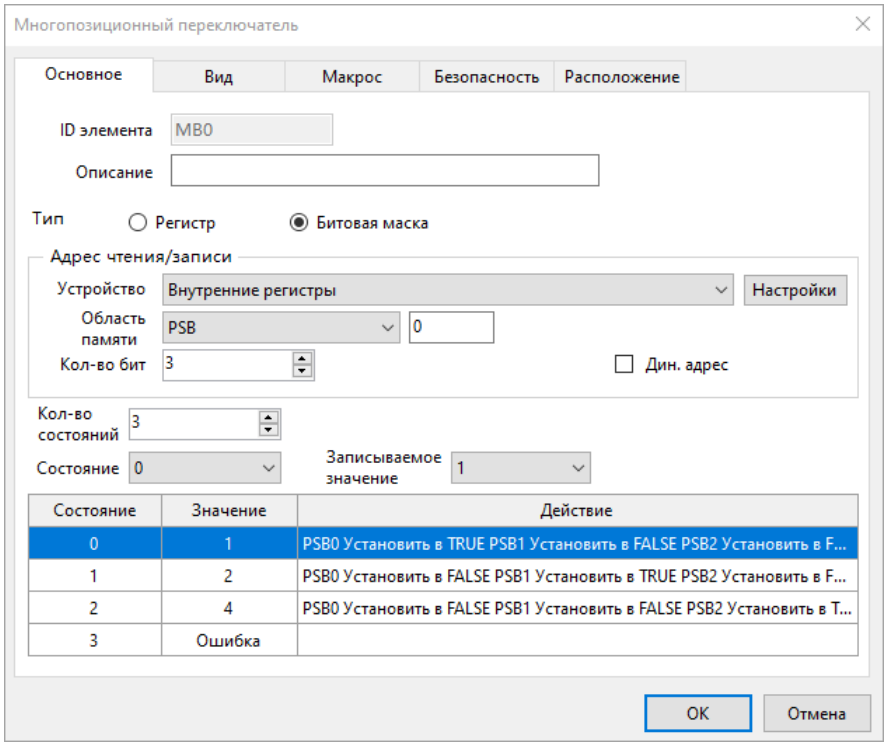


Рисунок 10.59 – Многопозиционный переключатель

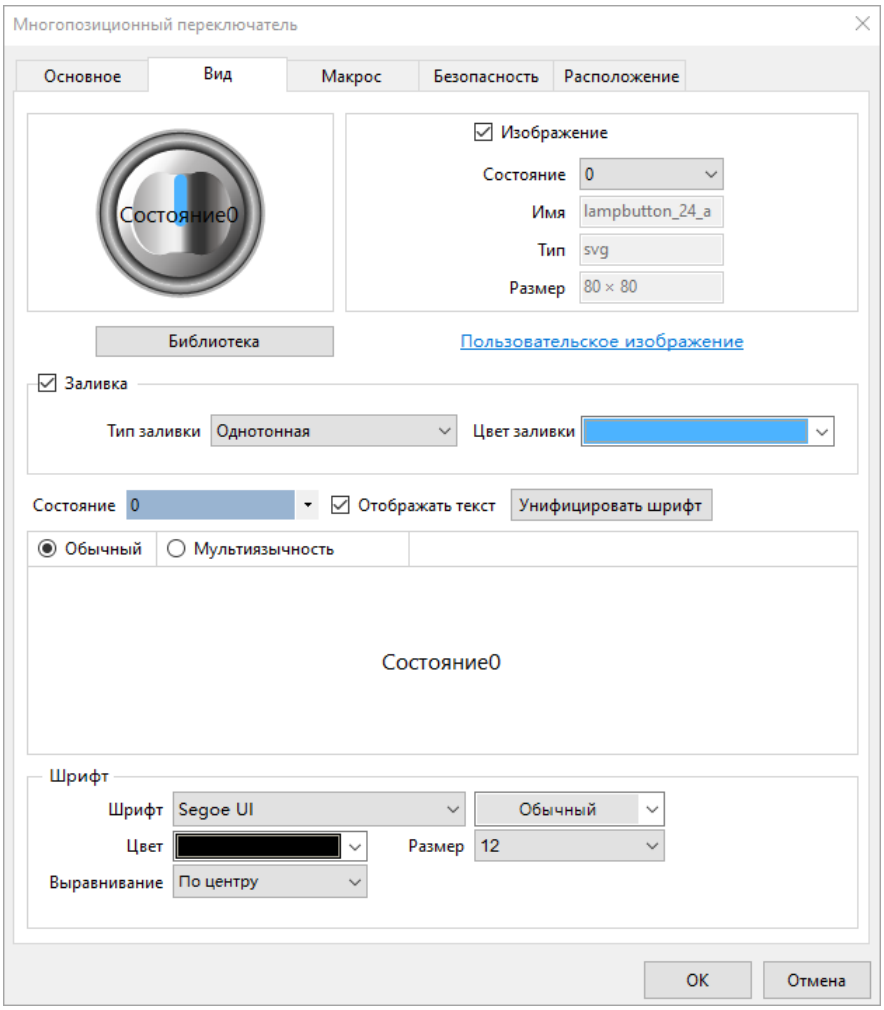
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

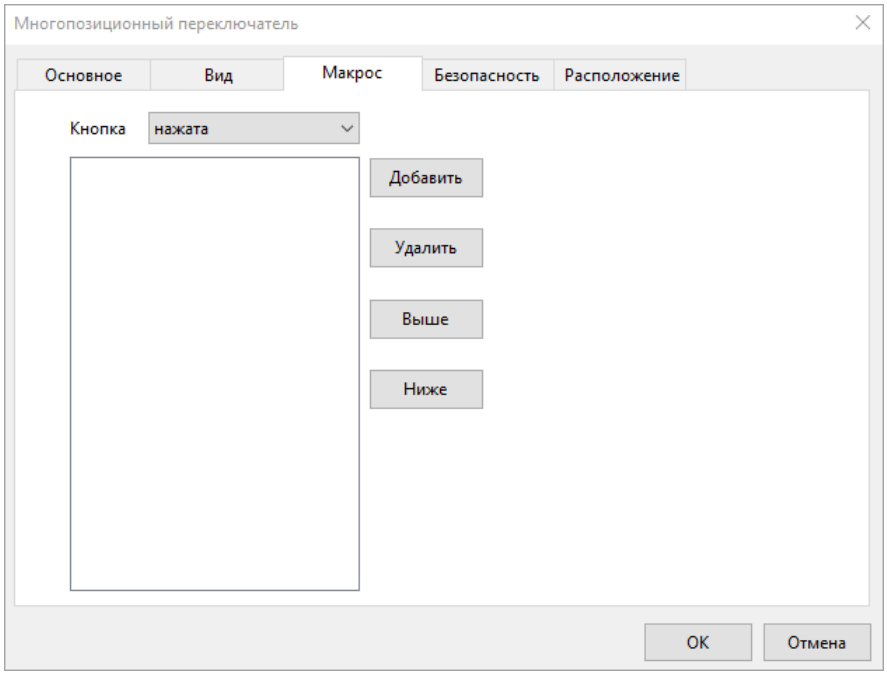
Таблица 10.25 – Настройки элемента Многопозиционный переключатель

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.60 – Вкладка Основное**	
Тип	Возможные варианты: • Регистр – состояние элемента определяется значением указанного регистра; • Битовая маска – состояние элемента определяется значением набора битов. Для этого режима указывается адрес начального бита и количество битов, используемое элементом
Кол-во состояний	Количество состояний элемента. Дополнительно к ним по умолчанию присутствует ненастраиваемое состояние **Ошибка**, соответствующее всем значениям регистра/набора битов, не указанным в элементе в столбце **Действие**. Если элемент находится в состоянии **Ошибка**, то нажатие на него приводит к переходу в первое из настроенных состояний

Продолжение таблицы 10.25

Настройка	Описание
Настройки	Для режима **Регистр**: для каждого состояния задаётся соответствующее ему значение регистра Для режима **Битовая маска**: для каждого состояния задаётся соответствующий ему набор значений битов
Вкладка **Вид**  **Рисунок 10.61 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата **.svg**, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата **.svg**, поддерживающих соответствующий параметр
Состояние	Выбор состояния элемента. Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента

Продолжение таблицы 10.25

Настройка	Описание
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста
Вкладка **Макрос**  **Рисунок 10.62 – Вкладка Макрос**	
Кнопка	Событие, при котором будут выполнены добавленные макросы – нажатие или отпускание кнопки. Для нажатия и отпускания может быть настроен вызов разных макросов
Макрос	Добавленные макросы выполняются в порядке их размещения. Информация о режиме выполнения (синхронно/асинхронно) приведена в пункте 12.6.10

10.6.7 Функциональная кнопка

Элемент **Функциональная кнопка** используется для размещения на экране панели кнопки, при нажатии или отпускании которой выполняется набор заданных действий. Действия выполняются в порядке добавления.

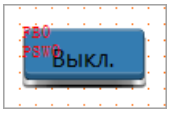


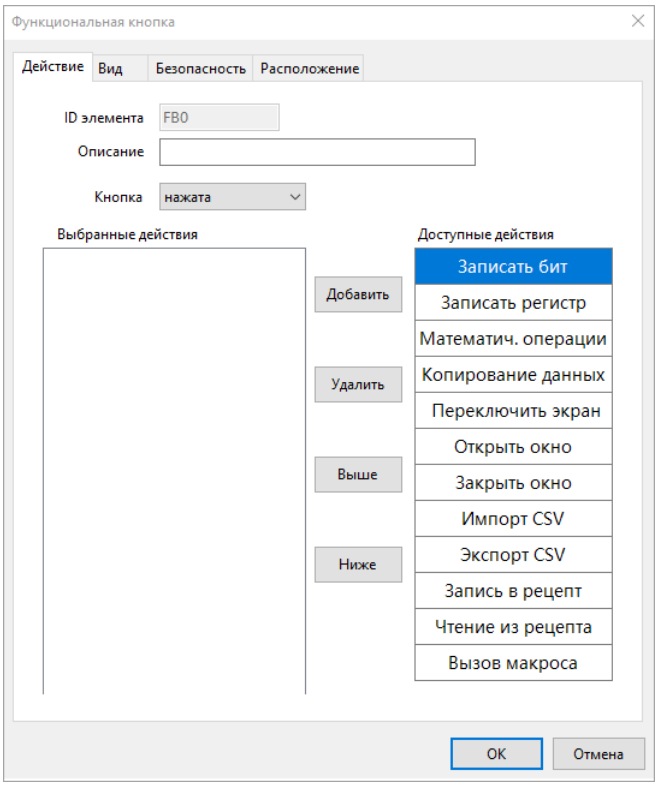
Рисунок 10.63 – Функциональная кнопка

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице 10.26.

Доступные действия описаны в таблице 10.27.

Таблица 10.26 – Настройки элемента Функциональная кнопка

Настройка	Описание
Вкладка **Действие**  **Рисунок 10.64 – Вкладка Действие**	
Кнопка	Событие, при котором будут выполнены добавленные действия – нажатие или отпускание кнопки. Для нажатия и отпускания может быть настроен вызов разных действий
Аудиофайл	Только для модификаций СП210-07-Р, СП210-10-Р и СП210-15-Р. Выбор аудиофайла, который будет воспроизводиться при нажатии на элемент

Продолжение таблицы 10.26

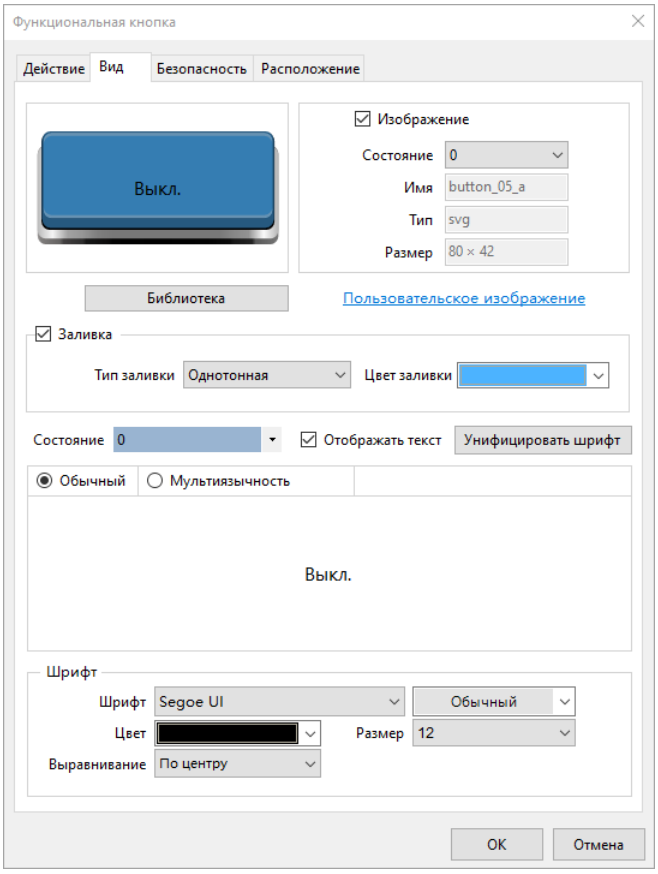
Настройка	Описание
Вкладка **Вид**  **Рисунок 10.65 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния кнопки: **0** – отпущена, **1** – зажата Все остальные настройки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе

Таблица 10.27 – Действия элементов Функциональная кнопка и Функциональная область

Действие	Описание
Записать бит	Изменение значения указанного бита: • Установить в TRUE – переключение бита в значение **TRUE**; • Установить в FALSE – переключение бита в значение **FALSE**; • Инвертировать – изменение значения бита на противоположное.
Записать регистр	Изменение значения указанного регистра: • Записать значение – запись в данный регистр константы или значения выбранного регистра; • Сложение/Вычитание – увеличение или уменьшение значения данного регистра на константу или значение выбранного регистра. В случае установки галочки **Переполнение** при достижении верхнего/нижнего предела данный регистр получит значение указанного нижнего/верхнего предела (задаётся константой или путём указания адреса регистра). В случае отсутствия галочки **Переполнение** – после достижения верхнего/нижнего предела действие перестанет выполняться
Математич. операции	Выполнение арифметической операции – сложения, вычитания, умножения или деления. Операнды и пределы задаются в виде констант или путём указания адресов регистров. Результат операции ограничивается значениями нижнего/верхнего предела и записывается в указанный регистр
Копирование данных	Копирование указанного количества регистров или бит. Параметр **Кол-во** определяет количество копируемых объектов заданного типа (Word, Dword, DDword), а не их регистров
Переход на экран	Переключение отображаемого на панели экрана – на начальный, предыдущий отображаемый или заданный. В случае установки галочки **Открыть окно авторизации, если не хватает уровня доступа** при нажатии на данный элемент пользователем, не имеющего требуемого уровня доступа для указанного экрана, будет открыто системное окно **Авторизация** (25001)
Открыть окно	В зависимости от выбранного режима – переключение текущего окна на указанное (если функциональная кнопка расположена в окне) или открытие указанного окна. В случае установки галочки **Открыть окно авторизации, если не хватает уровня доступа** при нажатии на данный элемент пользователем, не имеющего требуемого уровня доступа для указанного окна, будет открыто системное окно **Авторизация** (25001)
Закрыть окно	В зависимости от выбранного режима – закрытие указанного окна или всех открытых окон

Продолжение таблицы 10.27

Действие	Описание
Экспорт CSV	Экспорт значений указанных регистров в .csv-файл (разделитель – точка с запятой), размещённый на USB-накопителе. Каждая операция приводит к созданию нового файла или полной перезаписи существующего (т. е. режим дозаписи не поддерживается). В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk** Название файла может быть задано как: • фиксированное; • формируемое на основе текущей метки времени (пример: **2025.10.08-08_17_35.csv**); • формируемое на основе содержимого указанных регистров (в регистрах должна быть размещена строка с именем файла в кодировке ASCII, без использования символов кириллицы и спецсимволов. Максимальное количество регистров – **256**). **Кол-во строк** – количество строк в создаваемом файле, задаётся константой или путём указания адреса регистра. Максимальное количество строк – **65355**. Каждая строка таблицы **Содержимое файла** описывает один сохраняемый параметр. Для каждого параметра указывается имя, тип и формат данных, а также количество знаков целой и дробной части. Для параметров строкового типа указывается кол-во объектов (Word, Dword или DDWord), которые занимает строка. Первая строка файла содержит имена параметров. Каждая их остальных строк содержит значения параметров. Общее количество регистров, занимаемых экспортируемыми данными, определяется как произведение значения параметра **Кол-во строк** на количество регистров, занимаемых одной строкой (которое зависит от типа данных параметров). Пример: • Адрес начального регистра = PSW0; • Файл содержит два параметра – один типа Word и один типа Dword); • Кол-во строк = 5. Тогда данные одной строки файла занимают 1 + 2 = 3 регистра, общее кол-во регистров = 3 · 5 = 15, и они занимают адреса PSW0…PSW14, где PSW0…PSW2 – данные первой строки файла, PSW3…PSW5 – данные второй строки файла и т. д. **Статус выполнения** – указанный бит имеет значение **TRUE** в процессе экспорта, после его завершения сбрасывается в **FALSE**. **Результат выполнения** – в указанный регистр будет записан результат завершения экспорта или код ошибки: • 0 – экспорт завершен без ошибок; • 1 – некорректное имя файла; • 2 – ошибка индекса файла; • 3 – указанный путь не существует • 4 – ошибка создания файла • 6 – ошибка записи в файл • 7 – ошибка чтения данных из slave-устройства; • 8 – USB-накопитель не подключен.

Продолжение таблицы 10.27

Действие	Описание
	Процесс выполнения – в указанный регистр записывается степень прогресса операции экспорта в процентах. Значение **100** соответствует завершению экспорта
Импорт CSV	Импорт значений из указанного .csv-файла (разделитель – точка с запятой), размещённого на USB-накопителе, в заданные регистры панели или slave-устройства. Представляет собой обратную процедуру по отношению к экспорту. Настройки аналогичны. Отличия: • для указания имени файла может использоваться графический элемент **Проводник**; • указываемые названия параметров не обязаны совпадать с названиями в файле. Они могут использоваться в качестве вспомогательной информации для разработчика проекта. **Результат выполнения** – в указанный регистр будет записан результат успешного выполнения импорта или код ошибки: • 0 – импорт завершен без ошибок; • 1 – некорректное имя файла; • 2 – ошибка индекса файла; • 3 – указанный путь не существует; • 4 – ошибка открытия файла; • 5 – фактический формат файла не соответствует настроенному; • 7 – ошибка записи данных в slave-устройство; • 8 – USB-накопитель не подключен.
Запись в рецепт	Запись значений указанных регистров в файл рецепта (указывается адрес начального регистра, количество регистров зависит от настроек группы рецептов). Группа рецептов выбирается с помощью выпадающего списка или путём указания адреса регистра, в который будет записан номер группы. Рецепт выбирается путем изменения системного регистра **SPFW256** (**Индекс рецепта**). В случае установки галочки **Бит уведомления** – после завершения операции указанный бит устанавливается в значение **TRUE**
Чтение из рецепта	Чтение значений из файла рецепта в указанные регистры (указывается адрес начального регистра, количество регистров зависит от настроек группы рецептов). Группа рецептов выбирается с помощью выпадающего списка или путём указания адреса регистра, в который будет записан номер группы. Рецепт выбирается путем изменения системного регистра **SPFW256** (**Индекс рецепта**). В случае установки галочки **Бит уведомления** – после завершения операции указанный бит устанавливается в значение **TRUE**
Вызов макроса	Вызов указанного макроса. Информация о режиме выполнения (синхронно/асинхронно) приведена в пункте 12.6.10

В настройках каждого действия есть вкладка **Безопасность**.

10.6.8 Функциональная область

Элемент **Функциональная область** является невизуализируемым и используется для выполнения заданных действий при наступлении заданных событий. Один из типичных сценариев использования элемента – организация обмена со slave-устройством с помощью групповых запросов.

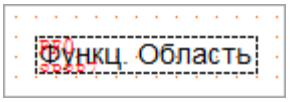


Рисунок 10.66 – Функциональная область

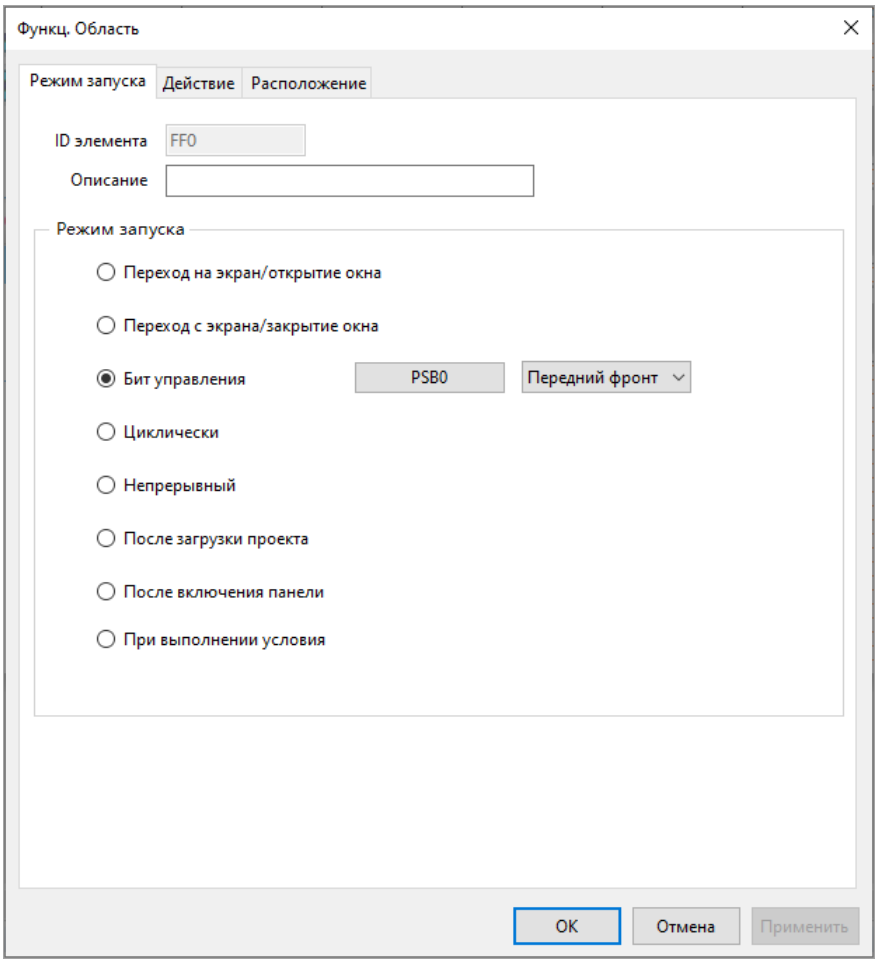
Для того чтобы элемент функционировал независимо от открытого в данный момент экрана, следует нажать на элемент **ПКМ** и использовать команду **Сделать глобальным**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице 10.28.

Доступные действия описаны в таблице 10.27.

Таблица 10.28 – Настройки элемента Функциональная область

Настройка	Описание
Вкладка **Режим запуска**  **Рисунок 10.67 – Вкладка Режим запуска**	
Переход на экран/ открытие окна	Настроенные действия будут однократно выполнены в момент перехода на экран или открытия окна, в котором размещена функциональная область

Продолжение таблицы 10.28

Настройка	Описание
Переход с экрана/ закрытие окна	Настроенные действия будут однократно выполнены в момент перехода с экрана или закрытия окна, в котором размещена функциональная область
Бит управления	Настроенные действия будут однократно выполнены по переднему или заднему фронту указанного бита
Циклически	Настроенные действия будут выполняться циклически с указанной периодичностью (размерность – 0.1 сек, значение 0 не поддерживается). В случае установки галочки **Бит управления** можно указать бит и его состояние, которое будет запускать функциональную область. При изменении бита на противоположное состояние – работа функциональной области будет останавливаться (в т. ч. остановится таймер отсчёта). После возвращения к заданному состоянию – работа функциональной области продолжится с момента остановки. В случае установки галочки **Вести отсчет времени** можно указать регистр, в который будет записываться текущее значение таймера функциональной области. Таймер обнуляется и начинает отсчёт заново с указанной периодичностью
Непрерывно	Настроенные действия выполняются с максимально возможной частотой. Это снижает производительность панели, поэтому данный режим следует использовать только в том случае, если такая частота выполнения обусловлена конкретными требованиями решаемой задачи. В случае установки галочки **Бит управления** можно указать бит и его состояние, которое будет запускать функциональную область. При изменении бита на противоположное состояние – работа функциональной области будет останавливаться (в т. ч. остановится таймер отсчёта). После возвращения к заданному состоянию – работа функциональной области продолжится с момента остановки
После загрузки проекта	Настроенные действия будут однократно выполнены после загрузки проекта из конфигуратора или с USB-накопителя
После запуска панели	Настроенные действия будут однократно выполнены после запуска или перезапуска панели. Перезапуск панели происходит при загрузке проекта, а также при установке системного бита **SPSB200**. Данное событие не обрабатывается в режиме эмуляции
При выполнении условия	Настроенные действия будут однократно выполняться при каждом срабатывании заданного условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному.

10.6.9 Выпадающий список

Элемент **Выпадающий список** используется для записи в регистр одного из предустановленных значений, каждому из которых соответствует текстовое описание.

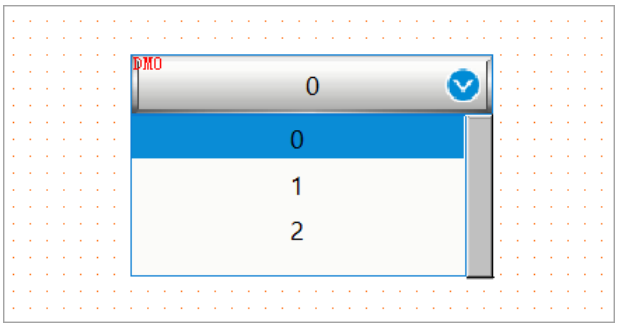
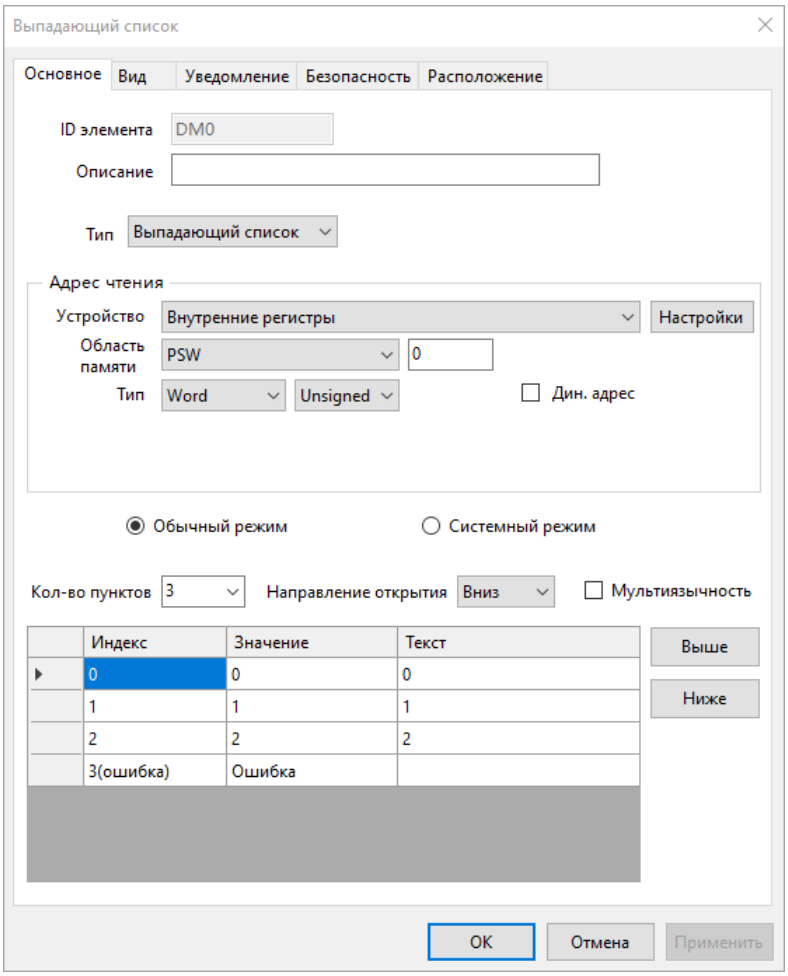


Рисунок 10.68 – Выпадающий список

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.29 – Настройки элемента Выпадающий список

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.69 – Вкладка Основное**	
Тип	Выпадающий список или панель. Панель представляет собой постоянно отображаемый список значений

Продолжение таблицы 10.29

Настройка	Описание
Режим	Обычный или системный. Системный режим доступен только для типа **Выпадающий список** и используется для создания списка пользователей панели в системных окнах 25001 и 25014; этот режим не должен применяться пользователем
Кол-во пунктов	Количество элементов списка
Направление открытия	Только для типа **Выпадающий список**. Направление открытия списка относительно элемента – вверх или вниз
Мультиязычность	В случае установки галочки вместо статических текстов для пунктов списка можно выбрать тексты из библиотеки локализации
Содержимое	• Индекс – не редактируется; • Значение регистра – значение, которое будет записано в регистр элемента при выборе данного пункта. Для редактирования значения следует два раза нажать **ЛКМ** на соответствующую ячейку; • Текст – текст, который будет отображаться в строке данного пункта. Для редактирования следует два раза нажать **ЛКМ** на соответствующую ячейку. Дополнительно к настроенным пользователем пунктам всегда присутствует пункт **Ошибка**, который будет отображаться в элементе, если значение его регистра не совпадает ни с одним из указанных в таблице (только для типа **Выпадающий список**)

Вкладка **Вид**

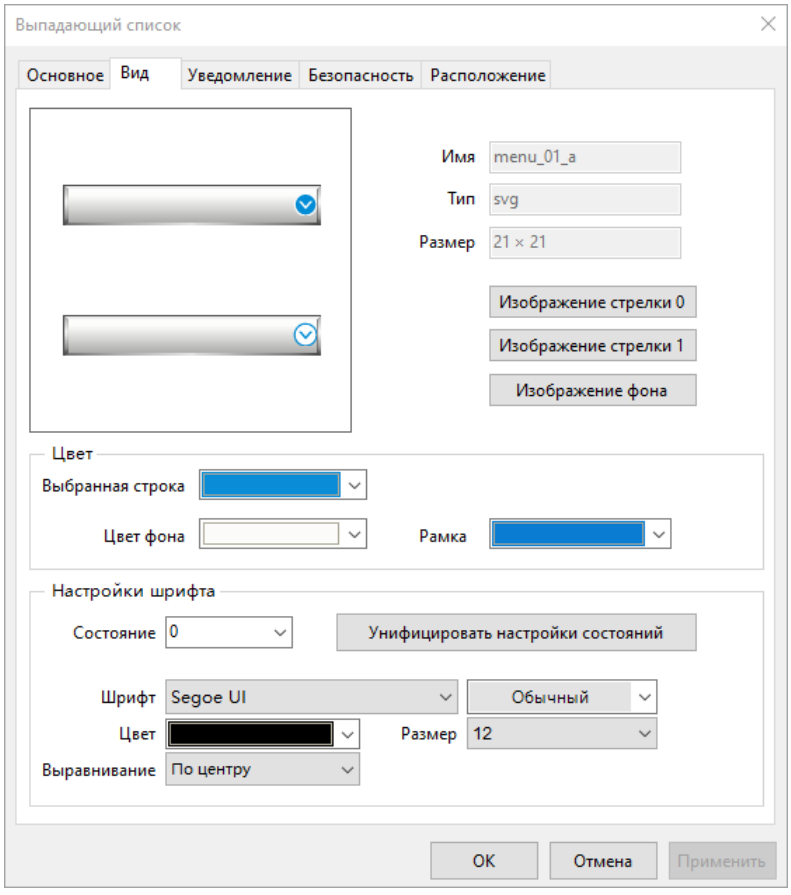


Рисунок 10.70 – Вкладка Вид

Изображение	Выбор изображений стрелок для обычного и раскрытого состояния списка (только для типа **Выпадающий список**) и фона элемента из **Библиотеки изображений** (папка **Menu**)

Продолжение таблицы 10.29

Настройка	Описание
Цвет	Цвет выбранного пункта списка, цвет фона и цвет рамки элемента (последние два цвета доступны только для типа **Выпадающий список**)
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста (для каждого пункта он может быть настроен индивидуально)
Унифицировать настройки состояний	В случае нажатия кнопки текущие настройки текста применяются ко всем состояниям элемента
Вкладка **Уведомление**	
Действие	Данные действия будут выполнены при переходе в состояние **Ошибка** (см. описание настройки **Содержимое**). Доступные действия: • запись значения **TRUE** или **FALSE** в выбранный бит; • запись заданного значения в выбранный регистр.

10.6.10 Проводник

Элемент **Проводник** используется совместно с элементом **Функциональная кнопка** с настроенным действием **Импорт CSV** для выбора файла, размещенного на USB-накопителе. В режиме эмуляции элемент отображает содержимое директории установки конфигуратора.

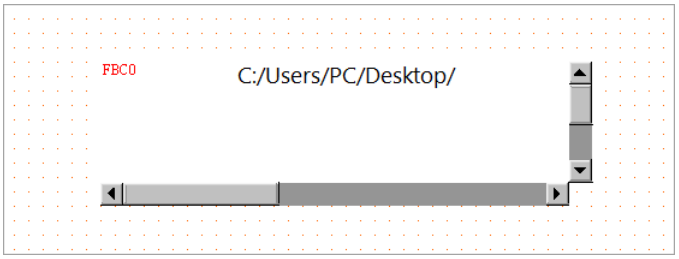
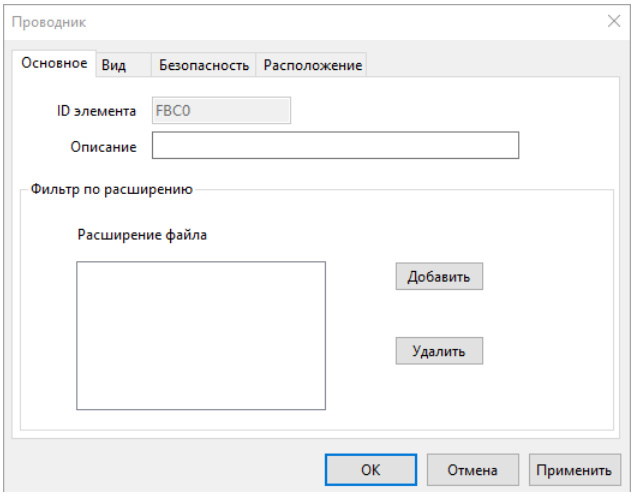


Рисунок 10.71 – Проводник

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.30 – Настройки элемента Проводник

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.72 – Вкладка Основное**	
Расширение файла	В данной таблице указываются расширения файлов, которые будут отображаться элементом (указывать точку в начале расширения необязательно)

ПРИМЕЧАНИЕ
Для каждого расширения рекомендуется создавать две записи в элементе – в нижнем и в верхнем регистре (например, csv и CSV). Это связано с различным способом обработки регистра символов в пути к файлу в ОС Windows (на которой запускается Конфигуратор СП210) и ОС Linux (которая используется в панели).

10.7 Приборы и инструменты

10.7.1 Время

Элемент **Время** используется для отображения текущего времени суток.

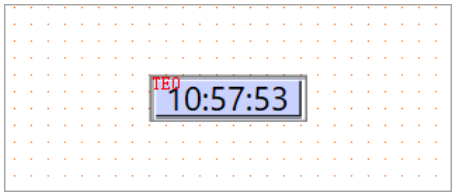
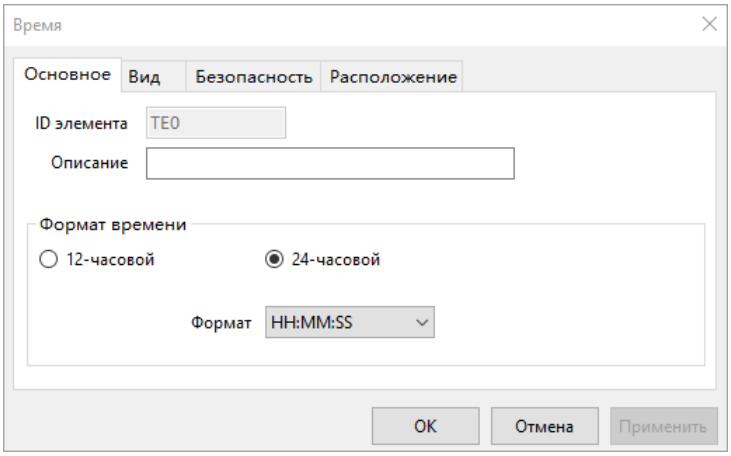


Рисунок 10.73 – Время

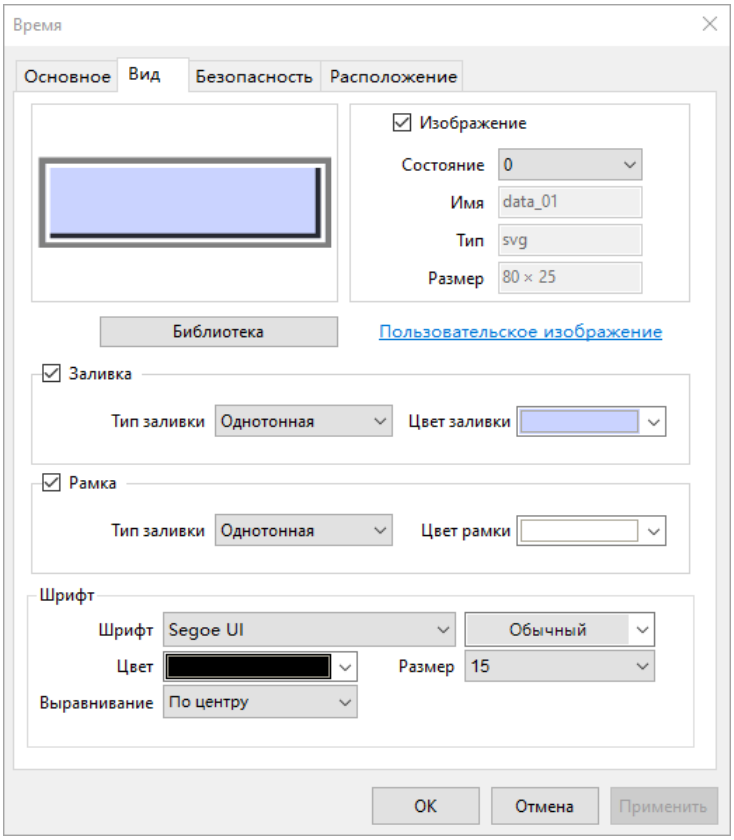
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.31 – Настройки элемента Время

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.74 – Вкладка Основное**	
Формат времени	12-часовой или 24-часовой
Формат отображения	Возможные варианты: • HH:MM:SS; • HH:MM; • HH:MM:SS:MS.

Продолжение таблицы 10.31

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.75 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.7.2 Дата

Элемент **Дата** используется для отображения текущей даты.

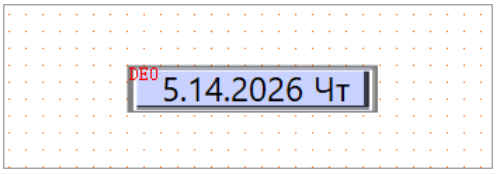
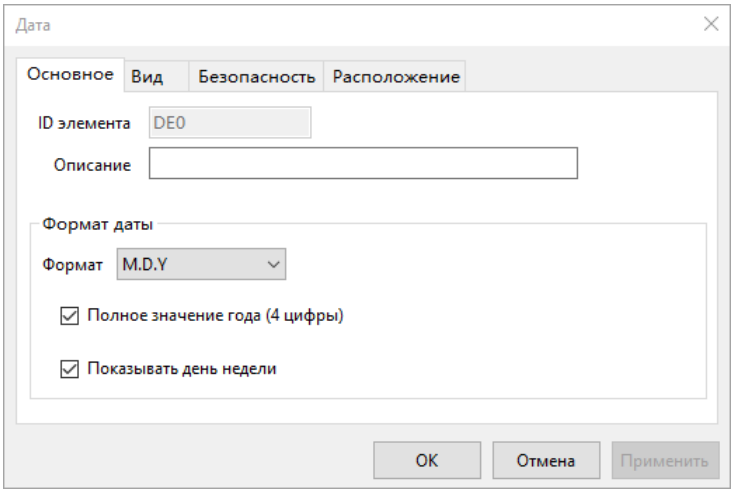


Рисунок 10.76 – Дата

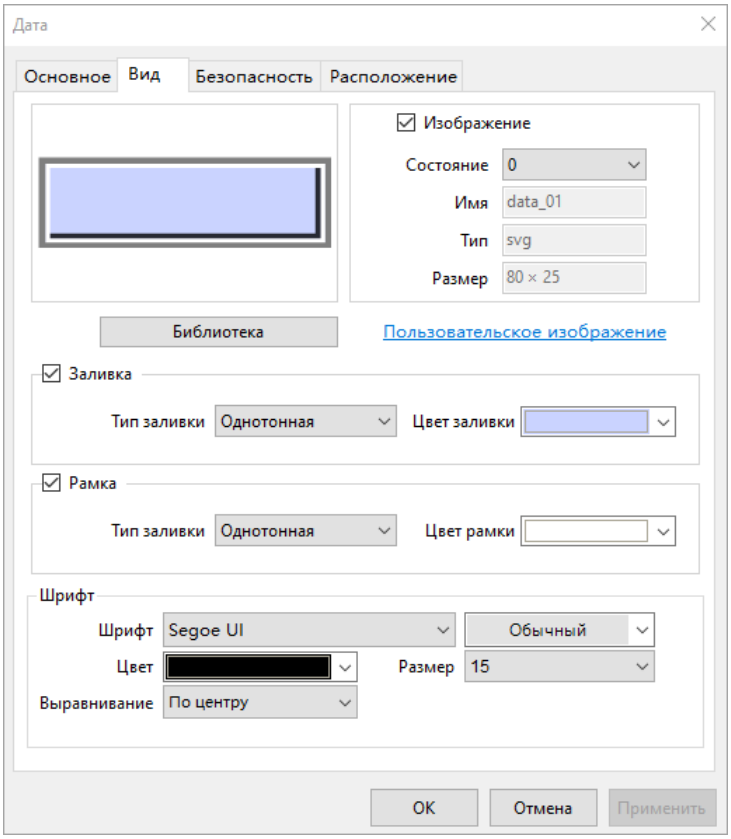
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.32 – Настройки элемента Дата

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.77 – Вкладка Основное**	
Формат даты	Возможные варианты: • M.D.Y; • D.M.Y; • Y.M.D.
Полное значение года (4 цифры)	В случае установки галочки отображаются все 4 цифры года, в случае отсутствия – только последние две
Показывать день недели	В случае установки галочки в элементе также отображается сокращенное обозначение дня недели

Продолжение таблицы 10.32

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.78 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.7.3 Трубопровод

Элемент **Трубопровод** используется для отображения трубы с протекающим по ней потоком вещества.

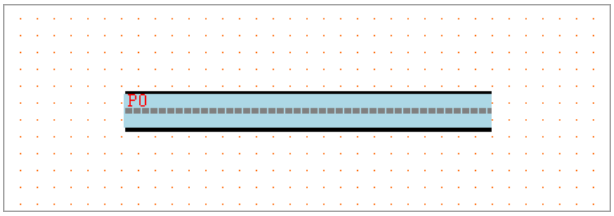
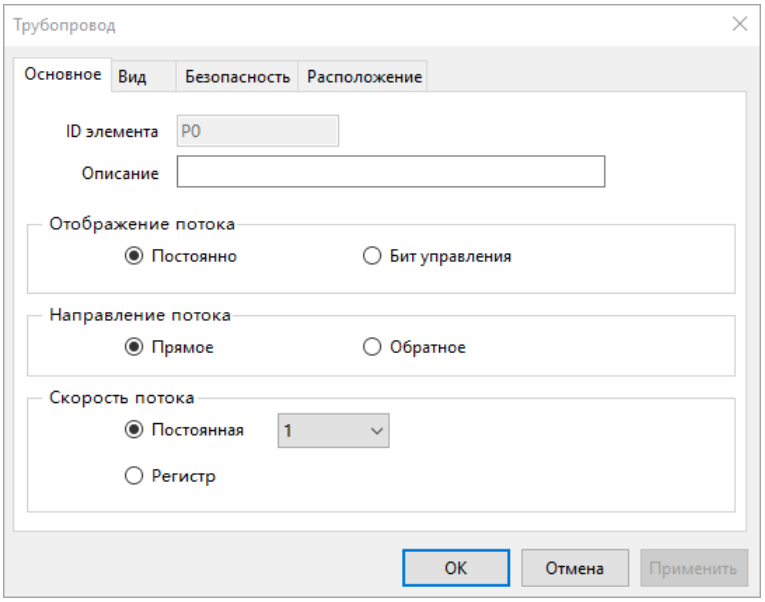
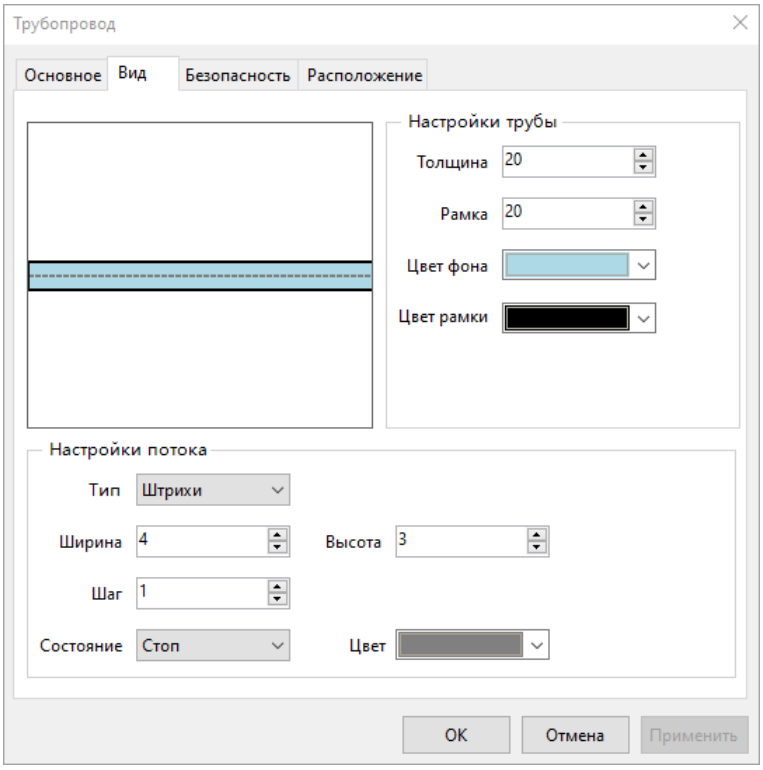


Рисунок 10.79 – Трубопровод

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.33 – Настройки элемента Трубопровод

Настройка	Описание
Вкладка **Основное**	
	
Рисунок 10.80 – Вкладка Основное	
Отображение потока	Отображение движения вещества в элементе: постоянно или только при заданном значении указанного бита
Направление потока	Прямое (сверху вниз, слева направо) или обратное (снизу вверх, справа налево)
Скорость потока	Задаётся константой или путём указания адреса регистра. Диапазон значений: 1…25, где 1 – самая медленная скорость, а 25 – самая быстрая
Вкладка **Вид**	
	
Рисунок 10.81 – Вкладка Вид	

Продолжение таблицы 10.33

Настройка	Описание
Настройки трубы	Размеры толщины и рамки трубы, цвет её фона и рамки
Настройки потока	Пиктограмма потока – штрихи или стрелки. Для пиктограммы настраиваются высота, ширина, шаг между элементами и цвет в активном/остановленном состоянии

10.7.4 Аналоговый дисплей

Элемент **Аналоговый дисплей** используется для отображения числового значения в виде стрелочного индикатора.

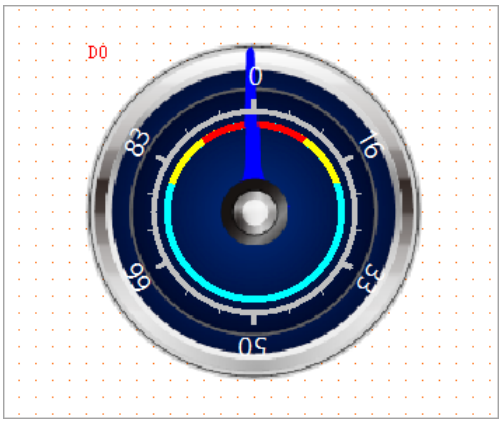
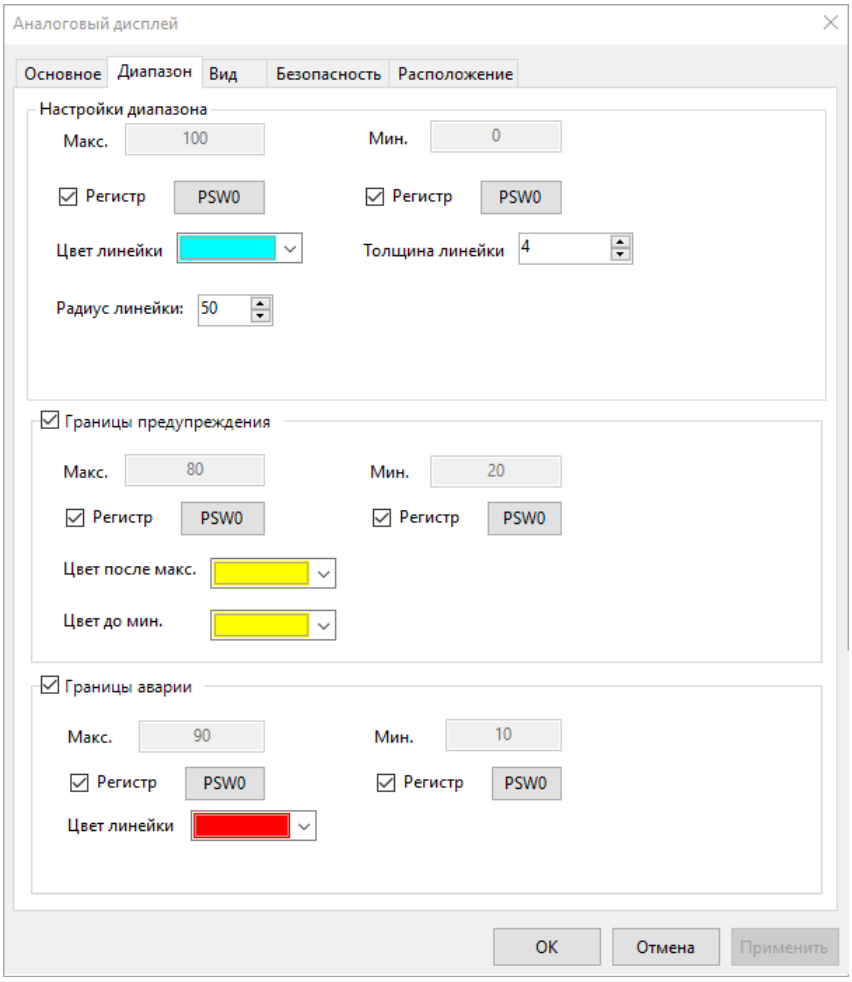


Рисунок 10.82 – Аналоговый дисплей

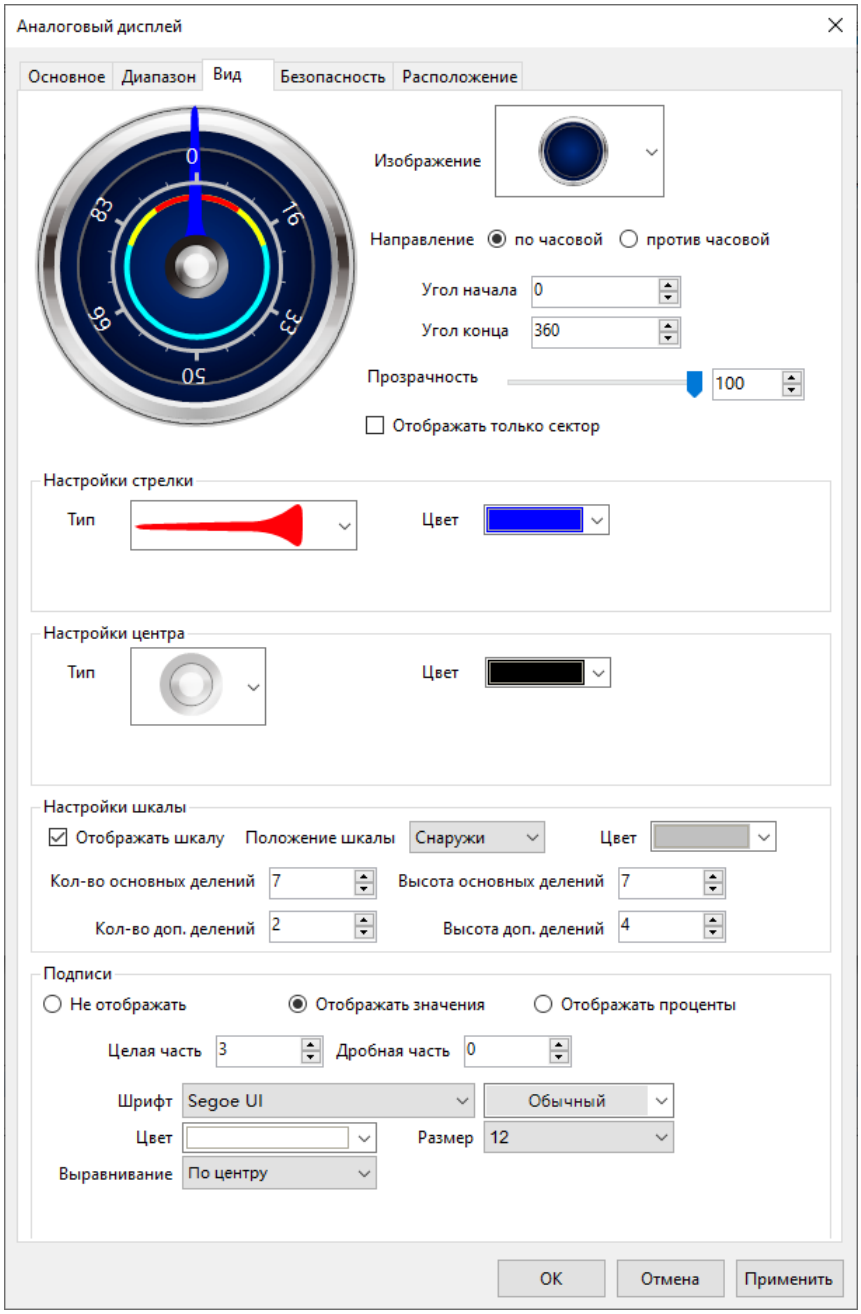
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.34 – Настройки элемента Аналоговый дисплей

Настройка	Описание
Вкладка **Диапазон**  **Рисунок 10.83 – Вкладка Диапазон**	
Диапазон	• Диапазон значений отображаемого элементом регистра (в интервале 0…999). Границы можно задать через константы или путём указания адресов регистров. • Цвет линейки в нормальном состоянии. • Толщина и радиус линейки
Границы предупреждения	Границы предупредительного состояния. Области линейки, выходящие за эти границы, отображаются указанными цветами. Значение внутри этих границ отображается цветом нормального состояния
Границы аварии	Границы аварийного состояния (должны быть «шире», чем границы предупредительного состояния). Область линейки, выходящая за эти границы, отображается указанным цветом

Продолжение таблицы 10.34

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.84 – Вкладка Вид**	
Изображение	Выбор изображения фона элемента из Библиотеки изображений (папка **Instrument**)
Направление	Направление поворота стрелки элемента – по часовой или против часовой
Угол	Угол начала и конца линейки и шкалы элемента
Прозрачность	Степень прозрачности элемента
Отображать только сектор	В случае установки галочки отображается только сектор элемента, ограниченный заданными углами (только в случае выбора кругового фонового изображения)
Настройки стрелки	Выбор изображения стрелки элемента из Библиотеки изображений (папка **Instrument**) и цвета стрелки

Продолжение таблицы 10.34

Настройка	Описание
Настройки центра	Выбор изображения центральной точки элемента из Библиотеки изображений (папка **Instrument**) и цвета точки
Настройки шкалы	Настройки отображения шкалы элемента: • Положение – снаружи, внутри или в центре; • Количество и высота основных делений; • Количество и высота дополнительных делений.
Подписи	Выбор режима отображения подписей: • не отображать; • отображать значения; • отображать проценты. Для подписей настраивается: • максимальное количество знаков целой и дробной части; • шрифт, стиль шрифта, цвет и размер текста.

10.7.5 Двигатель

Элемент **Двигатель** используется для отображения соответствующего технологического объекта.

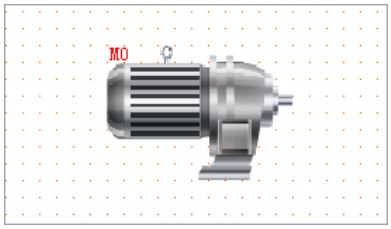
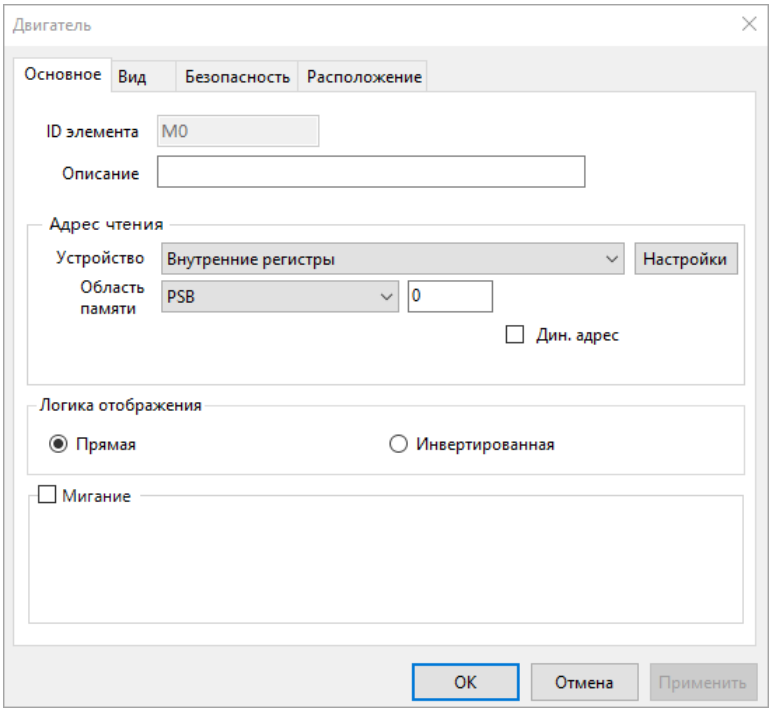
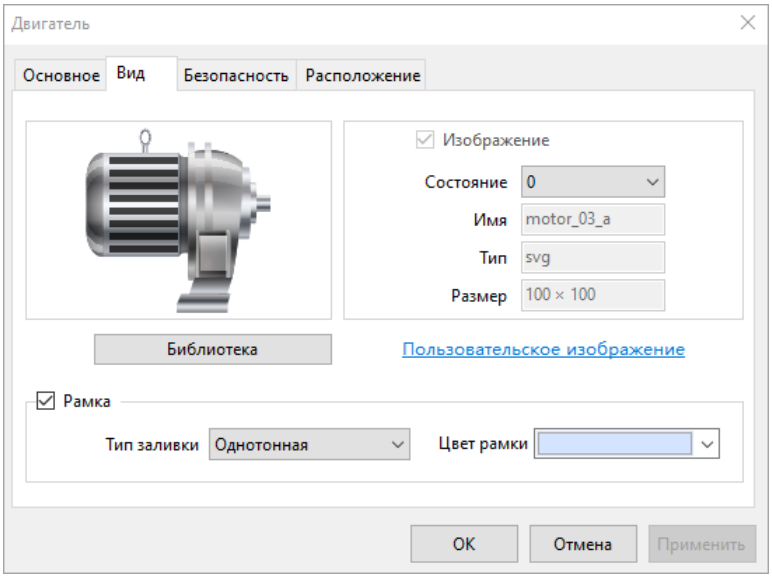


Рисунок 10.85 – Двигатель

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.35 – Настройки элемента Двигатель

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.86 – Вкладка Основное**	
Логика отображения	Возможные варианты: • Прямая – включенному состоянию элемента соответствует значение бита **TRUE**, отключенному – **FALSE**; • Инвертированная – включенному состоянию элемента соответствует значение бита **FALSE**, отключенному – **TRUE**.
Мигание	В случае установки галочки элемент будет мигать с заданным периодом, когда его бит перейдёт в заданное состояние. Диапазон периода мигания: 0.1…10.0 секунд (с дискретностью в 0.1 секунды)
Вкладка **Вид**  **Рисунок 10.87 – Вкладка Вид**	

Продолжение таблицы 10.35

Настройка	Описание
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

10.7.6 Емкость

Элемент **Емкость** используется для отображения соответствующего технологического объекта и степени заполнения его веществом, которая определяется значением привязанного регистра.

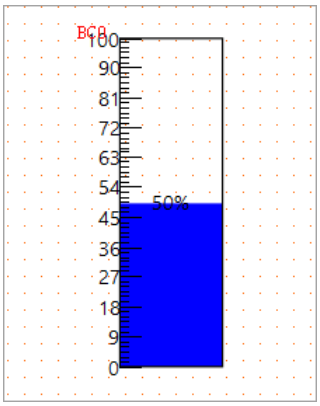
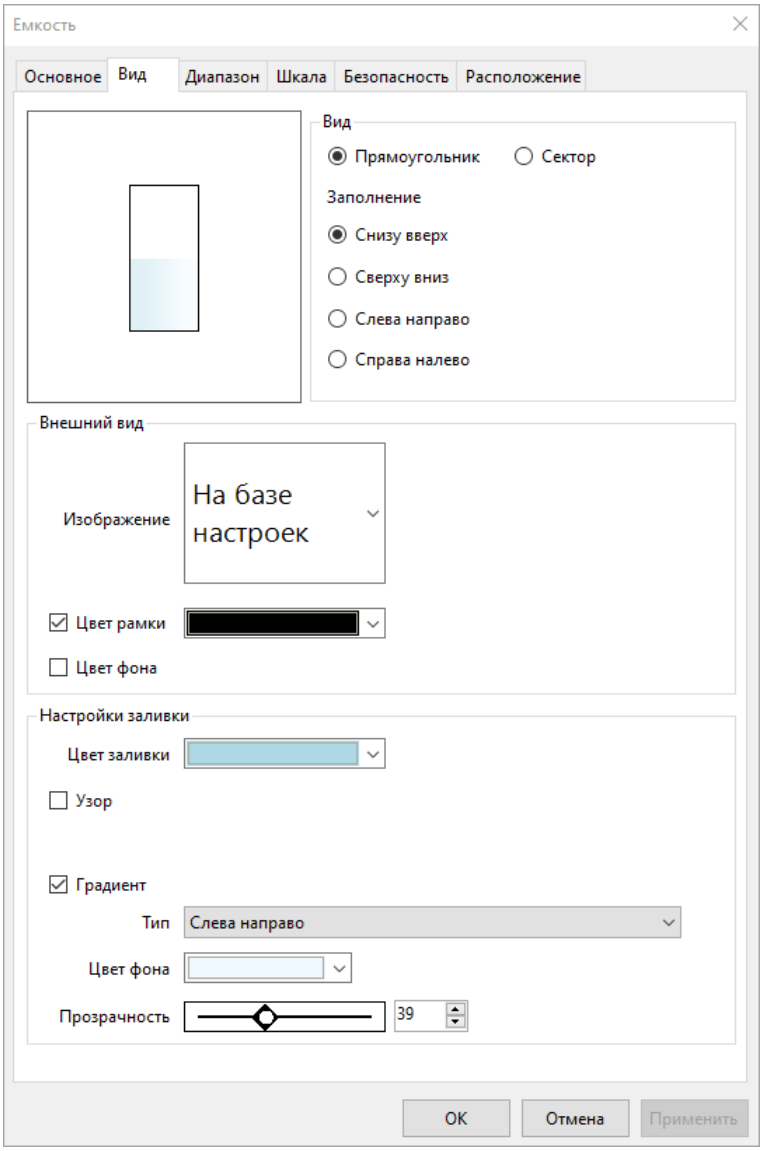


Рисунок 10.88 – Емкость

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

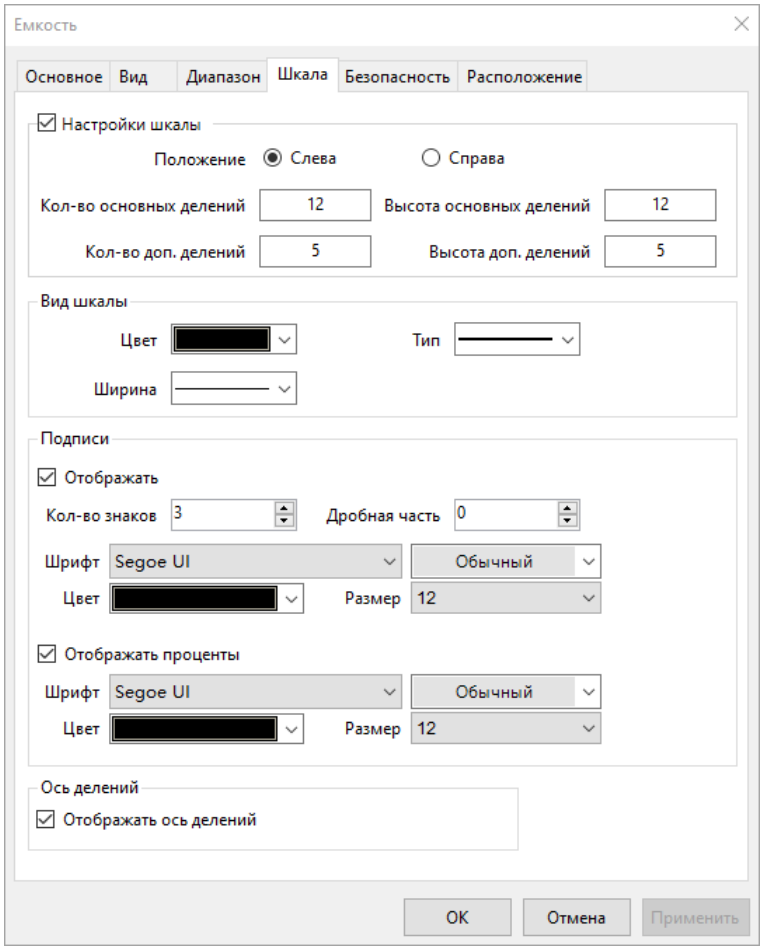
Таблица 10.36 – Настройки элемента Емкость

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.89 – Вкладка Вид**	
Вид	Тип элемента – прямоугольник или сектор
Заполнение	Настройка доступна только для типа **Прямоугольник**. Направление заполнения элемента: • Снизу вверх; • Сверху вниз; • Слева направо; • Справа налево.
Настройки сектора	Настройки доступны только для типа **Сектор**. • Процент заполнения – процент области элемента, на которую не будет распространяться заполнение (1…99); • Углы начала и конца элемента; • Направление заполнения – по часовой или против часовой стрелки.
Внешний вид	Выбор изображения элемента из Библиотеки изображений(папка **StickFigure**). В случае выбора варианта **На базе настроек** – можно указать цвет рамки и фона элемента

Продолжение таблицы 10.36

Настройка	Описание
Настройки заливки	Тип, цвет и степень прозрачности заливки элемента при заполнении. Поддерживаемые типы: • однотонная заливка; • двухцветный градиент; • узор.
Вкладка **Диапазон**  **Рисунок 10.90 – Вкладка Диапазон**	
Настройки диапазона	Диапазон значений отображаемого элементом регистра (в интервале 0…999). Границы можно задать через константы или путём указания адресов регистров
Целевой диапазон	Диапазон значений регистра, соответствующий нормальному состоянию. Область заполнения в этом диапазоне отображается указанным цветом. Задается в виде целевого значения и области вокруг него. Пример: • Целевое значение = 50; • Диапазон = 10. Итоговый диапазон нормального состояния: 40…60.
Границы аварии	Границы аварийного состояния (должны быть «шире», чем границы нормального состояния). При выходе значения регистра элемента за эти границы область заполнения отображается указанным цветом

Продолжение таблицы 10.36

Настройка	Описание
Вкладка **Шкала**  **Рисунок 10.91 – Вкладка Шкала**	
Настройки шкалы	Доступные настройки: • Положение шкалы – набор доступных вариантов зависит от параметров **Вид** и **Заполнение**; • Количество и высота основных делений шкалы; • Количество и высота дополнительных делений шкалы.
Вид шкалы	Цвет, тип и толщина шкалы
Отображать	В случае установки галочки рядом с делениями отображаются соответствующие им абсолютные значения. Для них можно задать количество знаков целой и дробной части, шрифт, стиль, цвет и размер текста
Отображать проценты	Данная галочка может быть установлена только при отсутствии галочки **Отображать значения**. В случае установки галочки рядом с делениями отображаются соответствующие им относительные значения в процентах. Для них можно задать шрифт, стиль, цвет и размер текста
Ось делений	В случае установки галочки отображается ось делений

10.7.7 Зуммер

Элемент **Зуммер** является невизуализируемым и используется для воспроизведения звукового сигнала. Для работы элемента в меню **Настройки проекта – Общие – Звук** должна быть снята галочка **Отключение зуммера**.

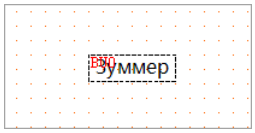


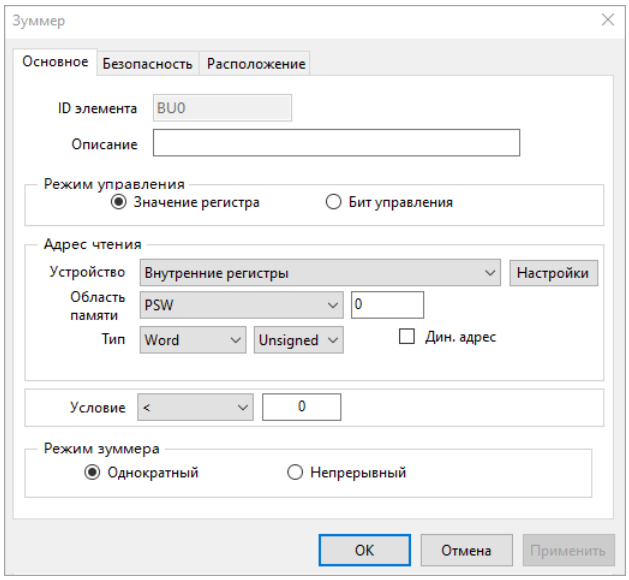
Рисунок 10.92 – Зуммер

Для того чтобы элемент функционировал независимо от открытого в данный момент экрана, следует нажать на элементе **ПКМ** и использовать команду **Сделать глобальным**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.37 – Настройки элемента Зуммер

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.93 – Вкладка Основное**	
Режим управления	Тип данных, используемых для управления элементом – регистр или бит
Условие	Для режима **Значение регистра** доступны следующие условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному. Для режима **Бит управления** выбирается состояние бита – **TRUE** или **FALSE**
Режим зуммера	• Однократный – звуковой сигнал кратковременно будет воспроизведён в момент выполнения заданного условия; • Непрерывный – звуковой сигнал будет воспроизводиться непрерывно, пока условие выполняется.

10.7.8 Подсветка

Элемент **Подсветка** является невизуализируемым и используется для перевода панели в режим отключения подсветки и выхода из этого режима.

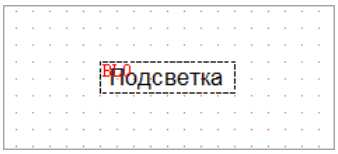


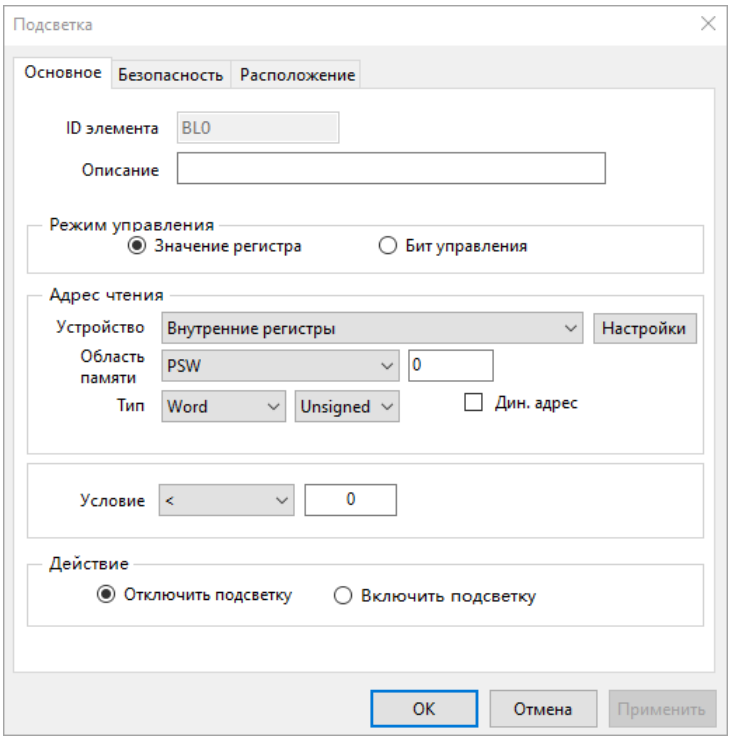
Рисунок 10.94 – Подсветка

Для того чтобы элемент функционировал независимо от открытого в данный момент экрана, следует нажать на нём **ПКМ** и использовать команду **Сделать глобальным**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.38 – Настройки элемента Управление подстветкой

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.95 – Вкладка Основное**	
Режим управления	Тип данных, используемых для управления элементом – регистр или бит
Условие	Для режима **Значение регистра** доступны следующие условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному. Для режима **Бит управления** выбирается состояние бита – **TRUE** или **FALSE**
Действие	Отключение подсветки или включение подсветки

10.7.9 Вентилятор

Элемент **Вентилятор** используется для отображения соответствующего технологического объекта.

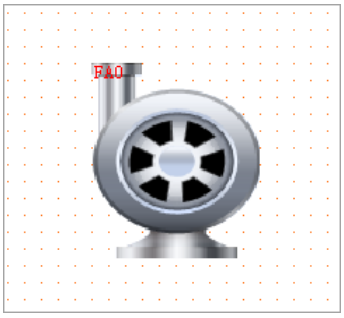
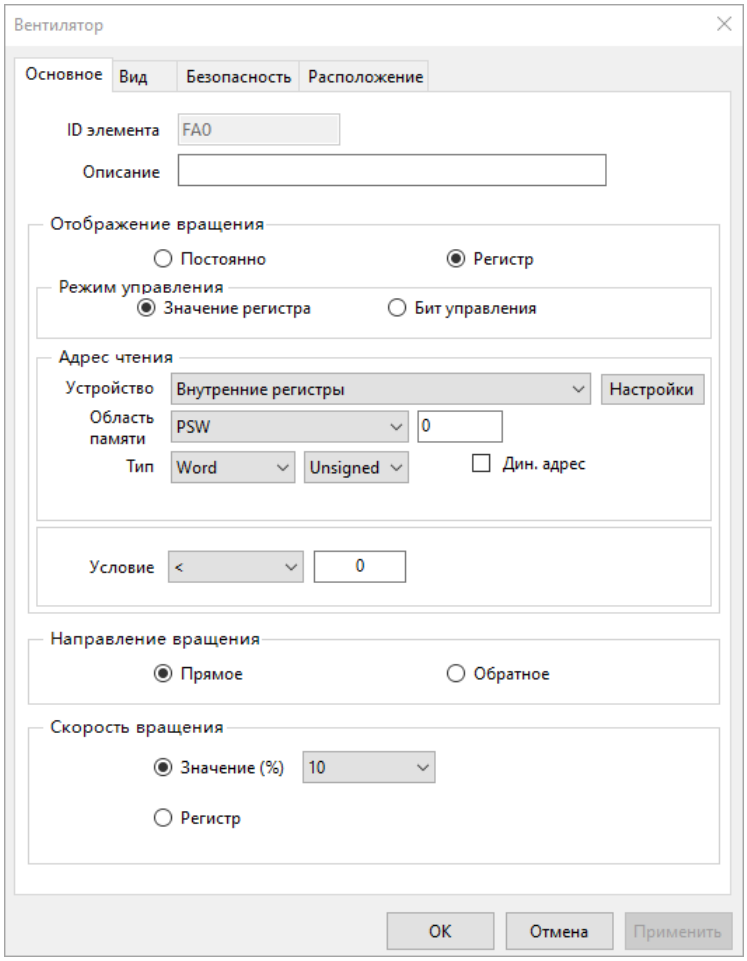


Рисунок 10.96 – Вентилятор

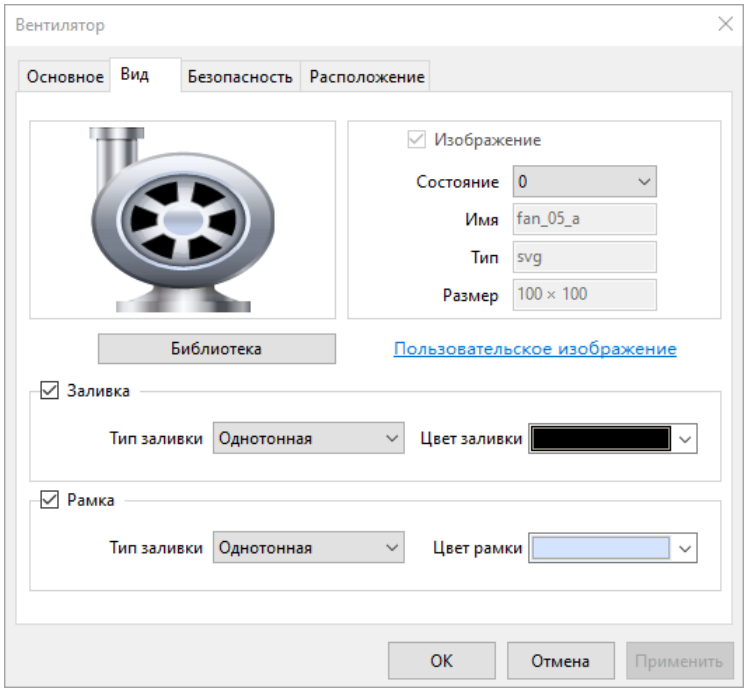
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.39 – Настройки элемента Вентилятор

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.97 – Вкладка Основное**	
Отображение вращения	Отображение вращения элемента: постоянно или только при выполнении заданного условия
Режим управления	Тип данных, используемых для управления элементом – регистр или бит

Продолжение таблицы 10.39

Настройка	Описание
Условие	Для режима **Значение регистра** доступны следующие условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному. Для режима **Бит управления** выбирается состояние бита – **TRUE** или **FALSE**
Направление вращения	Прямое (по часовой стрелке) или обратное (против часовой)
Скорость вращения	Задаётся константой или путём указания адреса регистра. Диапазон значений: 10…100, где 10 – самая медленная скорость, а 100 – самая быстрая
Вкладка **Вид**  **Рисунок 10.98 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

10.7.10 Перемешиватель

Элемент **Перемешиватель** используется для отображения соответствующего технологического объекта.

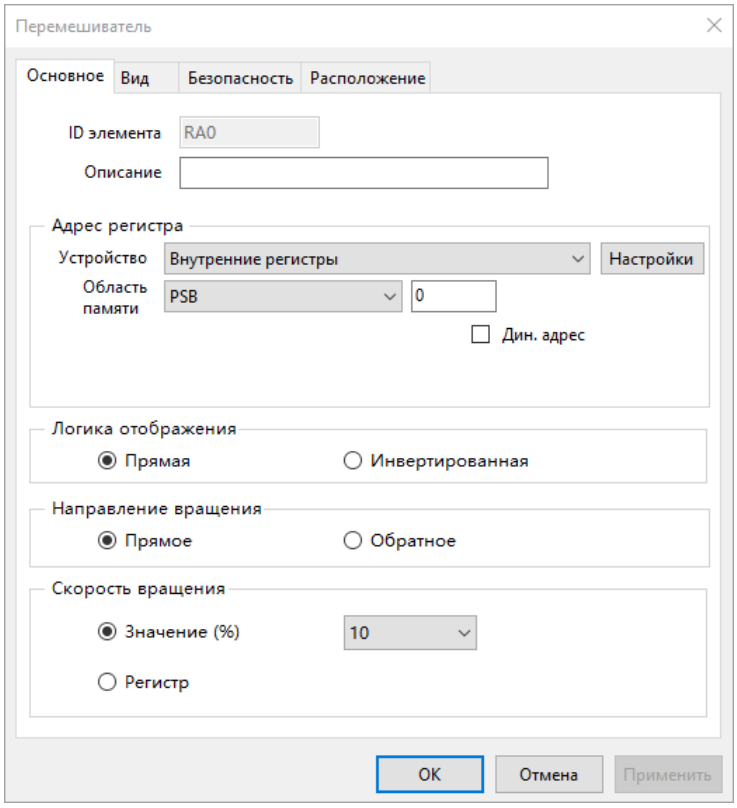


Рисунок 10.99 – Перемешиватель

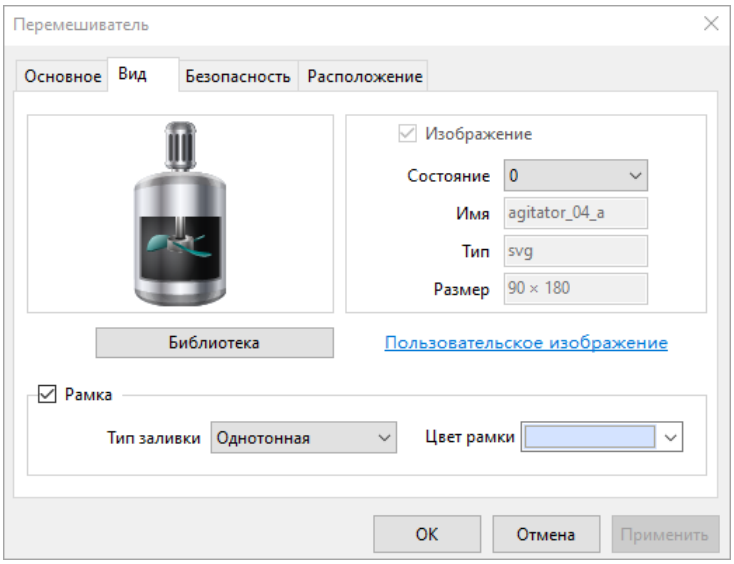
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.40 – Настройки элемента Перемешиватель

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.100 – Вкладка Основное**	
Логика отображения	Возможные варианты: • Прямая – включенному состоянию элемента соответствует значение бита **TRUE**, отключенному – **FALSE**; • Инвертированная – включенному состоянию элемента соответствует значение бита **FALSE**, отключенному – **TRUE**.

Продолжение таблицы 10.40

Настройка	Описание
Направление вращения	Прямое (по часовой стрелке) или обратное (против часовой)
Скорость вращения	Задаётся константой или путём указания адреса регистра. Диапазон значений: 10…100, где 10 – самая медленная скорость, а 100 – самая быстрая
Вкладка **Вид**  **Рисунок 10.101 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

10.7.11 Насос

Элемент **Насос** используется для отображения соответствующего технологического объекта.

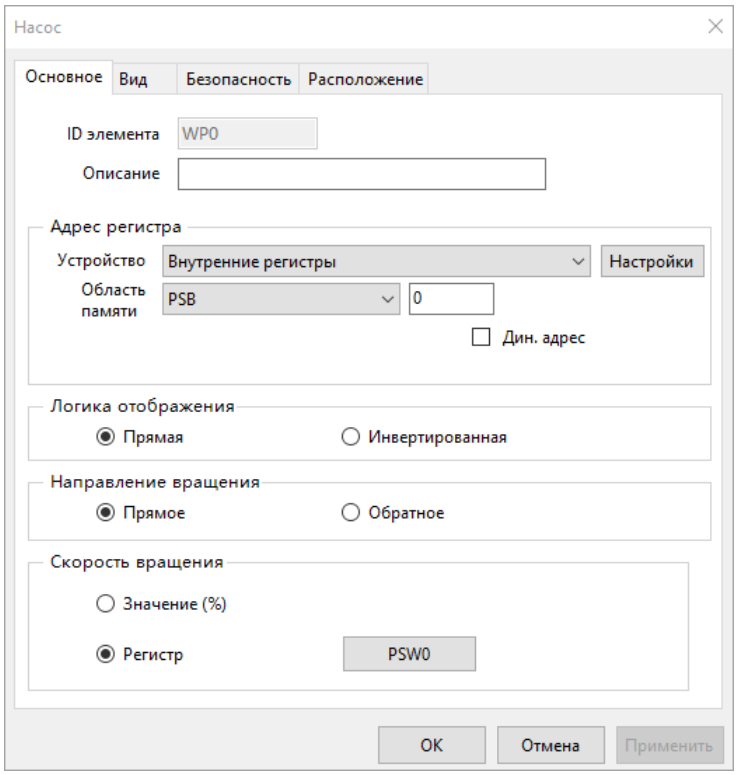


Рисунок 10.102 – Насос

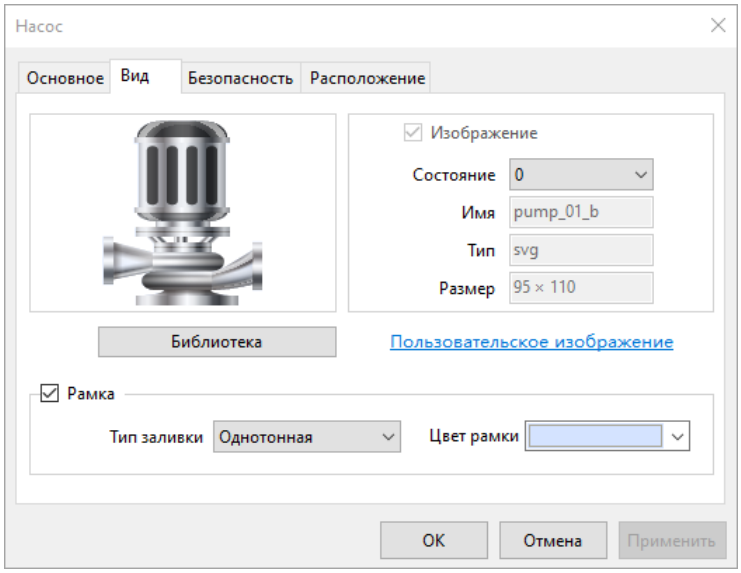
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.41 – Настройки элемента Насос

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.103 – Вкладка Основное**	
Логика отображения	Возможные варианты: • Прямая – включенному состоянию элемента соответствует значение бита **TRUE**, отключенному – **FALSE**; • Инвертированная – включенному состоянию элемента соответствует значение бита **FALSE**, отключенному – **TRUE**.
Направление вращения	Прямое (по часовой стрелке) или обратное (против часовой)
Скорость вращения	Задаётся константой или путём указания адреса регистра. Диапазон значений: 10…100, где 10 – самая медленная скорость, а 100 – самая быстрая

Продолжение таблицы 10.41

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.104 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

10.7.12 Клапан

Элемент **Клапан** используется для отображения соответствующего технологического объекта.

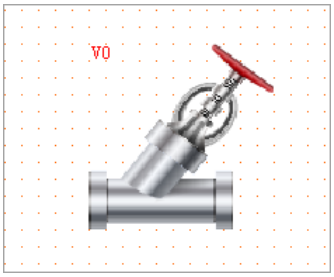
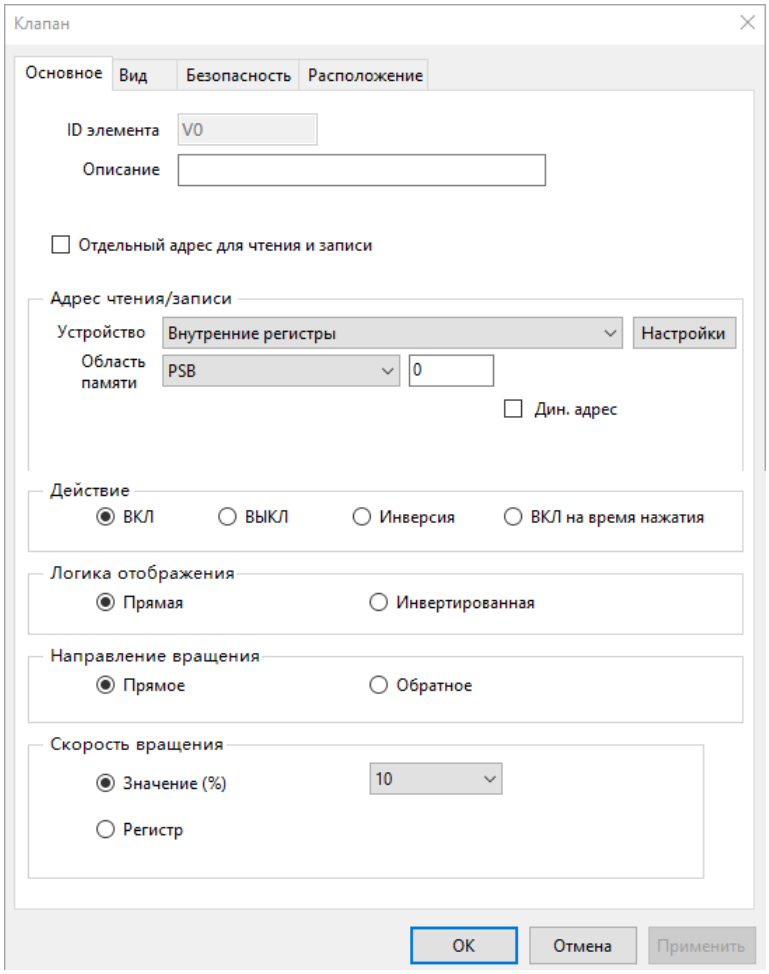


Рисунок 10.105 – Клапан

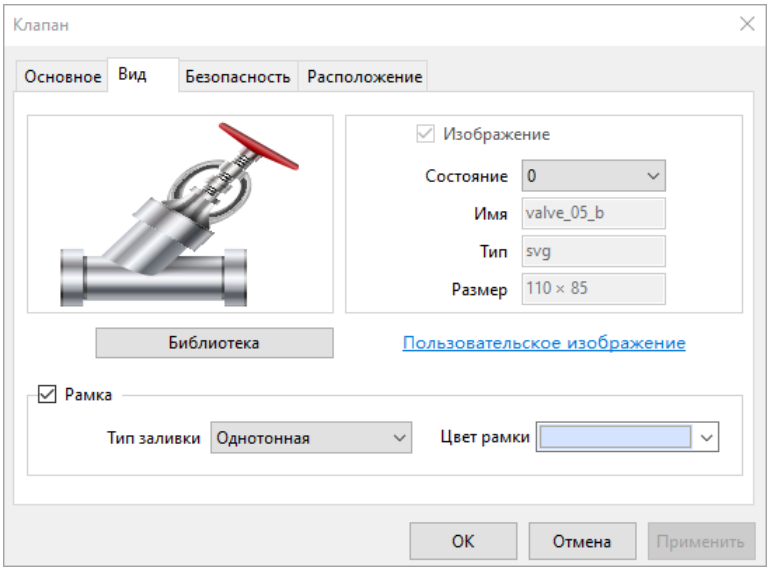
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.42 – Настройки элемента Клапан

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.106 – Вкладка Основное**	
Отдельный адрес для чтения и записи	В случае установки галочки элемент будет отображать значение одного бита, а изменять значение другого
Действие	Действие с битом, выполняемое при нажатии на элемент. Возможные варианты: • ВКЛ – переключение бита в значение **TRUE**; • ВЫКЛ – переключение бита в значение **FALSE**; • Инверсия – изменение значения бита на противоположное; • ВКЛ на время нажатия – установка бита в значение **TRUE** на время удержания элемента. После его отпускания бит будет переключен в значение **FALSE**.
Логика отображения	Возможные варианты: • Прямая – включенному состоянию элемента соответствует значение бита **TRUE**, отключенному – **FALSE**; • Инвертированная – включенному состоянию элемента соответствует значение бита **FALSE**, отключенному – **TRUE**.
Направление вращения	Прямое (по часовой стрелке) или обратное (против часовой)
Скорость вращения	Задаётся константой или путём указания адреса регистра. Диапазон значений: 10…100, где 10 – самая медленная скорость, а 100 – самая быстрая

Продолжение таблицы 10.42

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.107 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Рамка	Тип (однотонная или градиент) и цвет заливки контура элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр

10.7.13 Таймер

Элемент **Таймер** является невизуализируемым и используется для изменения значения указанного бита спустя заданное время после возникновения настроенного события.

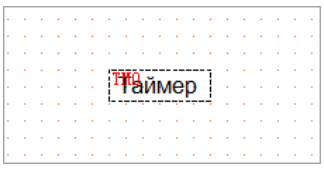


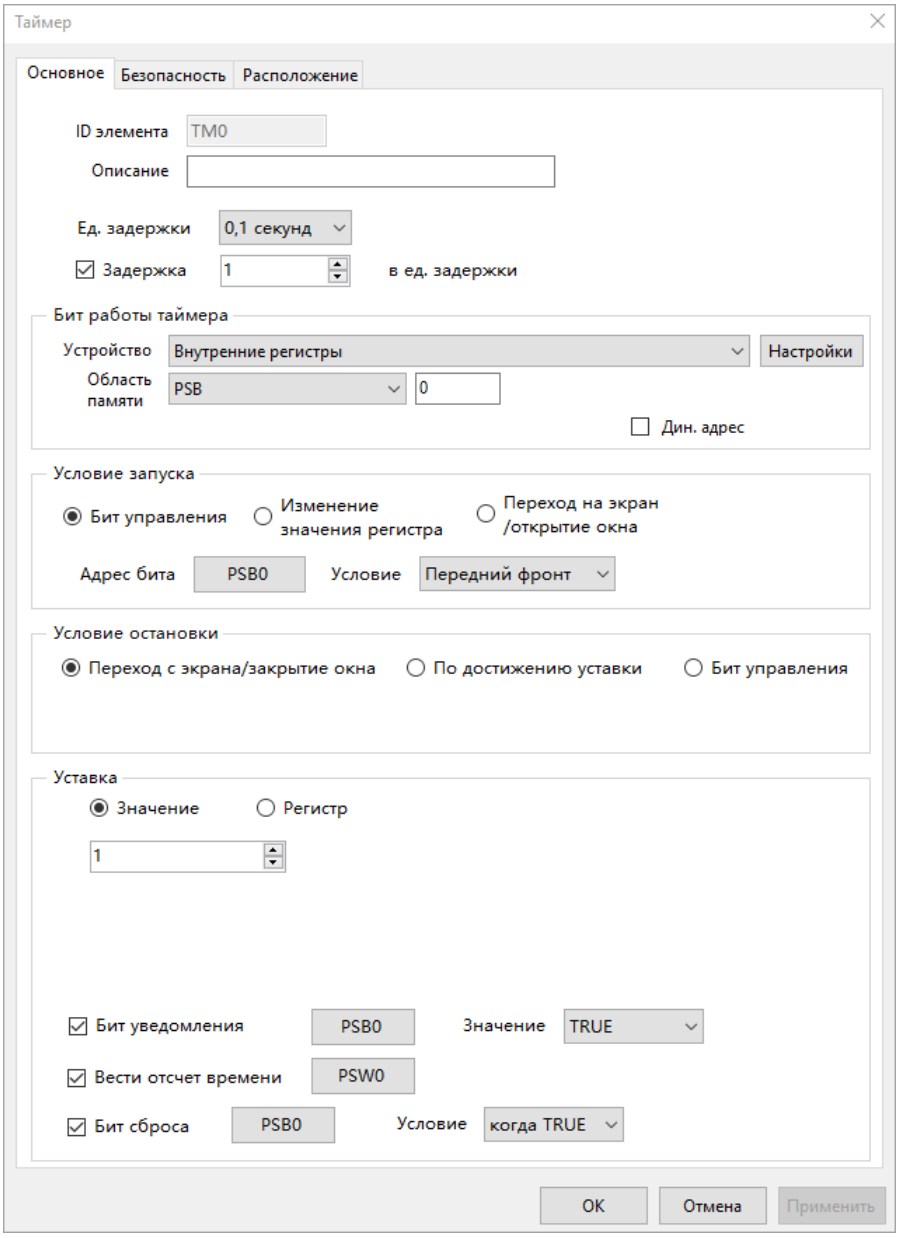
Рисунок 10.108 – Таймер

Для того чтобы элемент функционировал независимо от открытого в данный момент экрана, следует нажать на нём **ПКМ** и использовать команду **Сделать глобальным**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.43 – Настройки элемента Таймер

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.109 – Вкладка Основное**	
Единицы времени	Единицы времени, используемые для параметров **Задержка**, **Уставка** и **Вести отсчет времени**: • Десятые доли секунд; • Секунды; • Минуты.
Задержка	В случае установки галочки – можно указать интервал задержки между срабатыванием условия запуска и началом отсчёта таймера
Бит работы таймера	Бит, который имеет значение **TRUE** в процессе работы таймера

Продолжение таблицы 10.43

Настройка	Описание
Условие запуска таймера	Возможные варианты: • по переднему или заднему фронту указанного бита; • по изменению значения указанного регистра или, в случае установки галочки **Когда равно** – при его равенстве указанному значению; • при переходе на экран/окно, на котором размещен данный элемент. Если таймер находится в процессе работы, то повторная активация условия не приведет к его перезапуску
Условие остановки таймера	Возможные варианты: • при переходе с экрана или закрытия окна, в котором размещен данный элемент; • по достижению уставки; • по переднему или заднему фронту указанного бита.
Уставка	Интервал времени (в заданных единицах), в течение которого работает таймер (диапазон: 1…864000). В течение этого времени бит работы таймера имеет значение **TRUE**. Задается константой или путем указания адреса регистра
Бит уведомления	В случае установки галочки – указанный бит принимает заданное значение при остановке таймера
Вести отсчет времени	В случае установки галочки – в указанный регистр записывается текущее время работы таймера (в заданных единицах времени)
Бит сброса	В случае установки галочки – если заданный бит принимает указанное значение, то таймер прекращает свою работу и не может быть перезапущен до переключения бита в противоположное состояние

10.8 Тревоги

10.8.1 Настройка тревог

Данное меню используется для настройки тревог проекта, которые можно отобразить с помощью элементов **Таблица тревог** и **Баннер тревог**. История тревог сохраняется в файл.

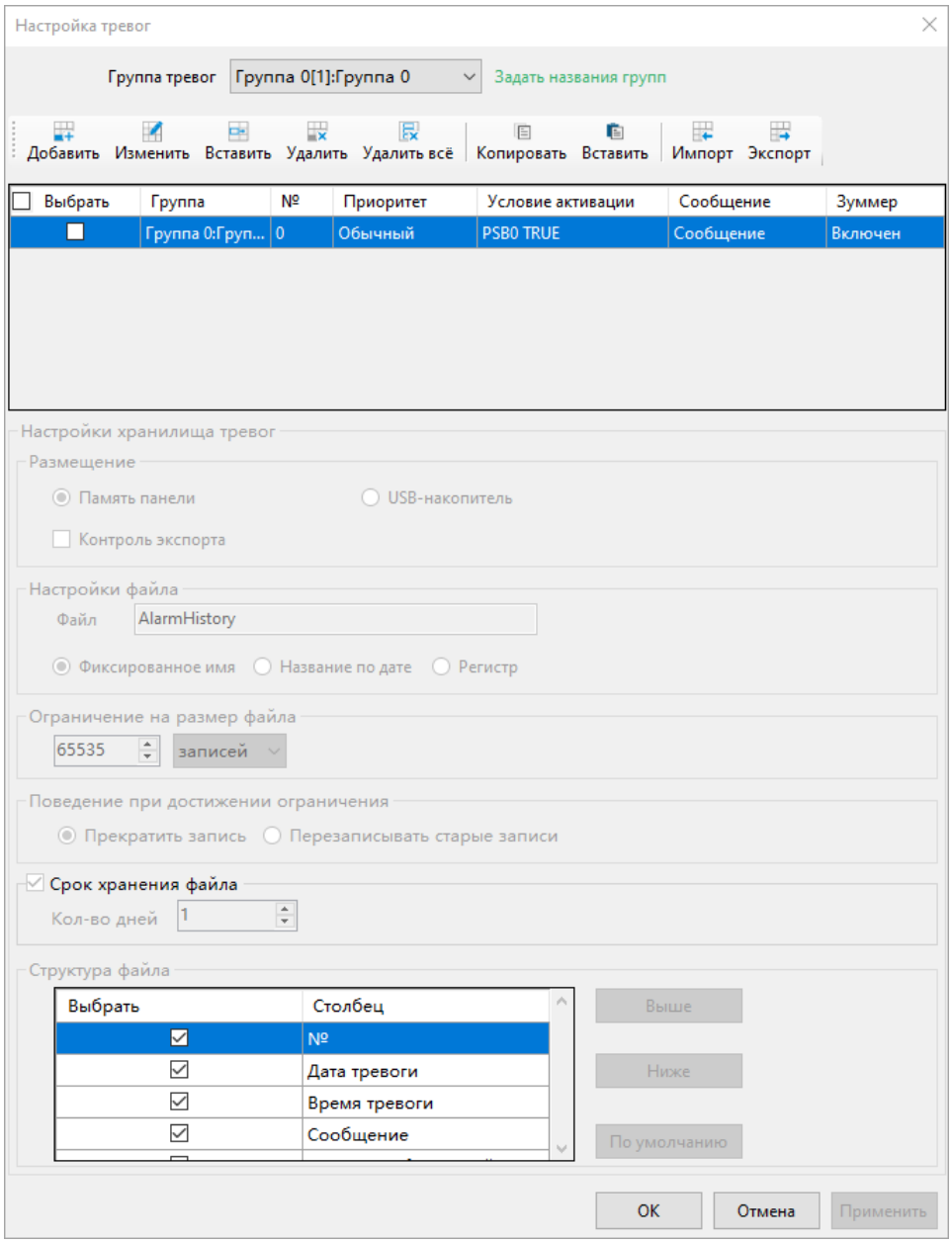


Рисунок 10.110 – Настройка тревог

Для создания новой тревоги следует выбрать группу тревог. Для изменения названий групп тревог следует нажать кнопку **Задать названия групп**.

Команда **Экспорт** позволяет выгрузить настройки тревог в виде файла .xlsx, .xls или .csv. Команда **Импорт** позволяет загрузить этот файл.

По переднему фронту системного бита **SPSB120** выполняется очистка истории тревог.

Настройки хранилища тревог описаны в таблице 10.44.

Настройки тревог описаны в таблице 10.45.

Таблица 10.44 – Настройки хранилища тревог

Настройка	Описание
Настройки хранилища тревог	Выбор места сохранения файла истории тревог: в памяти панели или на подключенном USB-накопителе (рекомендуется по возможности использовать второй вариант, чтобы снизить износ встроенной памяти панели). В случае сохранения файла в памяти панели доступна галочка **Контроль экспорта**, которая позволяет указать адрес первого из 3 регистров, в которые будет записываться информация экспорта. Предположим, указан регистр **PSW0**, тогда: **PSW0** – команда экспорта. При записи значения **1** происходит экспорт истории тревог на USB-накопитель. При записи значения **2** помимо экспорта выполняется удаление файла из памяти панели. Для выполнения новой команды в регистр предварительно должно быть записано значение **0**. **PSW1** – в этот регистр записывается значение прогресса операции экспорта в процентах. Значение **100** соответствует завершению экспорта. **PSW2** – в указанный регистр будет записан результат успешного выполнения экспорта или код ошибки: • 0 – экспорт в процессе; • 1 – экспорт завершен успешно; • 2 – USB-накопитель не подключен; • 3 – недостаточно памяти; • 4 – некорректный путь к файлу; • 5 – ошибка формата файла; • 6 – в процессе экспорта произошла ошибка. В режиме эмуляции: • для варианта **USB**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\alarm** с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf; • для варианта **Память панели**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\db\alarm** в бинарном формате. Для возможности его просмотра следует использовать команду экспорта (см. выше), которая сформирует в директории **\Temp\Run \storage\udisk\alarm** файл с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf.
Настройки файла	Название файла может быть задано как: • фиксированное; • формируемое на основе текущей даты (пример: **2025_10_08**) – соответственно, каждый день будет создаваться новый файл; • формируемое на основе содержимого указанных регистров (в регистрах должна быть размещена строка с именем файла в кодировке ASCII, без использования символов кириллицы и спецсимволов. Максимальное количество регистров – 256).
Ограничение на размер файла	Ограничение на размер файла истории тревог, задаваемое в виде количества записей или мегабайт. Максимальное значение для каждого из вариантов – **65535** (при выборе мегабайт нужно убедиться, что в памяти панели или на USB-накопителе достаточно свободного места)
Поведение при достижении ограничения	Данная настройка определяет поведение при достижении файлом истории тревог указанного выше ограничения – прекращение записи в файл или перезапись самых старых записей

Продолжение таблицы 10.44

Настройка	Описание
Срок хранения файла	В случае установки галочки можно указать интервал времени с момента создания файла (в виде количества дней, не более **1000**), по истечению которого он будет удален. Обычно используется совместно с названием файла, формируемым на основе текущей даты, чтобы сохранять файлы истории за последние N дней
Структура файла	Для режима сохранения **Память панели** можно указать данные, которые будут сохраняться в файл, и определить порядок столбцов: • Дата возникновения тревоги; • Время возникновения тревоги; • Сообщение тревоги; • Количество срабатываний тревоги; • Время квитирования; • Время деактивации.

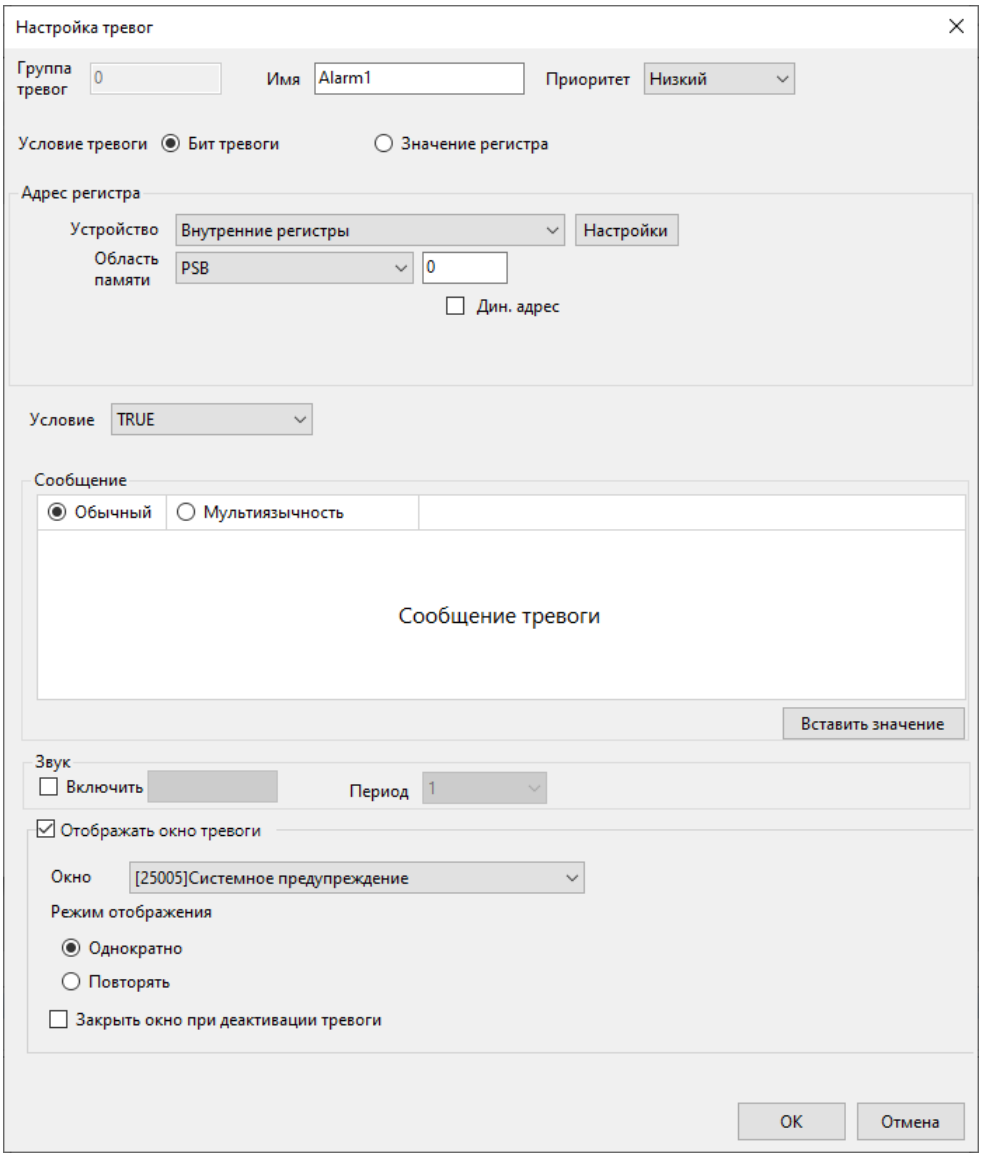


Рисунок 10.111 – Настройка тревог

Таблица 10.45 – Настройка тревог

Настройка	Описание
Имя	Имя тревоги (идентификатор)
Приоритет	Приоритет тревог – может использоваться в качестве типа фильтра в таблице тревог. Возможные значения: • низкий; • обычный; • высокий; • критический.
Условие	Для режима **Значение регистра** доступны следующие условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному. Для режима **Бит управления** выбирается состояние бита: • **TRUE**; • **FALSE**; • переход из **TRUE** в **FALSE**; • переход из **FALSE** в **TRUE**.
Сообщение тревоги	Статический текст или текст локализации, содержащий сообщение тревоги. С помощью команды **Вставить значение** можно добавить в текст сообщения заполнитель, в который будет подставлено значение указанного регистра
Звук	В случае установки галочки при возникновении тревоги в течение заданного времени будет активен зуммер или (только для расширенных модификаций панелей) воспроизводиться выбранный аудиофайл
Отображать окно тревоги	В случае установки галочки при возникновении тревоги будет открыто указанное окно. Если выбран режим отображения **Однократно** – то после закрытия окна оно будет открыто только при следующем выполнении условия тревоги. Если выбран режим отображения **Повторять** – то после закрытия окна оно будет открыто повторно спустя указанное время, если условие тревоги к этому моменту не перестанет выполняться. В случае установки галочки **Закрыть окно при деактивации тревоги** – окно автоматически будет закрыто в тот момент, когда условие тревоги перестанет выполняться

10.8.2 Таблица тревог

Элемент **Таблица тревог** используется для отображения тревог в табличном виде.

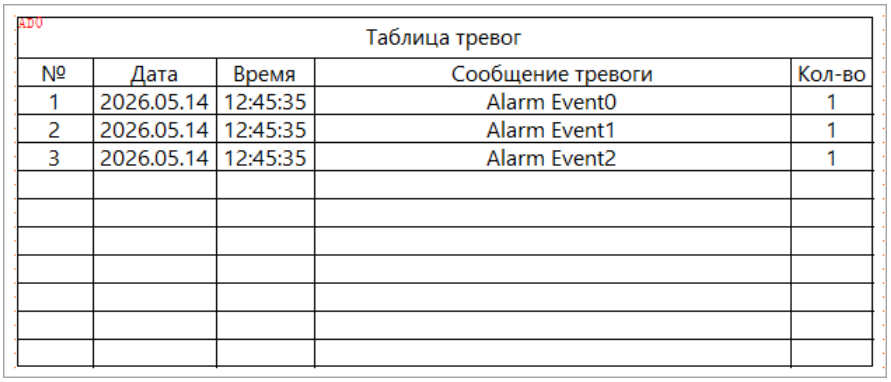
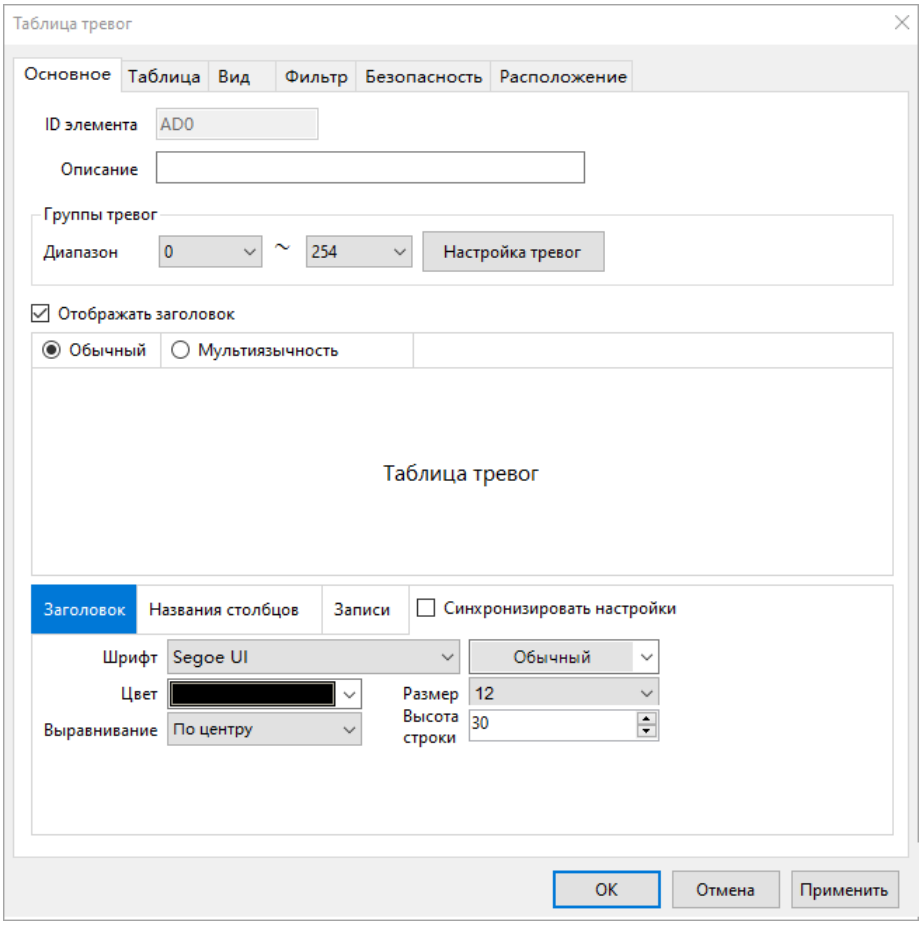


Рисунок 10.112 – Таблица тревог

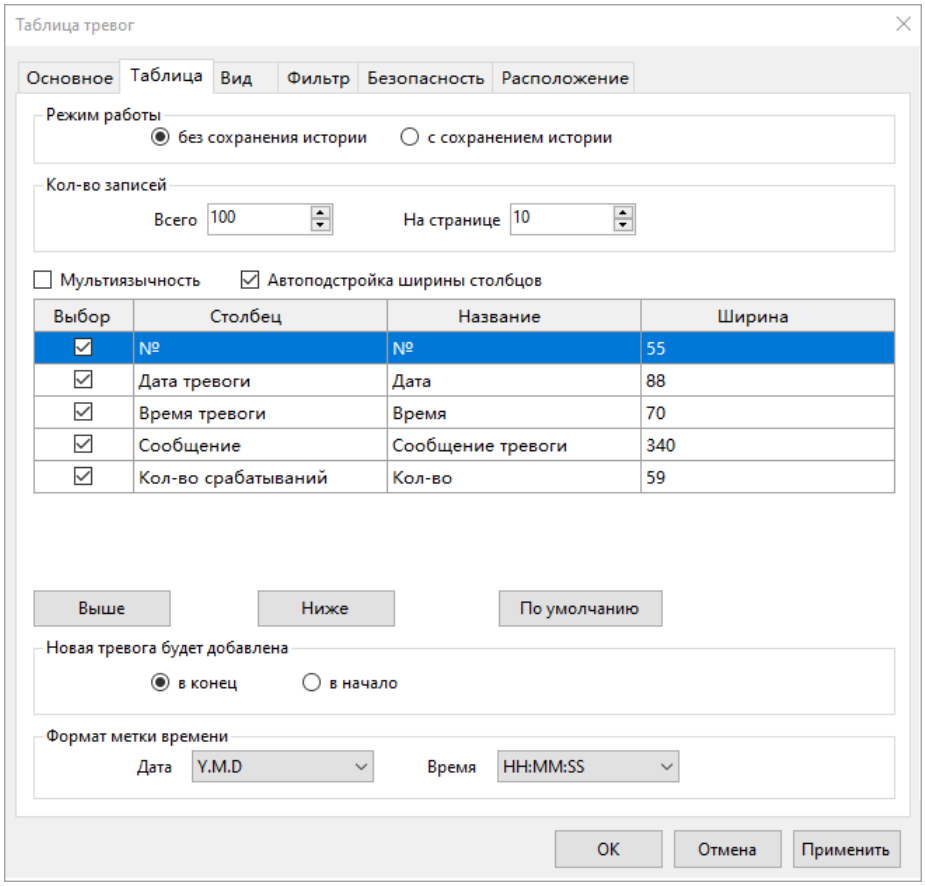
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.46 – Настройки элемента Таблица тревог

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.113 – Вкладка Основное**	
Группы тревог	Диапазон групп, тревоги которых будут отображаться в данной таблице. Нажатие на кнопку **Настройка тревоги** приводит к открытию соответствующего меню
Отображать заголовок	В случае установки галочки над таблицей отображается заголовок, который может быть задан статическим текстом или текстом локализации
Шрифт	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для заголовка таблицы, названий столбцов и текстов строк. Галочка **Синхронизировать настройки** позволяет сделать настройки всех этих текстов одинаковыми

Продолжение таблицы 10.46

Настройка	Описание
Вкладка **Таблица**  **Рисунок 10.114 – Вкладка Таблица**	
Режим работы	Без отображения истории (в этом случае в таблице отображаются только активные тревоги) или с отображением истории
Кол-во записей	Общее количество записей (не более **1000**), отображаемых в таблице, и количество записей на одной странице (не более **1000**). Для прокрутки таблицы используется полоса прокрутки. В режиме эмуляции для прокрутки таблицы нужно навести на неё курсор и использовать колёсико мыши
Мультиязычность	В случае установки галочки в качестве названий столбцов могут быть использованы тексты локализации
Автоподстройка ширины столбцов	В случае установки галочки ширина столбцов таблицы настраивается автоматически на основании ширины их заголовков
Столбцы	С помощью галочек пользователь указывает столбцы, которые будут отображаться в таблице: • Номер строки; • Дата возникновения тревоги; • Время возникновения тревоги; • Сообщение тревоги; • Количество срабатываний тревоги; • Время квитирования; • Время деактивации. Последние два столбца могут быть отображены только в режиме работы **С сохранением истории**. Для каждого столбца настраивается заголовок и ширина

Продолжение таблицы 10.46

Настройка	Описание
Новая тревога будет добавлена	Данная настройка определяет, в какую строку будет добавлена возникшая тревога – перед всеми существующими (сверху) или после всех существующих (снизу)
Формат метки времени	Возможные варианты для даты: • M.D.Y; • D.M.Y; • Y.M.D. Возможные варианты для времени: • HH:MM:SS; • HH:MM; • HH:MM:SS:MS.
Возможность квитирования	В случае установки галочки пользователь может квитировать (подтверждать) тревоги выбранным способом: • одиночным нажатием на строку тревоги (на любую из её ячеек); • двойным нажатием на строку тревоги; • нажатием и удержанием строки тревоги.
Управление видимостью	В случае установки галочки пользователь указывает адрес регистра, который будет содержать битовую маску для управления видимостью тревог в таблице: • бит 0: не отображать квитированные тревоги; • бит 1: не отображать неактивные тревоги; • бит 2: не отображать активные тревоги. Можно единовременно изменять значения нескольких битов

Вкладка **Вид**

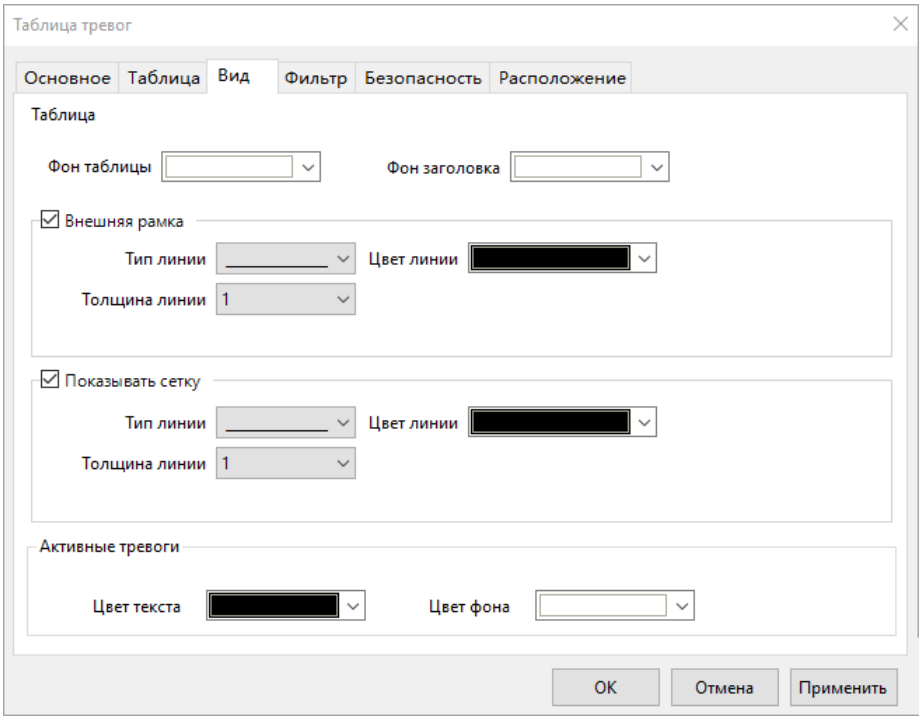


Рисунок 10.115 – Вкладка Вид

Таблица	Цвета фона заголовка и остальной части таблицы
Внешняя рамка	Тип, толщина и цвет линий внешней рамки таблицы
Показывать сетку	Тип, толщина и цвет линий сетки таблицы
Активные тревоги	Цвета текста и фона строк активных тревог

Продолжение таблицы 10.46

Настройка	Описание
Квитированные тревоги	Цвета текста и фона строк квитированных тревог
Деактивированные тревоги	Цвета текста и фона строк деактивированных тревог

Вкладка **Фильтр**

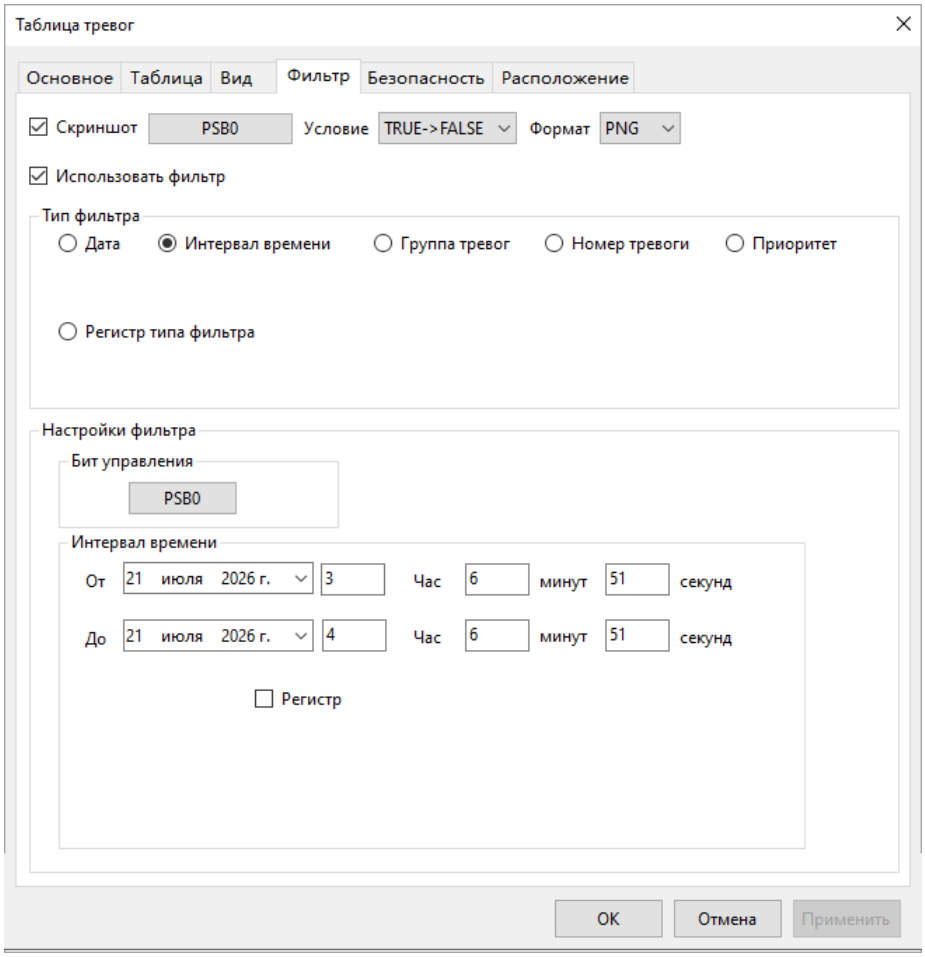


Рисунок 10.116 – Вкладка Фильтр

Скриншот	В случае установки галочки можно выбрать бит, по переднему или заднему фронту которого будет происходить экспорт видимого содержимого таблицы на USB-накопитель в виде файла формата .png. В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\png**
Использовать фильтр	В случае установки галочки пользователь выбирает тип фильтра, применяемый к таблице в течение времени, пока заданный бит имеет значение **TRUE**
Тип фильтра: дата	Дата задается константой или путём указания адреса регистра, начиная с которого будут размещены три регистра даты, содержащие её год, месяц и день (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: интервал времени	Интервал времени задается двумя константами или путём указания адреса регистра, начиная с которого будут размещены 12 регистров – 6 для метки времени начала интервала и 6 для метки времени конца интервала. Порядок разрядов метки времени – год, месяц, день, часы, минуты, секунды (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: группа тревог	Номер группы тревог задается константой или путём указания адреса регистра. Этот номер отображается в окне настройки тревог

Продолжение таблицы 10.46

Настройка	Описание
Тип фильтра: номер тревоги	Номер тревоги задается константой или путём указания адреса регистра. Этот номер отображается в окне настройки тревог
Тип фильтра: приоритет	Приоритет задается константой или путём указания адреса регистра. Возможные значения регистра: • 0 – низкий; • 1 – обычный; • 2 – высокий; • 3 – критический.
Тип фильтра: регистр	В данном случае тип фильтра определяется динамически на основании значения указанного регистра: • 0 – дата; • 1 – интервал времени; • 2 – группа тревог; • 3 – номер тревоги; • 4 – приоритет. Для любого из типов указывается адрес регистра, начиная с которого размещаются параметры фильтра (см. их описание выше)

10.8.3 Баннер тревог

Элемент **Баннер тревог** используется для отображения активных тревог в виде бегущей строки.

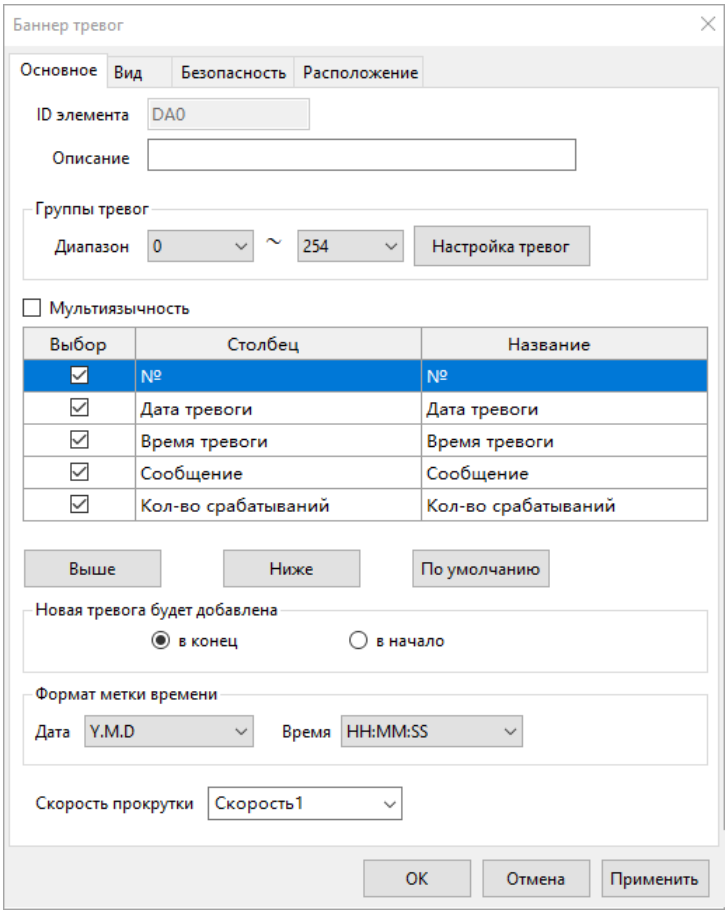


Рисунок 10.117 – Баннер тревог

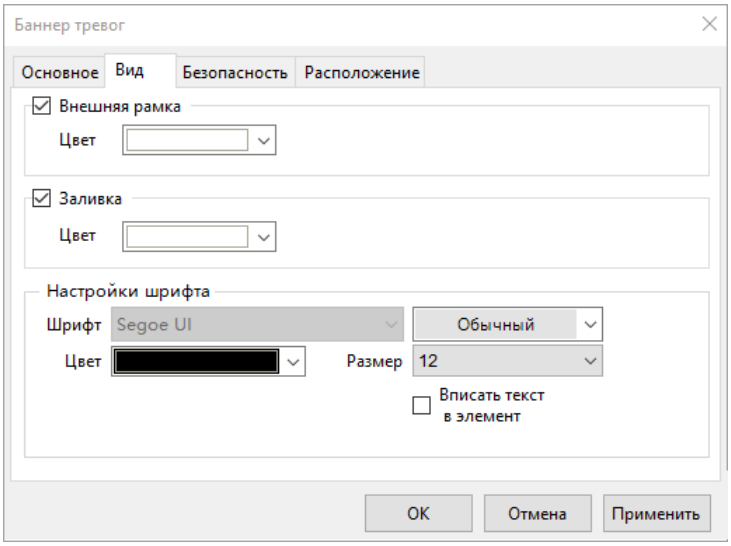
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.47 – Настройки элемента Баннер тревог

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.118 – Вкладка Основное**	
Группы тревог	Диапазон групп, тревоги которых будут отображаться в данном баннере. Нажатие на кнопку **Настройка тревоги** приводит к открытию соответствующего меню
Мультиязычность	В случае установки галочки в качестве названий столбцов могут быть использованы тексты локализации
Столбцы	С помощью галочек пользователь указывает столбцы, которые будут отображаться в баннере: • Номер строки; • Дата возникновения тревоги; • Время возникновения тревоги; • Сообщение тревоги; • Количество срабатываний тревоги. Для каждого столбца настраивается заголовок и ширина
Новая тревога будет добавлена	Данная настройка определяет, в каком месте баннера будет добавлено сообщение о возникшей тревоге – перед всеми отображаемыми сообщениями или после них

Продолжение таблицы 10.47

Настройка	Описание
Формат метки времени	Возможные варианты для даты: • M.D.Y; • D.M.Y; • Y.M.D. Возможные варианты для времени: • HH:MM:SS; • HH:MM; • HH:MM:SS:MS.
Скорость прокрутки	Скорость прокрутки баннера. Диапазон значений: 1…31, где 1 – самая медленная скорость, а 31 – самая быстрая
Вкладка **Вид**  **Рисунок 10.119 – Вкладка Вид**	
Внешняя рамка	Цвет рамки элемента
Заливка	Цвет заливки элемента
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста. В случае установки галочки **Вписать текст в элемент** размер шрифта будет установлен автоматически таким образом, чтобы текст элемента не выходил за его границы

10.9 Графики и таблицы

10.9.1 Выборка данных

Данное меню используется для настройки выборки данных – истории изменения числовых значений (каналов), которую можно отобразить с помощью элементов **Тренд** и **Таблица выборки**.

Выборка данных сохраняется в файл.

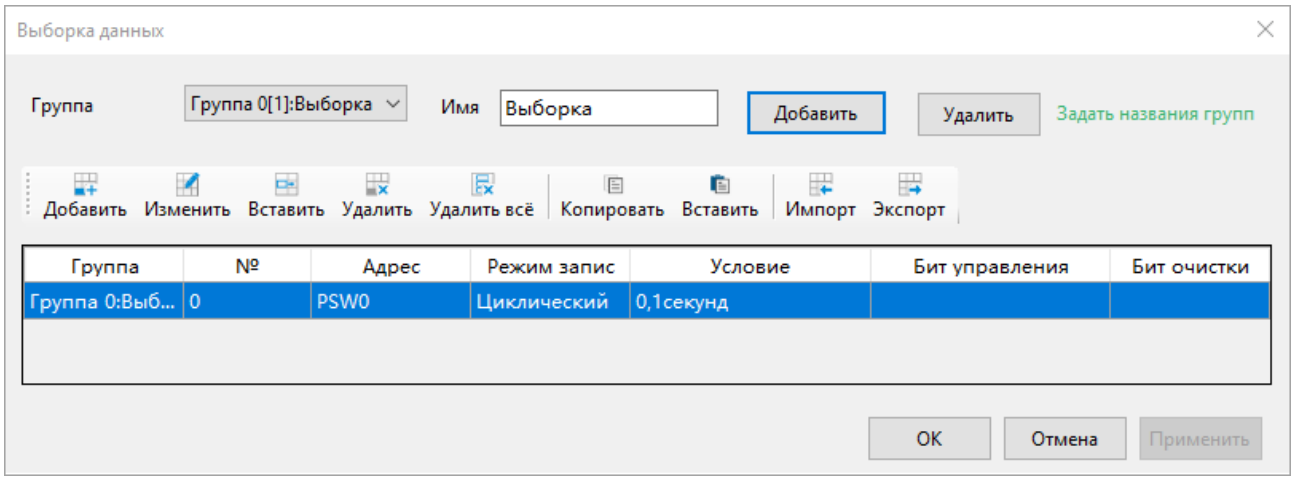


Рисунок 10.120 – Выборка данных

Сохраняемые значения распределяются по группам. Для создания новой группы следует ввести ее имя и нажать кнопку **Добавить**. Для изменения названий групп нужно нажать кнопку **Задать названия групп**.

Команда **Экспорт** позволяет выгрузить настройки выборок в виде файла .xlsx, .xls или .csv. Команда **Импорт** позволяет загрузить этот файл.

Настройки групп описаны в таблице:

Таблица 10.48 – Настройки групп выборки данных

Настройка	Описание
Вкладка **Основное**	

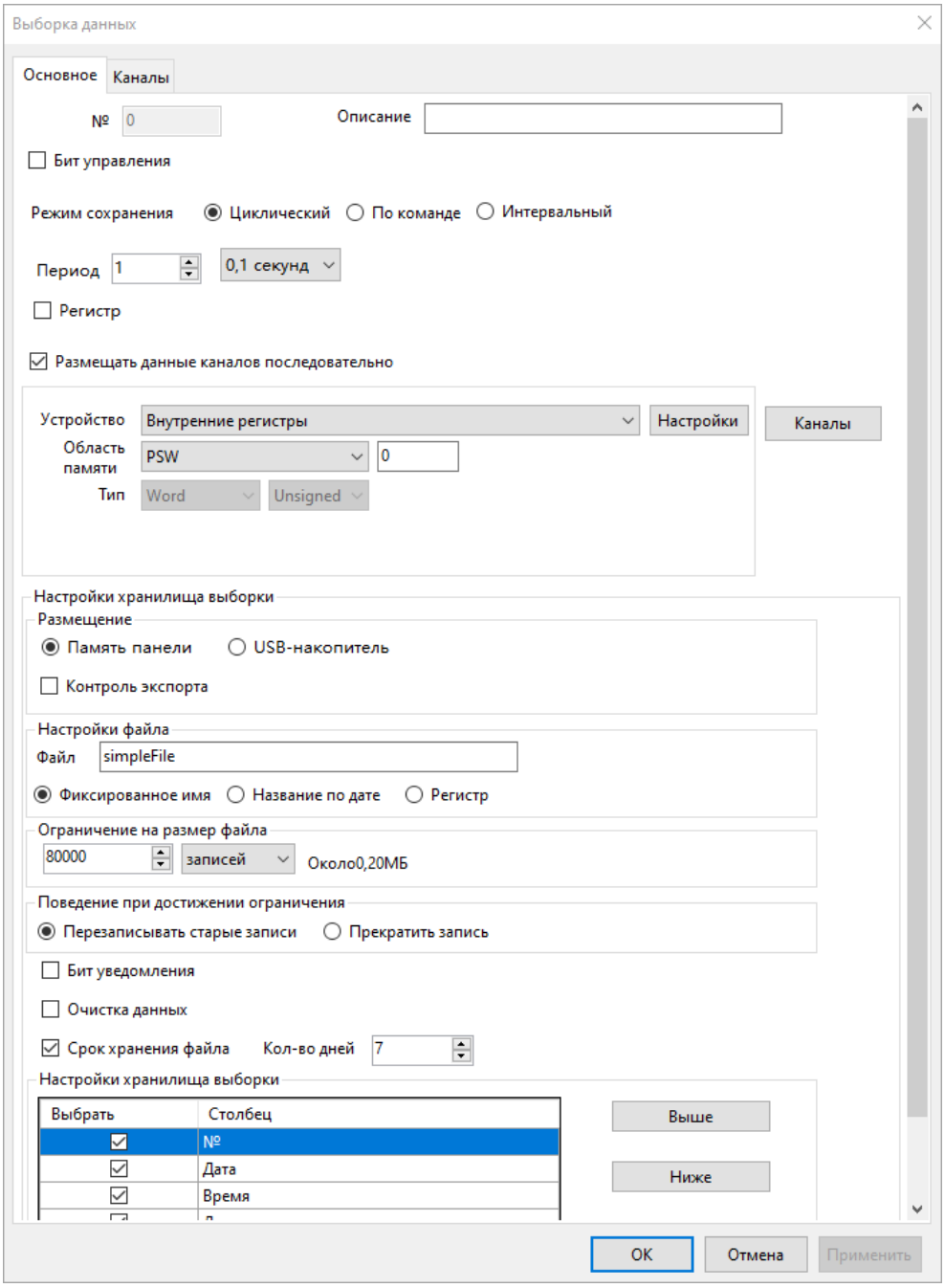


Рисунок 10.121 – Вкладка Основное

Бит управления	В случае установки галочки можно указать адрес бита. Сохранение значений каналов будет происходить только в том случае, если бит имеет заданное значение

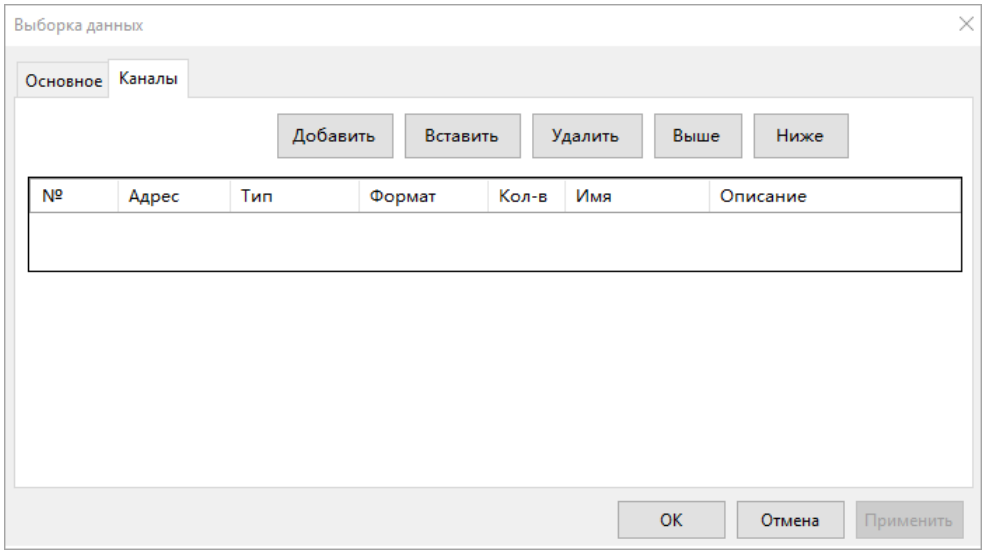
Продолжение таблицы 10.48

Настройка	Описание
Режим сохранения	Возможные варианты: • Циклический; • По команде; • Интервальный. В циклическом режиме сохранение значений каналов происходит с заданным периодом, значение которого определяется константой или путём указания адреса регистра. Единица измерения периода выбирается пользователем – десятые доли секунд, секунды или минуты. В режиме **По команде** сохранение значений каналов происходит каждый раз при выполнении заданного условия. Для варианта **Регистр** доступны следующие условия: • значение регистра элемента больше заданного; • значение регистра элемента меньше заданного; • значение регистра элемента больше или равно заданному; • значение регистра элемента меньше или равно заданному; • значение регистра элемента равно заданному; • значение регистра элемента не равно заданному. Для варианта **Бит** выбирается состояние бита – **TRUE** или **FALSE** В случае установки галочки **Инверсия бита после записи** (доступен только для варианта **Бит**) после каждого сохранения заданный бит переключается в противоположное состояние. В случае установки галочки **Частая проверка бита** условие сохранения будет проверяться раз в 50 мс. В режиме **Интервальный** сохранение значений каналов происходит с заданным периодом, но, в отличие от циклического, только в течение указанного интервала суток.
Размещать данные каналов последовательно	В случае установки галочки указывается адрес регистра, начиная с которого будут последовательно размещаться значения каналов данной группы. В случае отсутствия галочки – адреса регистров каналов указываются на вкладке **Настройка каналов** (индивидуально для каждого канала). Нажатие на кнопку **Каналы** приводит к открытию соответствующей вкладки

Продолжение таблицы 10.48

Настройка	Описание
Размещение	Выбор места сохранения файла выборки данных – в памяти панели или на подключенном USB-накопителе (рекомендуется по возможности использовать второй вариант, чтобы снизить износ встроенной памяти панели). В случае сохранения файла в памяти панели доступна галочка **Контроль экспорта**, которая позволяет указать адрес первого из 3 регистров, в которые будет записываться информация экспорта. Предположим, указан регистр PSW0, тогда: **PSW0** – команда экспорта. При записи значения 1 происходит экспорт выборки данных на USB-накопитель. При записи значения 2 помимо экспорта выполняется удаление файла из памяти панели. Для выполнения новой команды в регистр предварительно должно быть записано значение 0. **PSW1** – в этот регистр записывается значение прогресса операции экспорта в процентах. Значение **100** соответствует завершению экспорта. **PSW2** – в указанный регистр будет записан результат успешного выполнения экспорта или код ошибки: • 0 – экспорт в процессе; • 1 – экспорт завершен успешно; • 2 – USB-накопитель не подключен; • 3 – недостаточно памяти; • 4 – некорректный путь к файлу; • 5 – ошибка формата файла; • 6 – в процессе экспорта произошла ошибка. В режиме эмуляции: • для варианта **USB**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\sample** с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf. • для варианта **Память панели**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\db\sample** в бинарном формате. Для возможности его просмотра следует использовать команду экспорта (см. выше), которая сформирует в директории **\Temp\Run\storage\udisk\sample** файл с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf.
Настройки файла	Название файла может быть задано как: • фиксированное; • формируемое на основе текущей даты (пример: **2025_10_08**) – соответственно, каждый день будет создаваться новый файл; • формируемое на основе содержимого указанных регистров (в регистрах должна быть размещена строка с именем файла в кодировке ASCII, без использования символов кириллицы и спецсимволов. Максимальное количество регистров – 256).
Ограничение на размер файла	Ограничение на размер файла выборки данных, задаваемое в виде количества записей или мегабайт. Максимальное значение зависит от размеров встроенной памяти панели (но нужно убедиться, что в памяти панели или на USB-накопителе хватит свободного места)
Поведение при достижении ограничения	Данная настройка определяет поведение при достижении файлом выборки данных указанного выше ограничения – прекращение записи в файл или перезапись самых старых записей

Продолжение таблицы 10.48

Настройка	Описание
Срок хранения файла	В случае установки галочки можно указать интервал времени с момента создания файла (в виде количества дней, не более **1000**), по истечению которого файл будет удален. Обычно используется совместно с названием файла, формируемым на основе текущей даты, чтобы сохранять файлы выборки за последние N дней
Бит уведомления	В случае установки галочки можно указать бит, который будет устанавливаться в **TRUE** при достижении ограничения на размер файла
Очистка данных	По переднему или заднему фронту указанного бита выполняется очистка файла выборки
Структура файла	Для режима сохранения **Память панели** можно указать данные, которые будут сохраняться в файл, и определить порядок столбцов: • Номер записи; • Дата; • Время; • Значение.
Вкладка **Каналы**  **Рисунок 10.122 – Вкладка Каналы**	
Содержимое	Для каждого канала указывается: • адрес регистра (в случае снятия на вкладке **Основное** галочки **Размещать данные каналов последовательно**); • тип данных; • формат данных; • кол-во регистров строки (только для типа String); • название канала; • описание.

10.9.2 Тренд

Элемент **Тренд** используется для отображения значений выборки данных в виде графика.

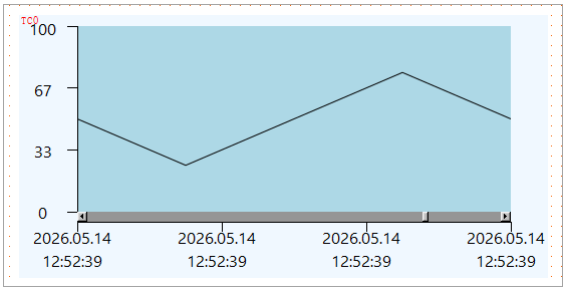
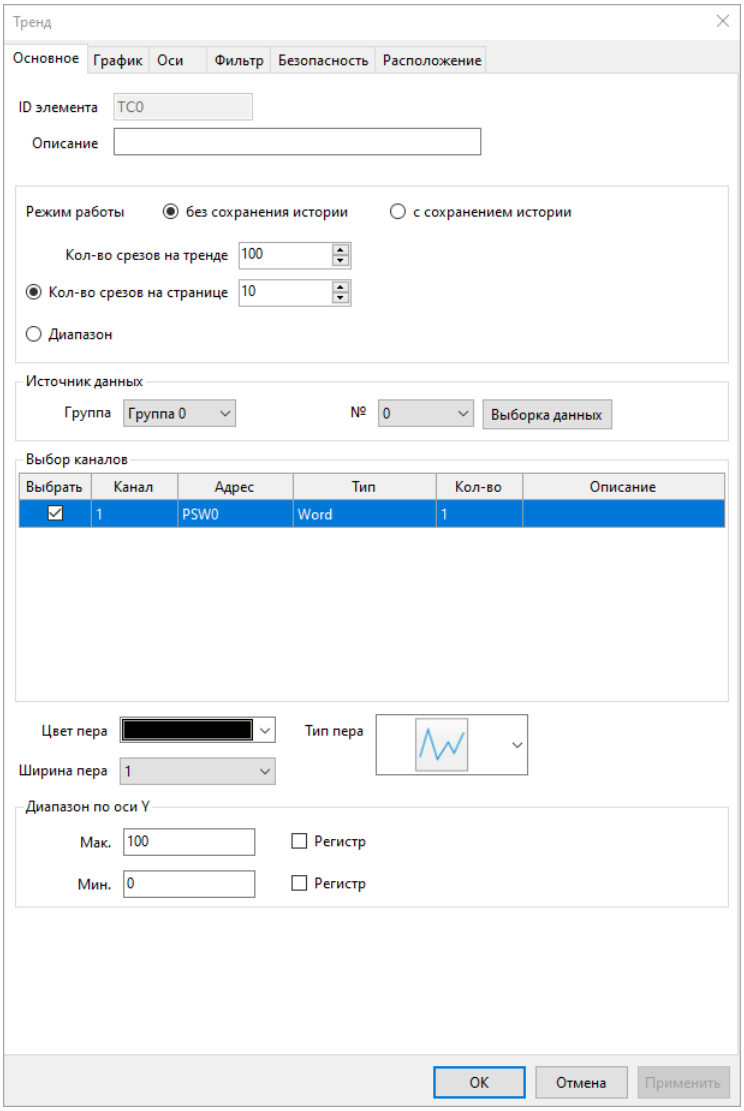


Рисунок 10.123 – Тренд

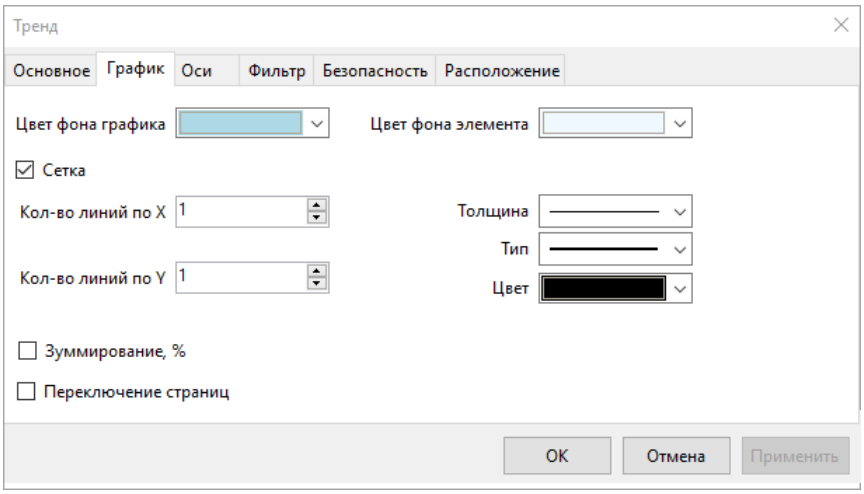
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

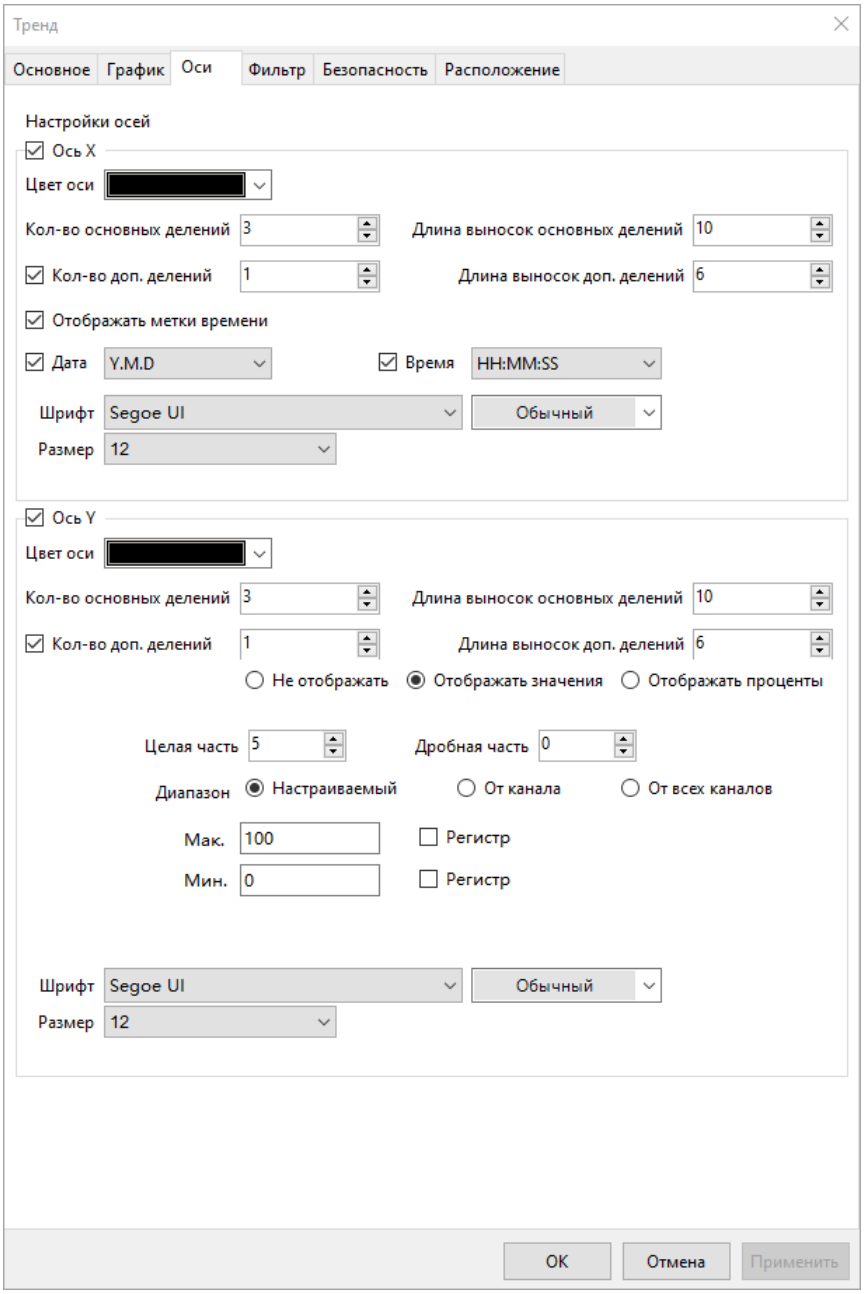
Таблица 10.49 – Настройки элемента Тренд

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.124 – Вкладка Основное**	
Режим работы	Без отображения истории (в этом случае на графике отображается только одна страница) или с отображением истории
Кол-во срезов на тренде	Общее количество записей, отображаемых на тренде (не более **5000**), и количество записей на одной странице (не более **5000**). Вместо указания количества записей на странице можно выбрать диапазон времени в указанных единицах измерения. Для прокрутки страницы графика используется полоса прокрутки
Источник данных	Выбор группы, каналы которой будут отображаться на графике. Нажатие на кнопку **Выборка данных** приводит к открытию соответствующего меню
Выбор каналов	Выбор каналов группы, перья которых будут отображаться на графике
Настройки пера	Цвет, тип, толщина и диапазон отображаемых значений пера по оси Y. Границы диапазона задаются константами или путём указания адресов регистров

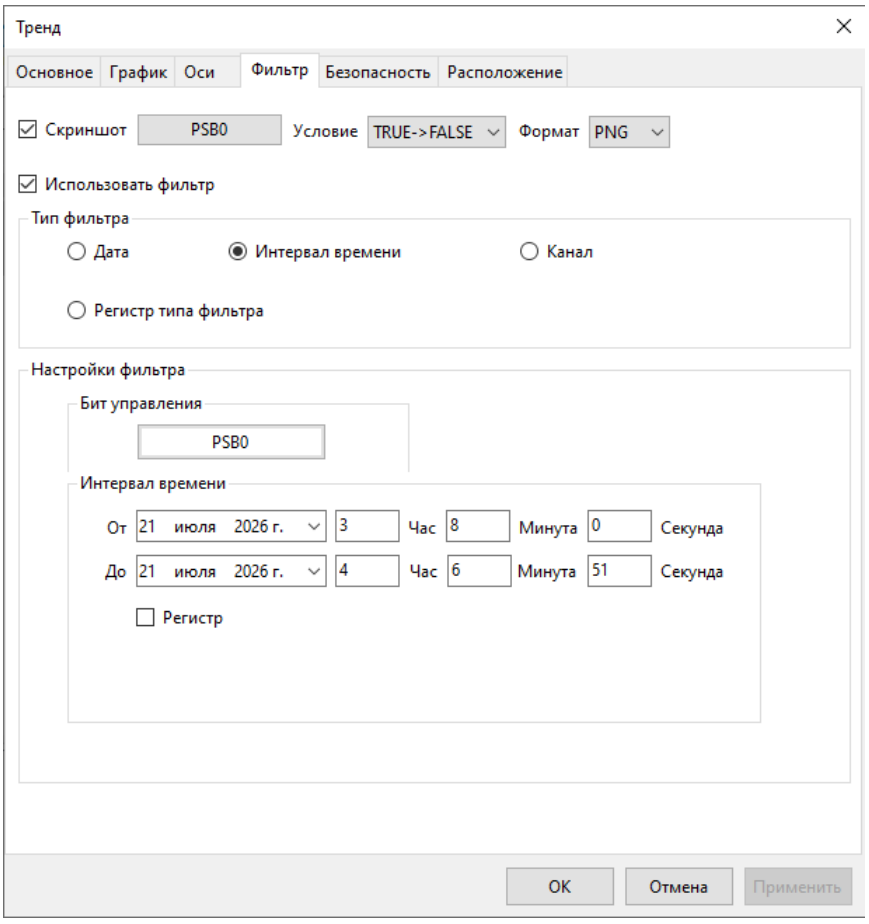
Продолжение таблицы 10.49

Настройка	Описание
Вкладка **График**  **Рисунок 10.125 – Вкладка График**	
Цвет фона графика	Цвет фона графика
Цвет фона элемента	Цвет фона элемента. Область элемента расположена за границами области графика и включает в себя его оси с их делениями и подписями
Сетка	В случае установки галочки на графике отображаются линии сетки. Для сетки указывается число линий по осям X и Y, а также толщина, тип и цвет этих линий
Отображать курсор	В случае установки галочки на графике отображается курсор, с помощью которого можно выделить конкретный срез графика (путём нажатия на соответствующую область). Рядом с курсором размещается легенда, в которой отображаются цвета перьев, названия их каналов, дата и время среза, а также значения перьев в этот момент. Набор столбцов легенды, а также цвет ее фона, линий и текста определяется пользователем
Зуммирование, %	В случае установки галочки можно указать адрес регистра, который будет использоваться для изменения масштаба тренда по оси X. Значение зума задается в процентах – например, значение **50** соответствует уменьшению масштаба в 2 раза
Переключение страниц	В случае установки галочки можно указать адрес регистра, который будет использоваться для прокрутки страниц тренда. Значение **0** соответствует самой ранней странице

Продолжение таблицы 10.49

Настройка	Описание
Вкладка **Оси**  **Рисунок 10.126 – Вкладка Оси**	
Ось X	Доступные настройки: • цвет оси • количество основных делений шкалы и длина их выносок; • количество дополнительных делений шкалы (в случае установки галочки) и длина их выносок; • формат метки времени; • выбор шрифта, его типа, размера и выравнивания текста. Возможные варианты формата для даты: • M.D.Y; • D.M.Y; • Y.M.D. Возможные варианты формата для времени:

Продолжение таблицы 10.49

Настройка	Описание
	• HH:MM:SS; • HH:MM; • HH:MM:SS:MS.
Ось Y	Доступные настройки: • цвет оси • количество основных делений шкалы и длина их выносок; • количество дополнительных делений шкалы (в случае установки галочки) и длина их выносок; • формат метки времени; • выбор шрифта, его типа, размера и выравнивания текста. Для подписей настраивается: • максимальное количество знаков целой и дробной части; • диапазон. Возможные варианты диапазона по оси Y: • настраиваемый – задаётся константами или путём указания адресов регистров; • от канала – диапазон определяется диапазоном выбранного канала, заданным в настройках пера на вкладке **Основное**; • от всех каналов – отображаются диапазоны всех каналов.
Вкладка **Фильтр**  **Рисунок 10.127 – Вкладка Фильтр**	

Продолжение таблицы 10.49

Настройка	Описание
Скриншот	В случае установки галочки можно выбрать бит, по переднему или заднему фронту которого будет происходить экспорт видимого содержимого таблицы на USB-накопитель в виде файла формата .png. В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\png**
Использовать фильтр	В случае установки галочки пользователь выбирает тип фильтра, применяемый к таблице в течение времени, пока заданный бит имеет значение **TRUE**
Тип фильтра: дата	Дата задается константой или путём указания адреса регистра, начиная с которого будут размещены три регистра даты, содержащие её год, месяц и день (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: интервал времени	Интервал времени задается двумя константами или путём указания адреса регистра, начиная с которого будут размещены 12 регистров – 6 для метки времени начала интервала и 6 для метки времени конца интервала. Порядок разрядов метки времени – год, месяц, день, часы, минуты, секунды (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: канал	Номер канала задается константой или путём указания адреса регистра
Тип фильтра: регистр	В данном случае тип фильтра определяется динамически на основании значения указанного регистра: • 0 – дата; • 1 – интервал времени; • 2 – канал. Для любого из типов указывается адрес регистра, начиная с которого размещаются параметры фильтра (см. описание выше)

10.9.3 Таблица выборки

Элемент **Таблица выборки** используется для отображения значений выборки данных в табличном виде.

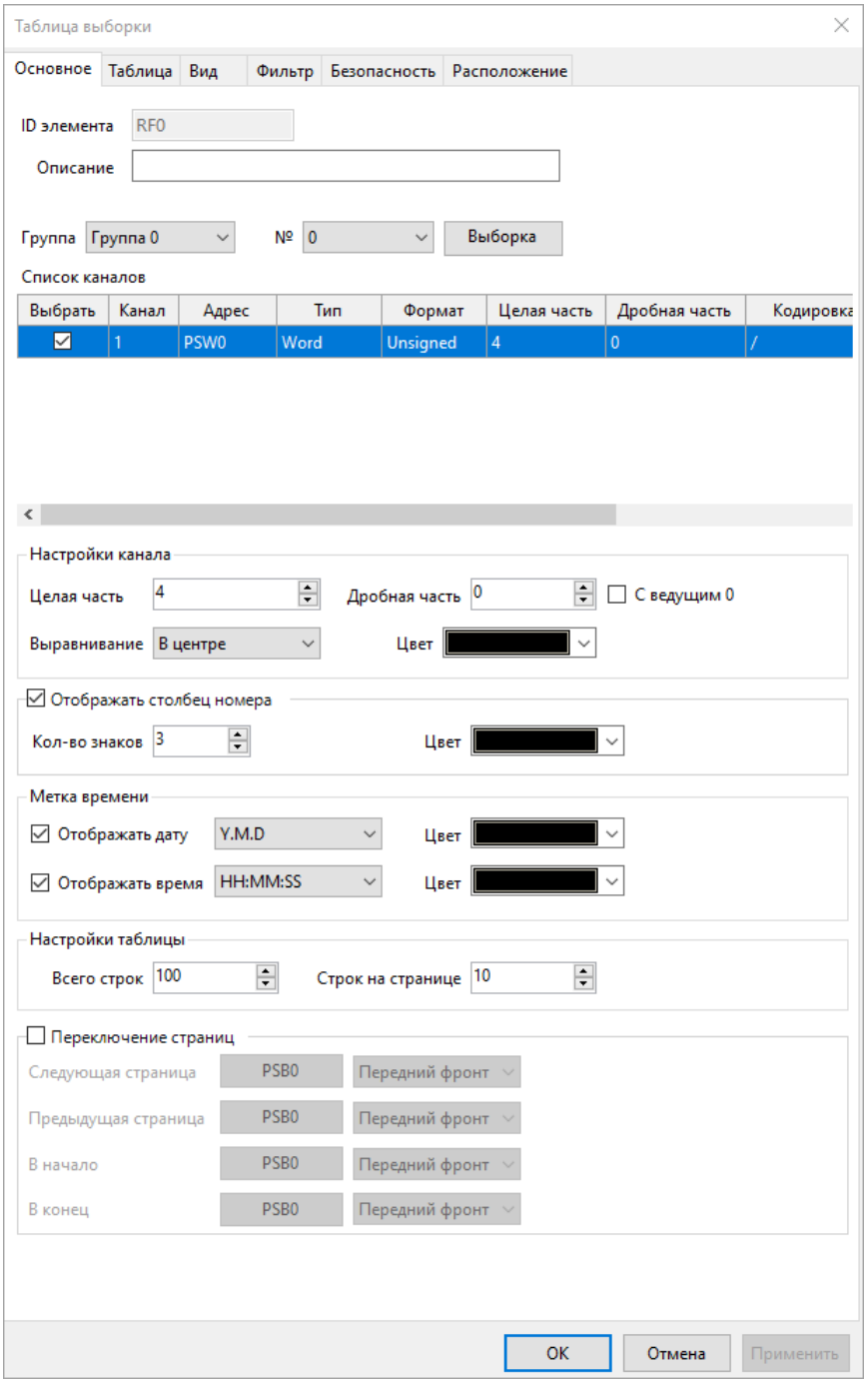


Рисунок 10.128 – Таблица выборки

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.50 – Настройки элемента Таблица выборки

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.129 – Вкладка Основное**	
Группа	Выбор группы, каналы которой будут отображаться в таблице. Нажатие на кнопку **Выборка данных** приводит к открытию соответствующего меню
Список каналов	Выбор каналов группы, значения которых будут отображаться в таблице
Настройки канала	Настройки выбранного в таблице канала: • максимальное количество знаков целой и дробной части; • в случае установки галочки **С ведущим 0** значение дополняется ведущими нулями до указанного количества разрядов целой части; • выравнивание текста (слева/справа/по центру); • цвет текста.

Продолжение таблицы 10.50

Настройка	Описание
Отображать столбец номера	В случае установки галочки в таблице отображается столбец с номерами строк. Для него указывается максимальное количество знаков номера и цвет текста
Метка времени	В случае установки галочек в таблице отображаются столбцы даты и времени, для которых настраивается формат и цвет текста. Возможные варианты формата для даты: • M.D.Y; • D.M.Y; • Y.M.D. Возможные варианты формата для времени: • HH:MM:SS; • HH:MM; • HH:MM:SS:MS.
Настройки таблицы	Общее количество записей (не более **100**), отображаемых в таблице, и количество записей на одной странице записей (не более **100**). Для прокрутки таблицы используется полоса прокрутки
Переключение страниц	Выбор адресов битов, по заданному фронту или инверсии которых будет происходить прокрутка таблицы: • к следующей странице; • к предыдущей странице; • к первой странице; • к последней странице.

Продолжение таблицы 10.50

Настройка	Описание
Вкладка **Таблица**	

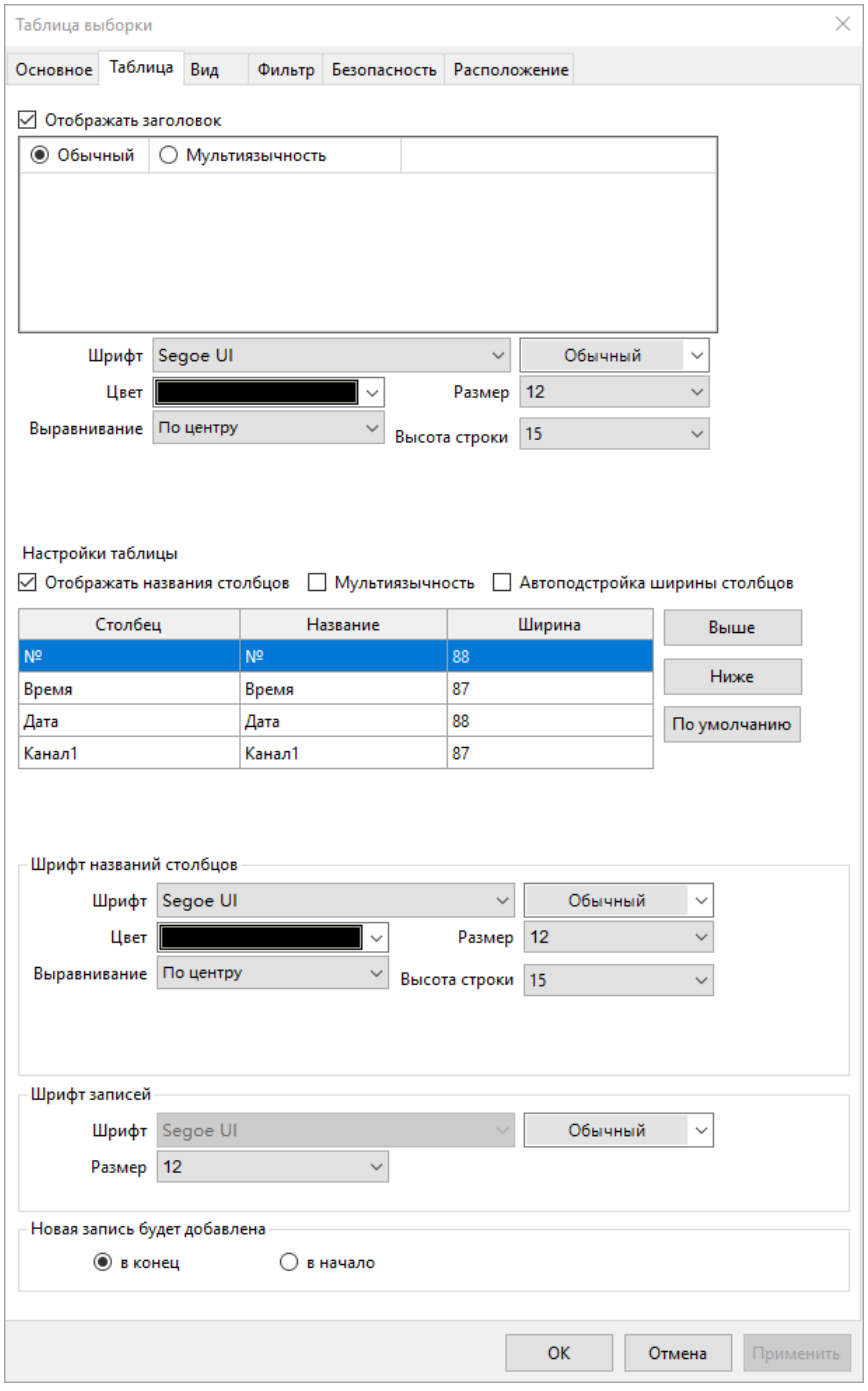


Рисунок 10.130 – Вкладка Таблица

Отображать заголовок	В случае установки галочки над таблицей отображается заголовок, который может быть задан статическим текстом или текстом локализации
Шрифт	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для заголовка таблицы
Отображать названия столбцов	В случае установки галочки в таблице будут отображаться названия столбцов
Мультиязычность	В случае установки галочки в качестве названий столбцов могут быть использованы тексты локализации
Автоподстройка ширины столбцов	В случае установки галочки ширина столбцов таблицы настраивается автоматически на основании ширины их заголовков

Продолжение таблицы 10.50

Настройка	Описание
Настройки таблицы	Для каждого столбца в таблице указывается заголовок и ширина
Шрифт названий столбцов	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для названий столбцов таблицы
Шрифт записей	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для текстов строк таблицы
Новая запись будет добавлена	Данная настройка определяет, в какую строку будет добавлена новая запись – перед всеми существующими (сверху) или после всех существующих (снизу)
Вкладка **Вид**	

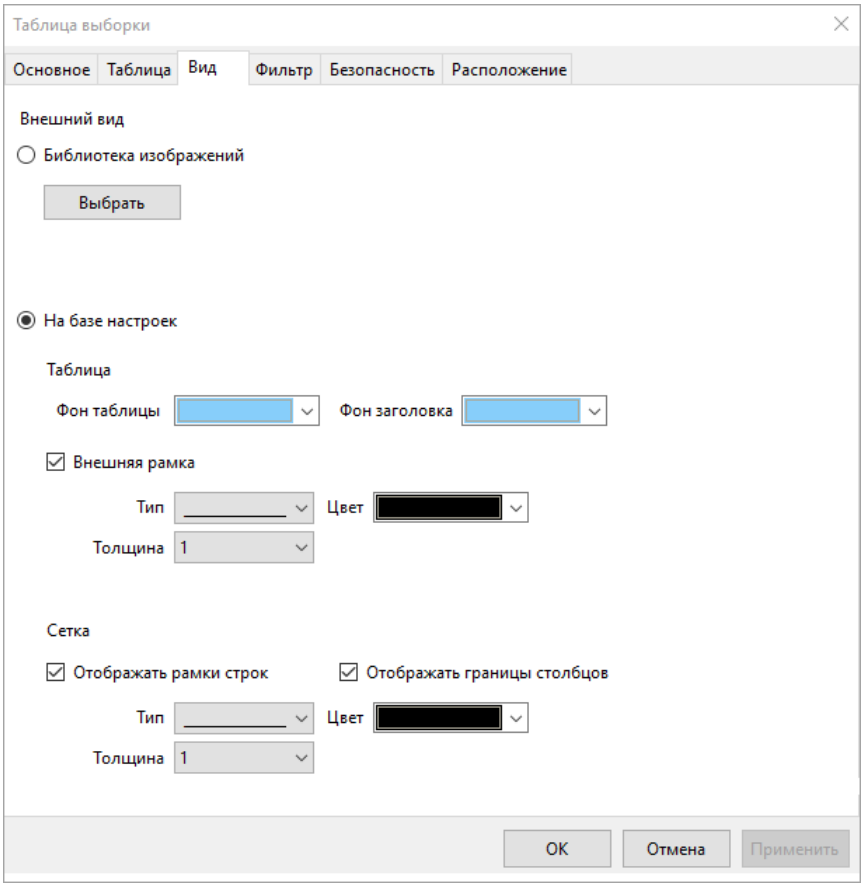


Рисунок 10.131 – Вкладка Вид

Библиотека изображений	Выбор изображения фона элемента из Библиотеки изображений (папка **DataForm**)
На базе настроек	В данном режиме внешний вид таблицы определяется заданными пользователем настройками, к которым относятся: • цвет фона таблицы и фона её заголовка; • тип, толщина и цвет внешней рамки таблицы; • тип, толщина и цвет сетки таблицы.
Вкладка **Фильтр**	
Скриншот	В случае установки галочки можно выбрать бит, по переднему или заднему фронту которого будет происходить экспорт видимого содержимого таблицы на USB-накопитель в виде файла формата .png. В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\png**

Продолжение таблицы 10.50

Настройка	Описание
Использовать фильтр	В случае установки галочки пользователь выбирает тип фильтра, применяемый к таблице в течение времени, пока заданный бит имеет значение **TRUE**
Тип фильтра: дата	Дата задается константой или путём указания адреса регистра, начиная с которого будут размещены три регистра даты, содержащие её год, месяц и день (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: интервал времени	Интервал времени задается двумя константами или путём указания адреса регистра, начиная с которого будут размещены 12 регистров – 6 для метки времени начала интервала и 6 для метки времени конца интервала. Порядок разрядов метки времени – год, месяц, день, часы, минуты, секунды (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: регистр	В данном случае тип фильтра определяется динамически на основании значения указанного регистра: • 0 – дата; • 1 – интервал времени. Для любого из типов указывается адрес регистра, начиная с которого размещаются параметры фильтра (см. их описание выше)

10.9.4 Таблица данных

Элемент **Таблица данных** используется для отображения значений регистров в табличном виде.

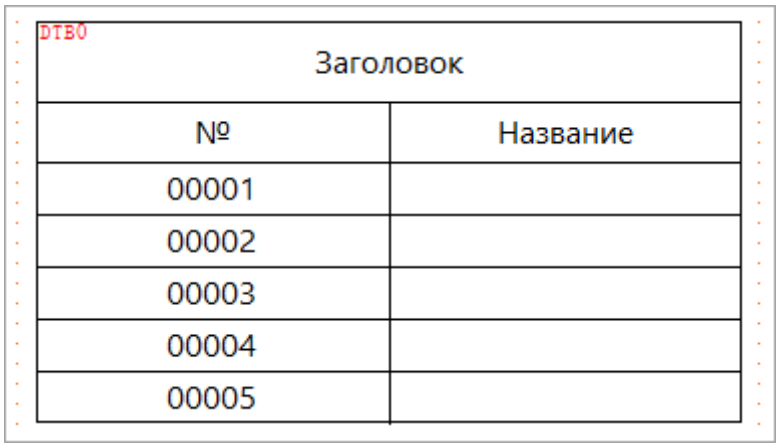
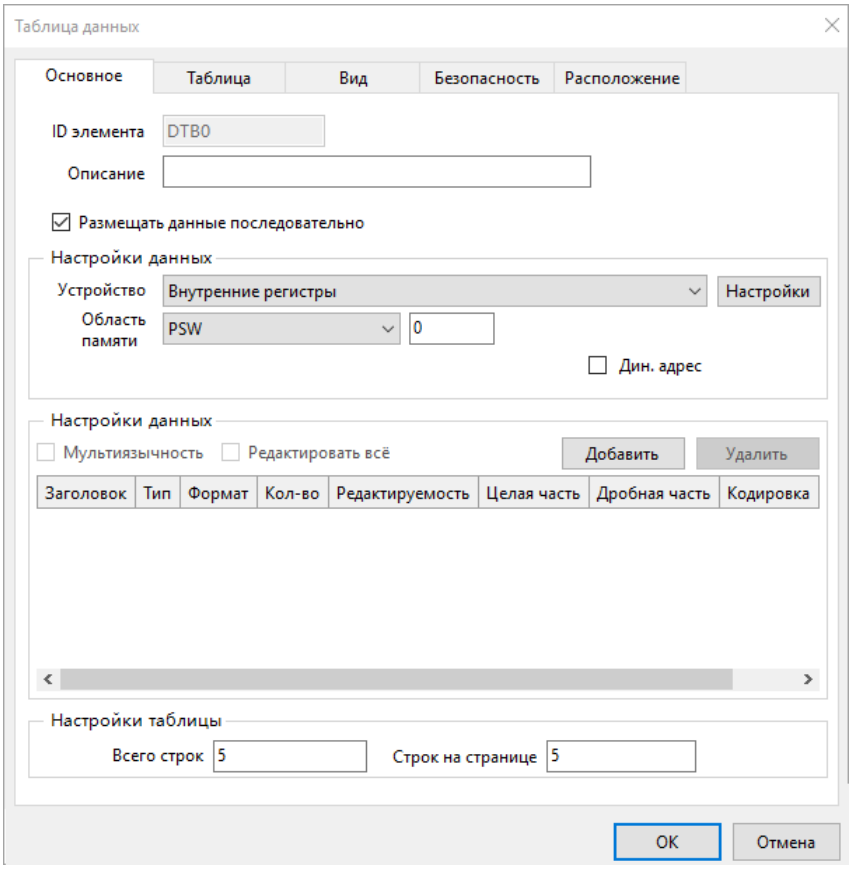


Рисунок 10.132 – Таблица данных

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.51 – Настройки элемента Таблица данных

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.133 – Вкладка Основное**	
Размещать данные последовательно	В случае установки галочки **Размещать данные последовательно** указывается адрес регистра, начиная с которого размещаются значения, указанные в таблице. В области **Настройки данных** добавляются столбцы, отображаемые в таблице. Для каждого столбца указывается: • название (в виде статического текста или, в случае установки галочки **Мультиязычность**, текста локализации); • тип данных; • формат; • ширина столбца; • максимальное количество знаков целой и дробной части (только для числовых типов); • в случае установки галочки **С ведущим 0** значение дополняется ведущими нулями до указанного количества разрядов целой части; • кодировка и количество объектов, занимаемых строкой (объект зависит от типа строки: для **Word_String** он представляет собой **WORD** и т. д.) – только для строковых типов. Галочка **Редактируемость** определяет возможность редактирования значения данного параметра. Для ввода значения следует нажать на соответствующую ячейку таблицы. Галочка **Редактировать всё** устанавливает галочки **Редактируемость** для всех столбцов таблицы. В случае отсутствия галочки **Размещать данные последовательно** описанные выше параметры задаются индивидуально для каждой ячейки таблицы путём нажатия на неё в редакторе визуализации

Продолжение таблицы 10.51

Настройка	Описание
Настройки таблицы	Общее количество строк, отображаемых в таблице (не более **100**), и количество строк на одной странице (не более **100**). Для прокрутки таблицы используется полоса прокрутки

Вкладка **Таблица**

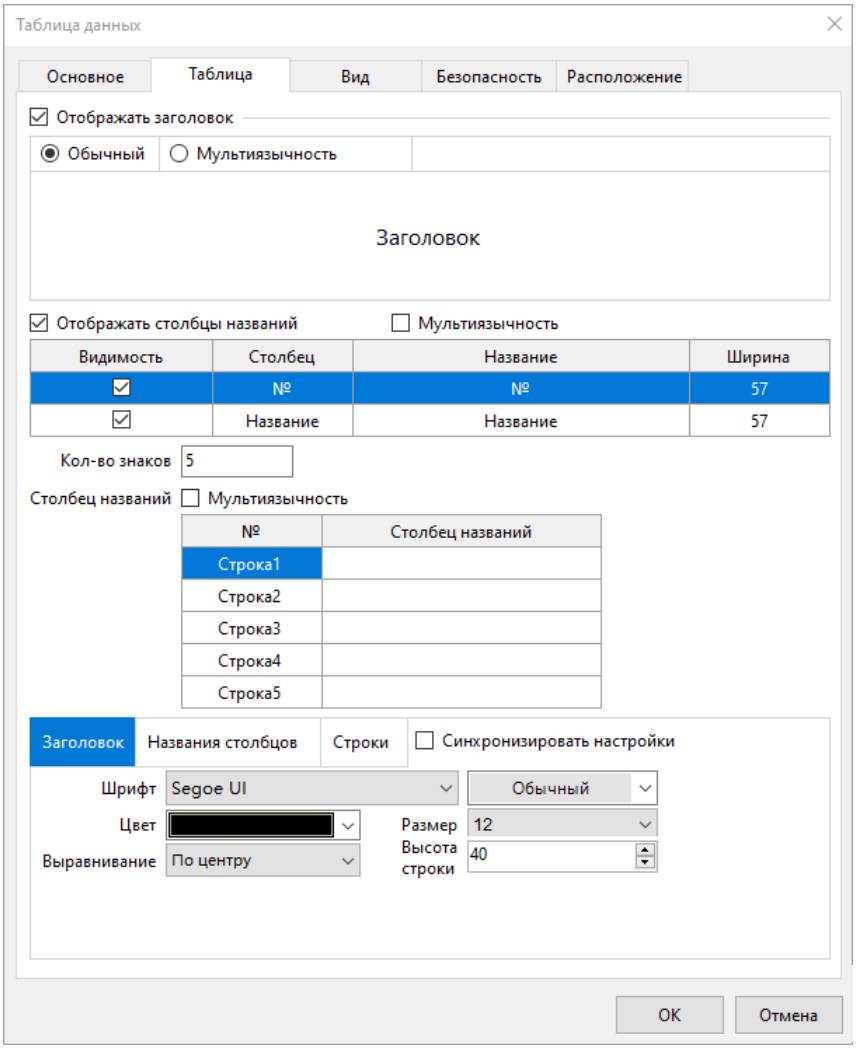
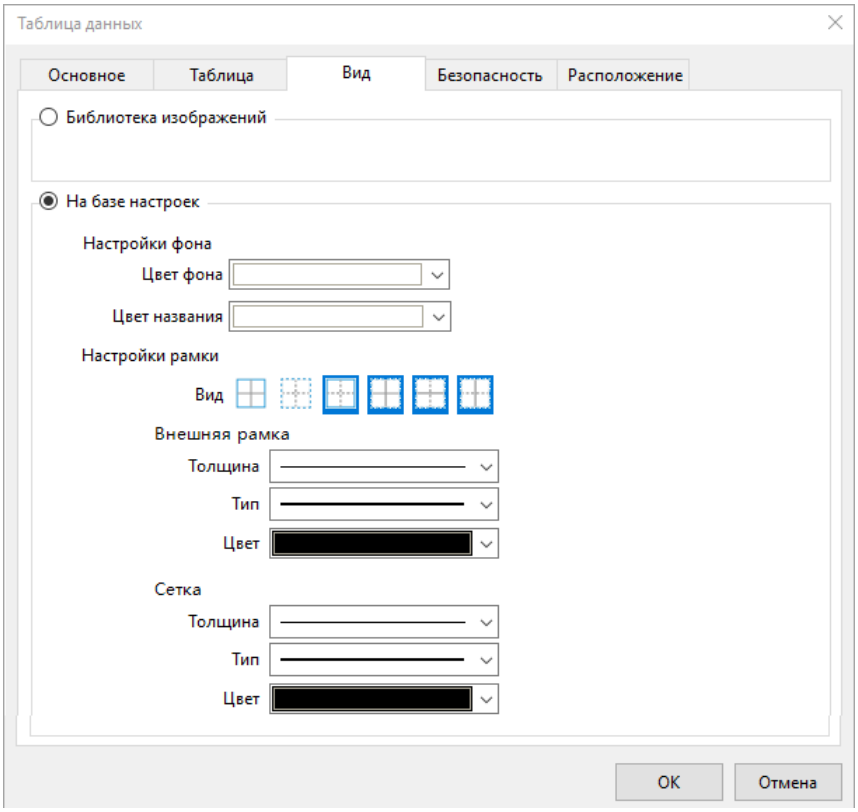


Рисунок 10.134 – Вкладка Таблица

Отображать заголовок	В случае установки галочки над таблицей отображается заголовок, который может быть задан статическим текстом или текстом локализации
Отображать столбец названия	В случае установки галочки в таблице будет отображаться столбец названий строк
Мультиязычность	В случае установки галочки в качестве названий столбца номеров и столбца названий строк могут быть использованы тексты локализации
Настройки	Видимость, названия и ширина столбцов номеров и названий строк. Для столбца номера указывается максимальное количество знаков
Столбец названий	Названия строк. В случае установки галочки **Мультиязычность** в качестве названий могут использоваться тексты локализации
Шрифт	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для заголовка таблицы, названий столбцов и текстов строк. Галочка **Синхронизировать настройки** позволяет сделать настройки всех этих текстов одинаковыми

Продолжение таблицы 10.51

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.135 – Вкладка Вид**	
Библиотека изображений	Выбор изображения фона элемента из **Библиотеки изображений** (папка **DataForm**)
На базе настроек	В данном режиме внешний вид таблицы определяется заданными пользователем настройками: • цвет фона таблицы и фона её заголовка; • внешний вид рамки таблицы; • тип, толщина и цвет внешней рамки таблицы; • тип, толщина и цвет сетки таблицы.

10.9.5 XY график

Элемент **XY график** используется для отображения графика, координаты каждой точки которого считываются из указанных регистров. В момент обновления графика происходит перерисовка всех его точек.

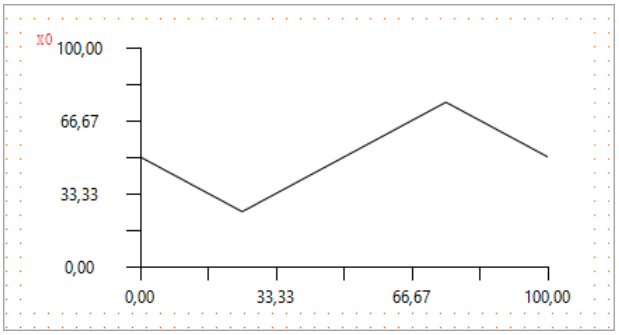
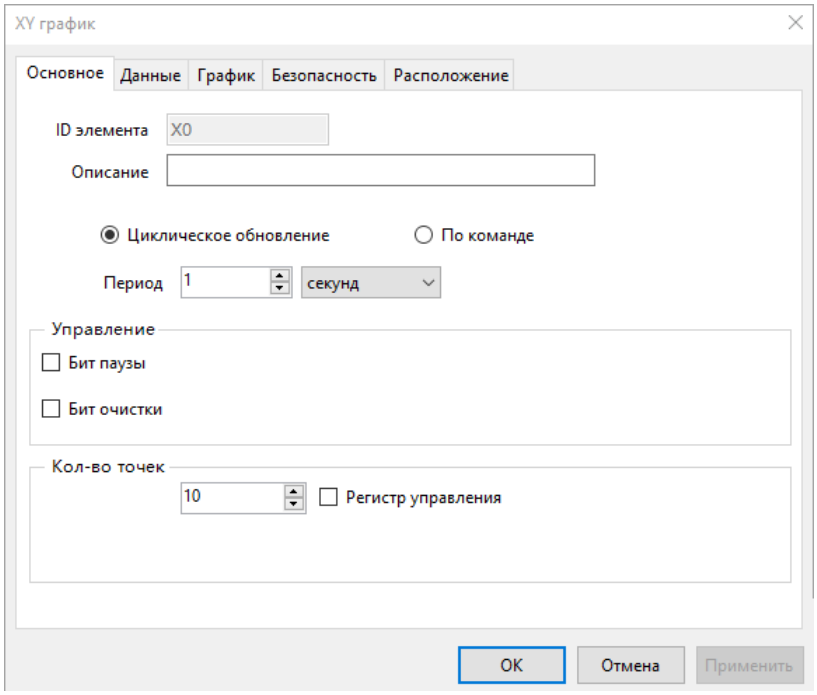


Рисунок 10.136 – XY график

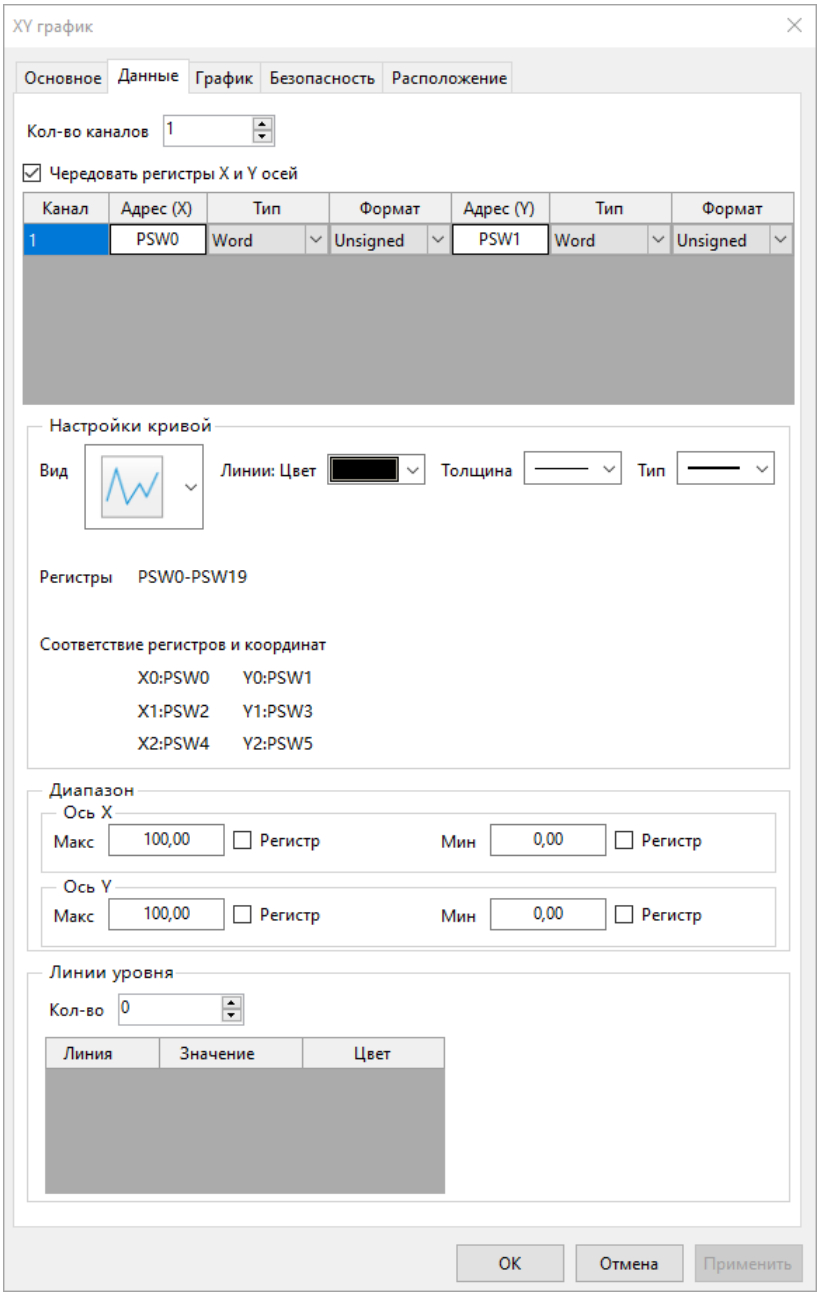
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.52 – Настройки элемента XY график

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.137 – Вкладка Основное**	
Режим	Режим обновления графика: • циклически (в заданных единицах времени – десятых долях секунд, секундах или минутах; максимальное значение – **35791**); • по команде (по выбранному фронту бита с указанным адресом)
Управление	В случае установки галочек можно указать адреса бита паузы и бита очистки. Если бит паузы имеет заданное значение, то перерисовка графика останавливается до того момента, пока он не примет противоположное значение. По выбранному фронту бита очистки происходит удаление всех линий графика (это не влияет на значения регистров с координатами точек)
Кол-во точек	Количество точек, отображаемых на графике (не более **1024**). Задается константой или путем указания адреса регистра

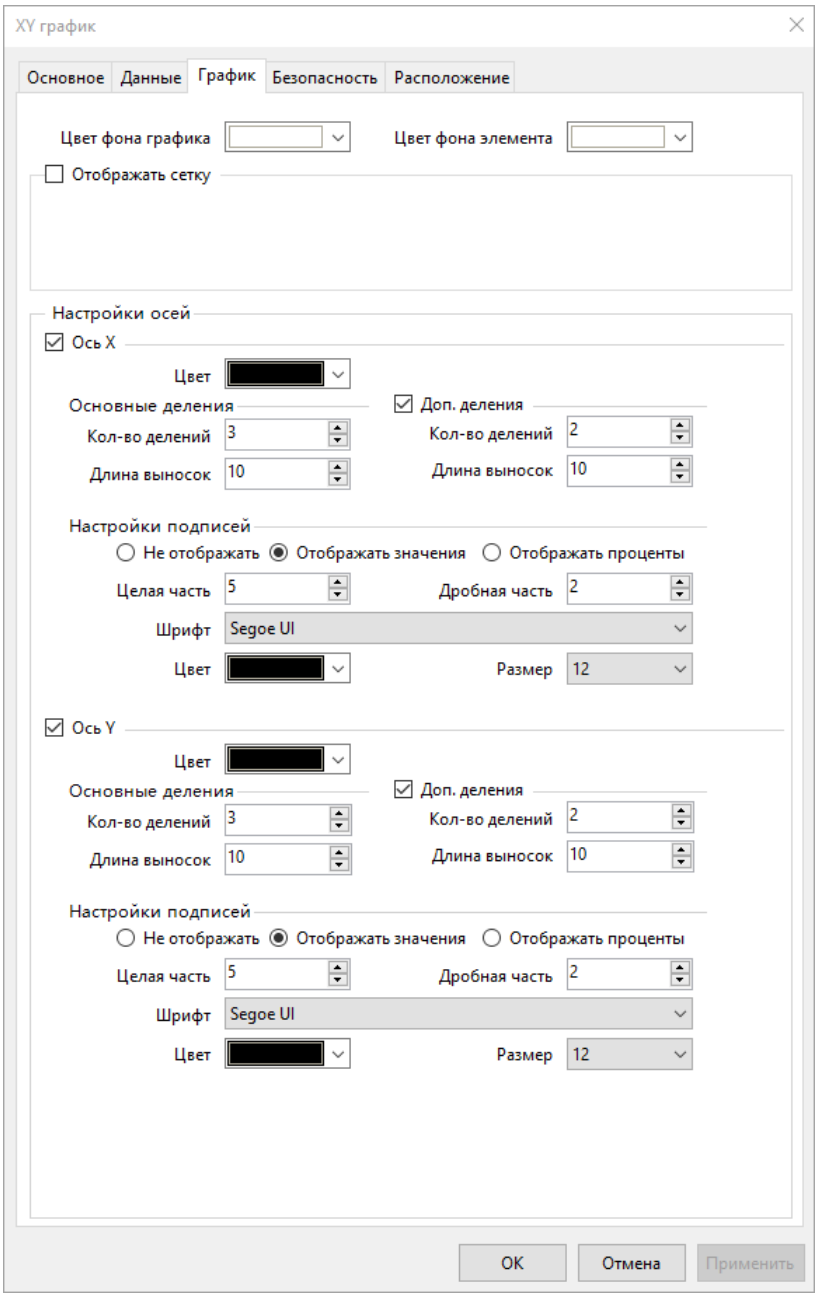
Продолжение таблицы 10.52

Настройка	Описание
Вкладка **Данные**  **Рисунок 10.138 – Вкладка Данные**	
Кол-во каналов	Количество каналов данных, отображаемых на графике (не более **16**)

Продолжение таблицы 10.52

Настройка	Описание
Чередовать регистры X и Y осей	В случае установки галочки в таблице ниже для каждого канала указывается только адрес начального регистра для координат X точек графика. Адреса регистров для координат Y рассчитываются автоматически как +1 к адресу регистра координаты X соответствующей точки. Пример: пусть **Кол-во точек = 3**, тип данных координат – Word, адрес регистра X – **PSW0**. Тогда координаты точек будут располагаться в следующих регистрах: • Точка 1: X – PSW0, Y – PSW1; • Точка 2: X – PSW2, Y – PSW3; • Точка 3: X – PSW4, Y – PSW5. В случае отсутствия галочки в таблице ниже для каждого канала указывается адрес начального регистра для координат X точек графика и адрес начального регистра для координат Y точек графика. Пример: пусть **Кол-во точек = 3**, тип данных координат – Word, адрес регистра X – **PSW0**, адрес регистра Y – **PSW10**. Тогда координаты точек будут располагаться в следующих регистрах: • Точка 1: X – PSW0, Y – PSW10; • Точка 2: X – PSW1, Y – PSW11; • Точка 3: X – PSW2, Y – PSW12.
Содержимое	Для каждого канала указывается: • адрес начального регистра для координат X точек графика; • адрес начального регистра для координат Y точек графика (в случае отсутствия галочки **Чередовать регистры X и Y осей**); • тип данных координат (Word, DWord или DDWord); • формат данных координат.
Настройки канала	Для выбранного в таблице выше канала настраивается: • вид графика; • цвет, толщина и тип линий и/или точек графика.
Диапазон	Диапазон графика по осям X и Y. Границы диапазона задаются константами или путём указания адресов регистров
Линии уровня	Линии уровня параллельны оси X и позволяют обозначить на графике отметки для заданных значений. Пользователь указывает количество линий уровня (не более **16**) и для каждой настраивает: • значение по оси Y (задается константой или путем указания адреса регистра); • цвет.

Продолжение таблицы 10.52

Настройка	Описание
Вкладка **График**  **Рисунок 10.139 – Вкладка График**	
Цвет фона графика	Цвет фона графика
Цвет фона элемента	Цвет фона элемента. Область элемента расположена за границами области графика и включает в себя его оси с их делениями и подписями

Продолжение таблицы 10.52

Настройка	Описание
Отображать сетку	В случае установки галочки в элементе отображается сетка. Пользователь настраивает: • количество линий сетки по осям X и Y; • тип линий; • цвет линий.
Настройки осей	Для каждой оси X и Y настраивается: • цвет; • количество основных делений шкалы и длина их выносок; • количество дополнительных делений шкалы (в случае установки галочки) и длина их выносок; • выбор режима отображения подписей (не отображать/отображать значения/отображать проценты). Для подписей настраивается: • максимальное количество знаков целой и дробной части; • шрифт, цвет и размер текста.

10.9.6 XY тренд

Элемент **XY тренд** используется для отображения графика, координаты последней точки которого считываются из указанных регистров. В момент обновления графика происходит добавление на график новой точки.

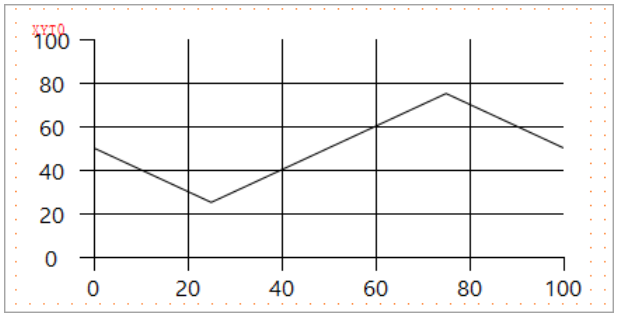


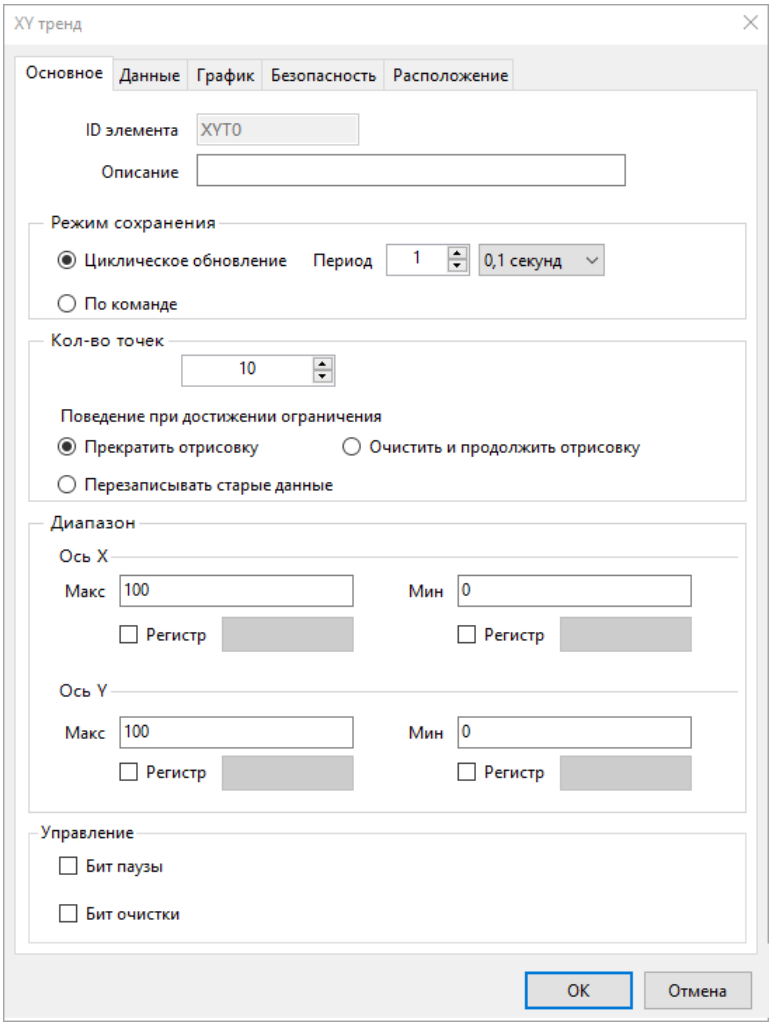
Рисунок 10.140 – XY тренд

Типовые настройки элемента описаны в разделе 10.1.

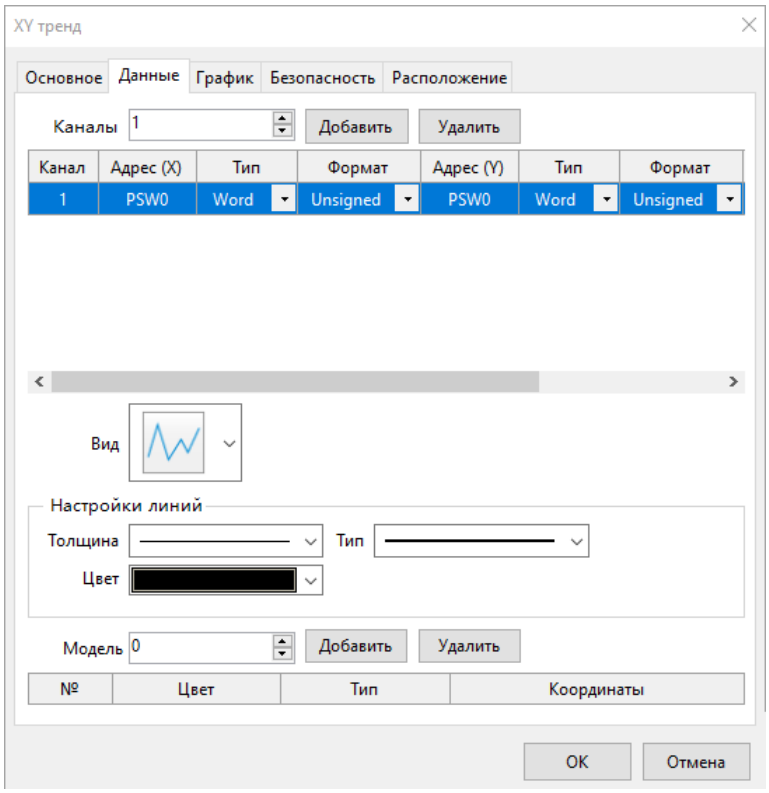
Настройки вкладки **График** описаны в таблице 10.52.

Уникальные настройки элемента описаны в таблице:

Таблица 10.53 – Настройки элемента XY тренд

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.141 – Вкладка Основное**	
Режим	Режим обновления графика: • циклически (в заданных единицах времени – десятых долях секунд, секундах или минутах; максимальное значение – **35791**); • по команде – по выбранному фронту бита с указанным адресом или при каждом срабатывании заданного условия: • значение регистра больше заданного; • значение регистра меньше заданного; • значение регистра больше или равно заданному; • значение регистра меньше или равно заданному; • значение регистра равно заданному; • значение регистра не равно заданному.
Кол-во точек	Количество точек, отображаемых на графике (не более **2048**). Задается константой или путем указания адреса регистра
Поведение при достижении ограничения	Поведение при отрисовке на графике указанного количества точек: • прекращение отрисовки; • полная очистка графика и продолжение его отрисовки с первой точки; • продолжение отрисовки с постепенным удалением самых «старых» точек.
Диапазон	Диапазон графика по осям X и Y. Границы диапазона задаются константами или путём указания адресов регистров

Продолжение таблицы 10.53

Настройка	Описание
Управление	В случае установки галочек можно указать адреса бита паузы и бита очистки. Если бит паузы имеет заданное значение, то перерисовка всех каналов графика останавливается до того момента, пока он не примет противоположное значение. По выбранному фронту бита очистки происходит удаление всех линий графика (это не влияет на значения регистров с координатами точек)
Вкладка **Данные**  **Рисунок 10.142 – Вкладка Данные**	
Каналы	Количество каналов данных, отображаемых на графике (не более **50**). Для каждого канала указывается: • адрес начального регистра для координат X точек графика; • адрес начального регистра для координат Y точек графика; • тип данных координат (Word, DWord или DDWord); • формат данных координат; • бит паузы – если он имеет заданное значение, то перерисовка этого канала графика останавливается до того момента, пока он не примет противоположное значение.
Настройки канала	Для выбранного в таблице выше канала настраивается: • вид графика; • цвет, толщина и тип линий и/или точек графика.
Модель	Пользователь может добавить до 16 моделей – заранее построенных графиков, которые будут отображаться элементом (например, в качестве эталона). Для каждой модели указывается: • тип отрисовки – линии или точки; • координаты точек; • цвет линий или точек.

10.9.7 Секторная диаграмма

Элемент **Секторная диаграмма** используется для отображения заданного количества регистров в виде секторной диаграммы. Значения регистров определяют относительный размер сектора. Например, если указано **Кол-во секторов = 4** и соответствующие им регистры имеют значения 100/40/40/20, то первый сектор займет половину области элемента (потому что сумма значений остальных также равна 100).

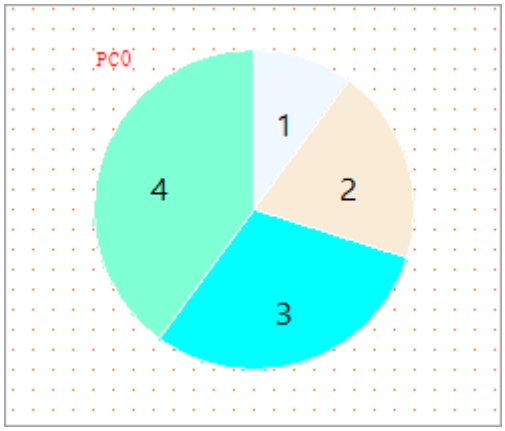
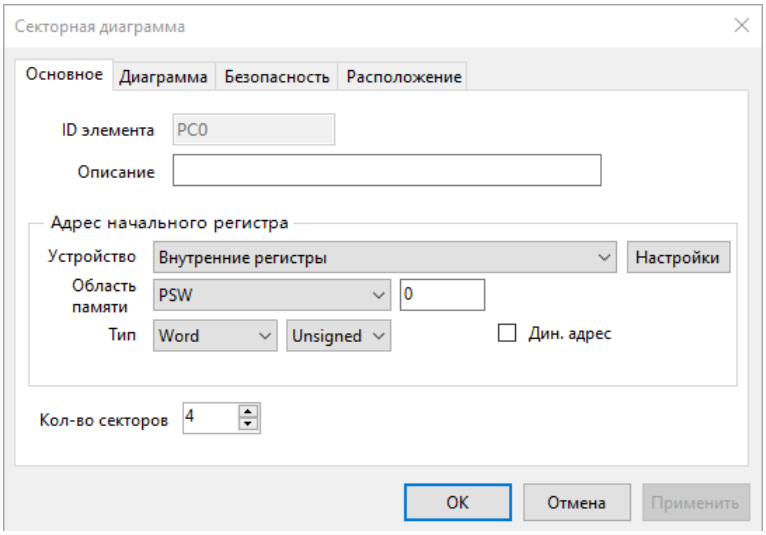


Рисунок 10.143 – Секторная диаграмма

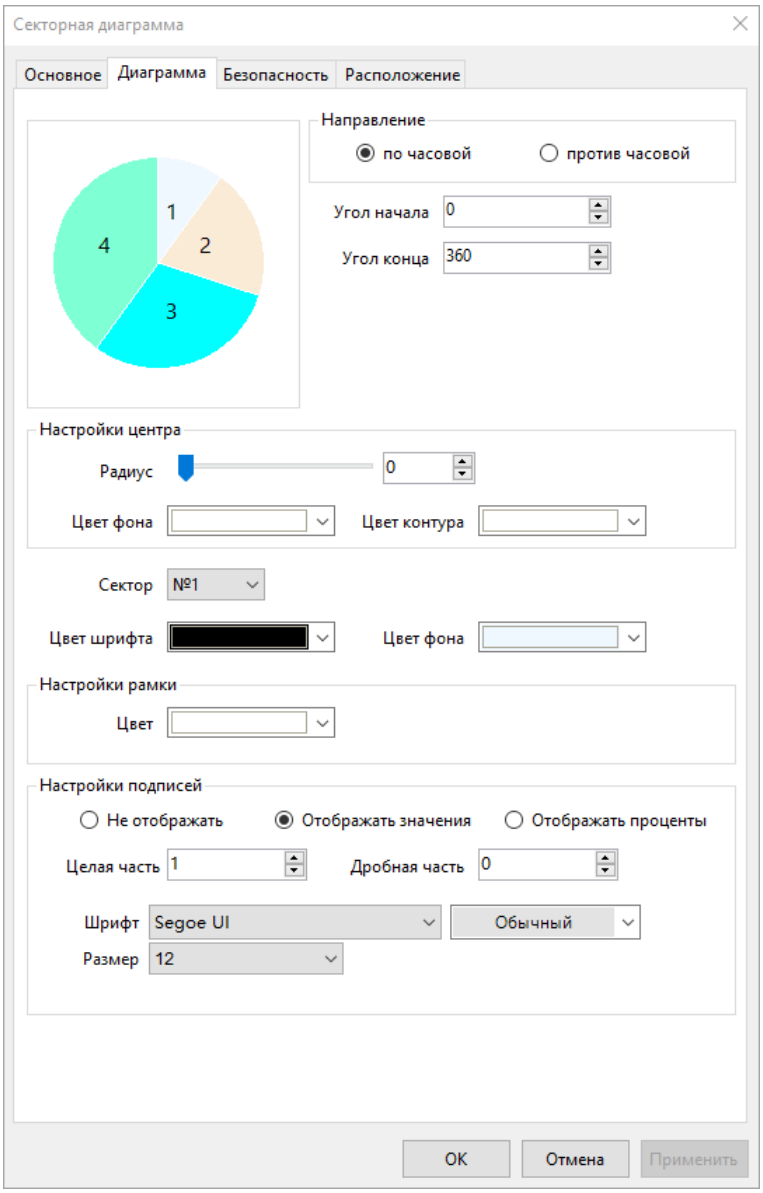
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.54 – Настройки элемента Секторная диаграмма

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.144 – Вкладка Основное**	
Адрес начального регистра	Адрес начального регистра, в котором хранятся значения, отображаемые элементом
Кол-во секторов	Кол-во значений, отображаемых элементов. Элемент будет использовать соответствующее количество регистров, начиная с указанного (в зависимости от типа параметра)

Продолжение таблицы 10.54

Настройка	Описание
Вкладка **Диаграмма**  **Рисунок 10.145 – Вкладка Диаграмма**	
Направление	Направление расположения порядка секторов элемента – по часовой стрелке или против часовой стрелки
Угол	Угол начала и конца элемента
Настройки центра	Настройки центральной части элемента: • радиус; • цвет заливки; • цвет контура.
Сектор	Для указанного сектора выбирается цвет фона и цвет текста
Настройки рамки	Цвет рамки элемента
Настройки подписей	Выбор режима отображения подписей (не отображать/отображать значения/ отображать проценты). Для подписей настраивается: • максимальное количество знаков целой и дробной части; • шрифт, его стиль и размер.

10.10 Рецепты

10.10.1 Настройка рецептов

Данное меню позволяет осуществить настройку рецептов – наборов значений, которые можно использовать для управления режимом работы оборудования и других задач. Рецепты создаются в рамках групп. Каждая группа описывает набор параметров, а каждый рецепт группы представляет собой набор конкретных значений этих параметров.

Рецепты сохраняются в виде файлов, размещаемых во встроенной памяти панели. Эти файлы не предназначены для непосредственного использования, поэтому их формат не документируется.

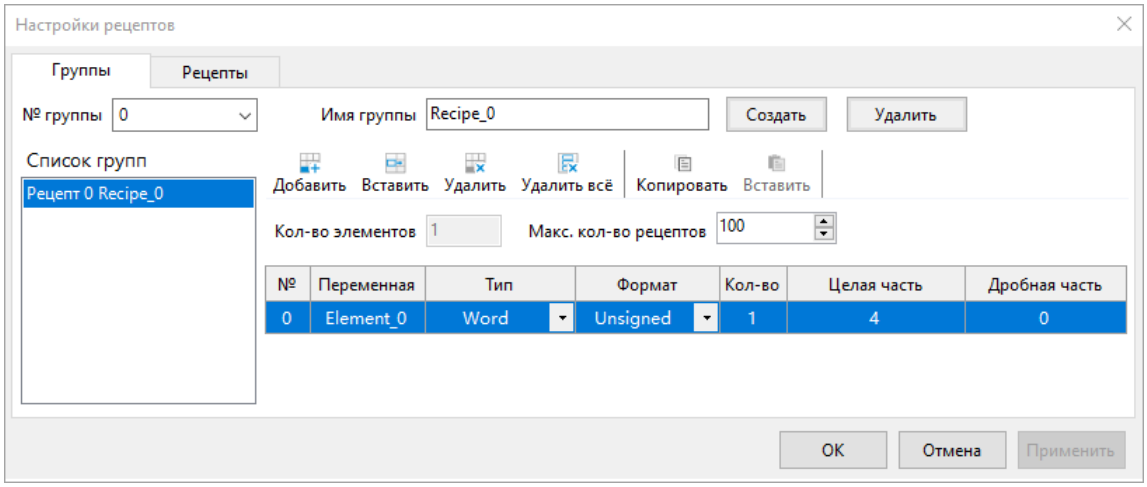
Для работы с рецептами используются элементы:

- **Таблица рецептов**– используется для отображения рецептов выбранной группы рецептов и их редактирования;
- **Чтение/запись рецепта**– используется для чтения значений файла рецепта в регистры панели и для записи значений регистров панели в файл рецепта;
- **Управление таблицей рецептов**– используется для выполнения команд, связанных с выделенной в данный момент строкой таблицы рецептов (например, удаления).

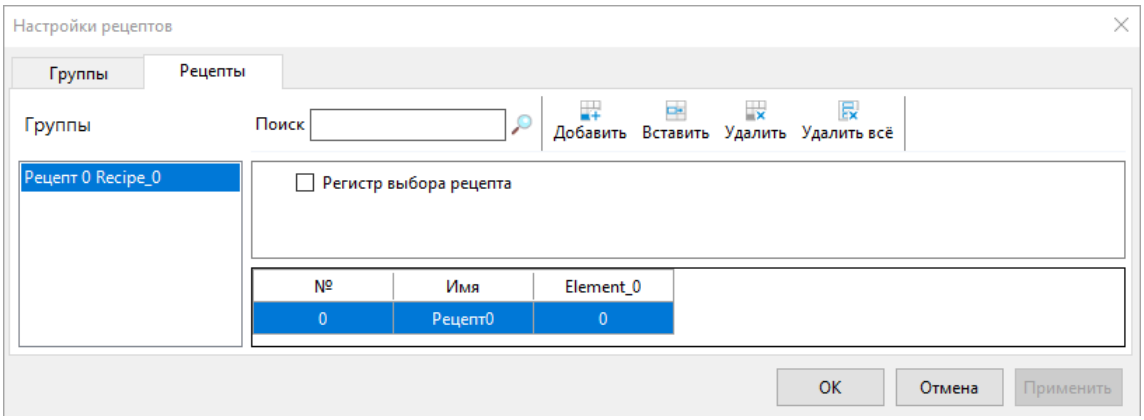
Также с рецептами связаны действия **Запись в рецепт** и **Чтение из рецепта**, которые можно настроить в элементах **Функциональная кнопка** и **Функциональная область**. Упомянутый выше элемент **Чтение/запись рецепта** реализован на базе этих действий.

Настройки рецептов описаны в таблице:

Таблица 10.55 – Настройки рецептов

Настройка	Описание
Вкладка **Группы**  **Рисунок 10.146 – Вкладка Группы**	
Список групп	Список созданных групп рецептов. Для создания новой группы следует ввести имя и нажать кнопку **Создать**. Выбор группы происходит с помощью нажатия на ее имя в списке **ЛКМ** или с помощью выпадающего списка номеров групп. Для удаления выбранной группы следует нажать кнопку **Удалить**
Макс. кол-во рецептов	Максимальное количество рецептов, которое может быть создано в рамках данной группы (не более **1000**)
Кол-во элементов	Количество параметров данной группы рецептов

Продолжение таблицы 10.55

Настройка	Описание
Содержимое	С помощью кнопок можно создавать и удалять параметры группы. Для каждого параметра указывается имя (не должно содержать пробелов), тип и формат данных. Для числовых значений указывается максимальное количество знаков целой и дробной части, для строковых значений – количество объектов, занимаемых строкой (объект зависит от типа строки: для Word_String он представляет собой WORD и т. д.)
Вкладка **Рецепты**  **Рисунок 10.147 – Вкладка Рецепты**	
Содержимое	С помощью кнопок можно создавать и удалять рецепты. Для каждого рецепта указывается имя (не должно содержать пробелов) и значения его параметров
Регистр выбора рецепта	В случае установки галочки можно указать адрес регистра, используемого для выбора рецепта данной группы рецептов. Выбор может осуществляться по номеру рецепта или его имени. В случае отсутствия галочки выбор рецепта происходит по его номеру с помощью системного регистра **SPFW256**

10.10.2 Таблица рецептов

Элемент **Таблица рецептов** используется для отображения и редактирования значений рецептов выбранной группы в табличном виде.

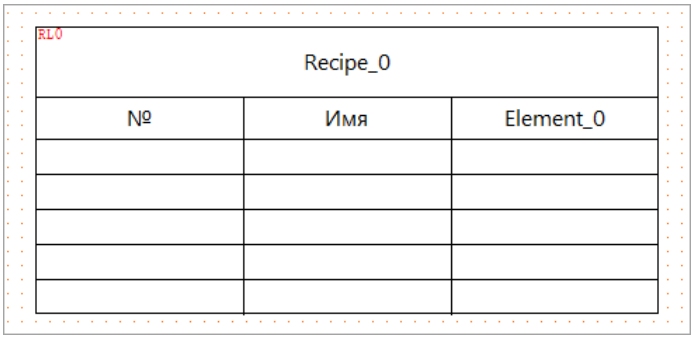


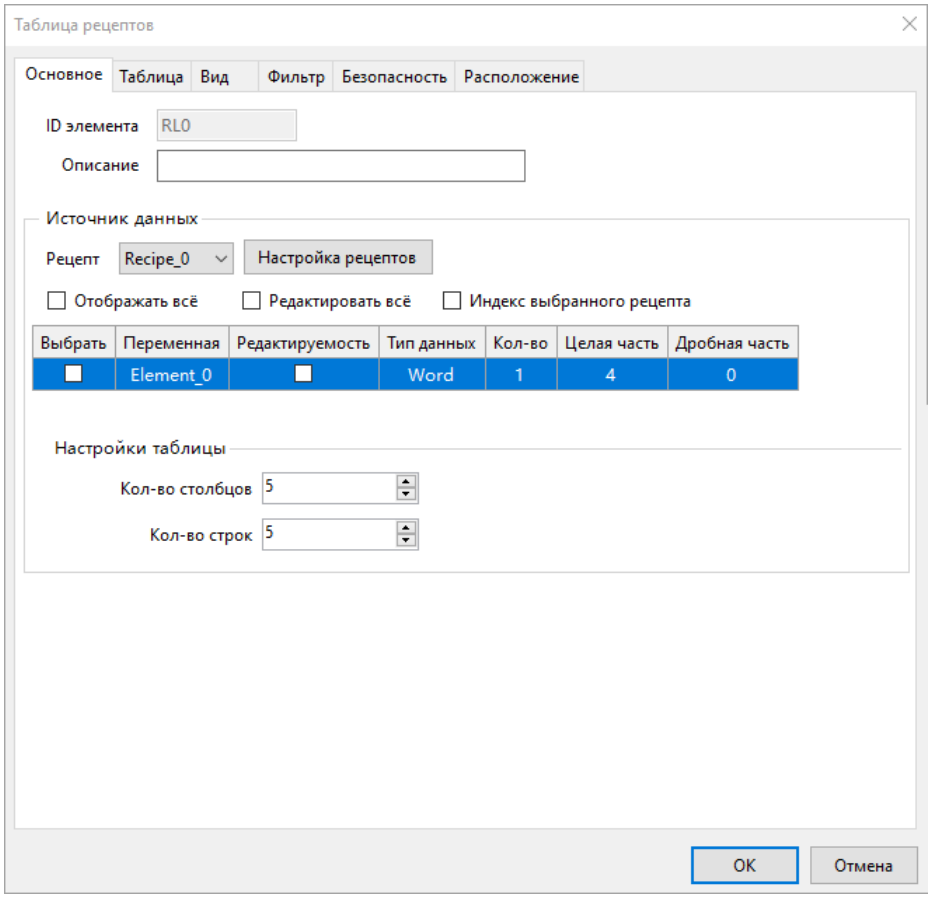
Рисунок 10.148 – Таблица рецептов

Для выполнения операций над таблицей используется элемент **Управление таблицей рецептов**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.56 – Настройки элемента Таблица рецептов

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.149 – Вкладка Основное**	
Источник данных	Группа рецептов, привязанная к таблице. Нажатие на кнопку **Настройка рецептов** приводит к открытию соответствующего меню
Отображать всё	В случае установки галочки в содержимом таблицы устанавливаются галочки **Выбрать** для всех параметров рецепта. В случае снятия галочки для всех параметров снимаются галочки **Выбрать**
Редактировать всё	В случае установки галочки в содержимом таблицы устанавливаются галочки **Редактируемость** для всех параметров рецепта. В случае снятия галочки для всех параметров снимаются галочки **Редактировать**
Индекс выбранного рецепта	В случае установки галочки номер выбранного в таблице рецепта будет записан в системный регистр **SPFW256**. Это упростит совместное использование таблицы с элементом **Чтение/запись рецепта**
Содержимое	Для каждого параметра группы рецептов отображается его имя, которое задаётся в настройках рецепта. Для числовых значений отображается максимальное количество знаков целой и дробной части, для строковых значений – количество объектов, занимаемых строкой (объект зависит от типа строки: для Word_String он представляет собой WORD и т. д.). Галочка **Выбрать** определяет отображение столбца параметра в таблице. Галочка **Редактируемость** определяет возможность редактирования значения данного параметра. Для ввода значения следует нажать на соответствующую ячейку таблицы
Настройки таблицы	Общее количество записей (не более 65536), отображаемых в таблице, и количество записей на одной странице (не более 65536). Для прокрутки таблицы используется полоса прокрутки. В режиме эмуляции для прокрутки таблицы нужно навести на неё курсор и использовать колёсико мыши

Продолжение таблицы 10.56

Настройка	Описание
Вкладка **Таблица**	

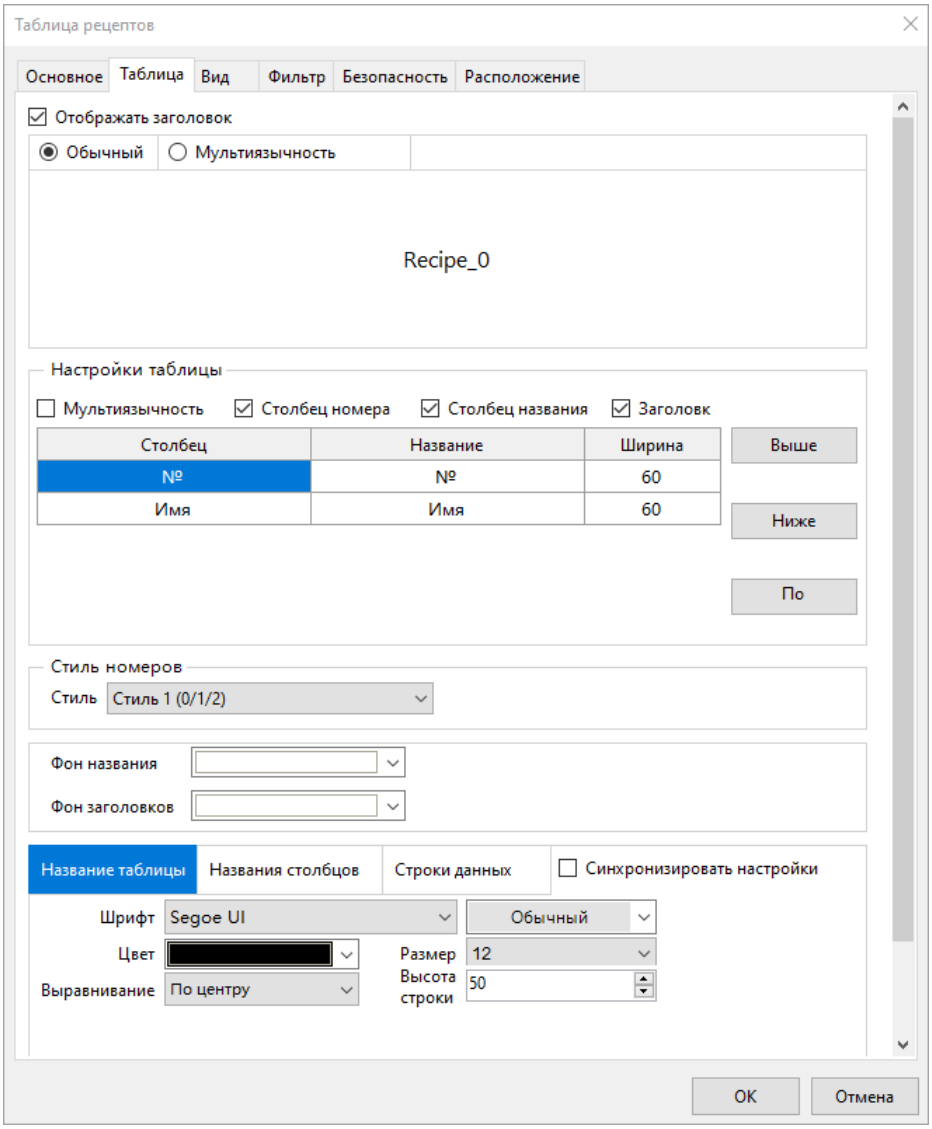


Рисунок 10.150 – Вкладка Таблица

Отображать заголовок	В случае установки галочки над таблицей отображается заголовок, который может быть задан статическим текстом или текстом локализации
Настройки таблицы	Для каждого столбца таблицы указывается текст заголовка и ширина. В случае установки галочки **Мультиязычность** в качестве текстов заголовков могут быть использованы тексты локализации В случае снятия галочки **Столбец номера** в таблице не будет отображаться столбец номера рецепта В случае снятия галочки **Столбец названия** в таблице не будет отображаться столбец названия параметра рецепта В случае снятия галочки **Заголовки** в таблице не будет отображаться строка названий заголовков столбцов
Стиль номеров	Выбор стиля отображения номеров рецептов. В стиле 1 номер отображается только числом, а в стиле 2 – числом с предваряющим его текстом **Рецепт**
Фон названия	Цвет фона названия таблицы
Фон заголовков	Цвет фона строки заголовков таблицы

Продолжение таблицы 10.56

Настройка	Описание
Шрифт	Выбор шрифта, его типа, цвета, размера, выравнивания текста и высоты строки для заголовка таблицы, названий столбцов и текстов строк. Галочка **Синхронизировать настройки** позволяет сделать настройки всех этих текстов одинаковыми
Вкладка **Вид**	

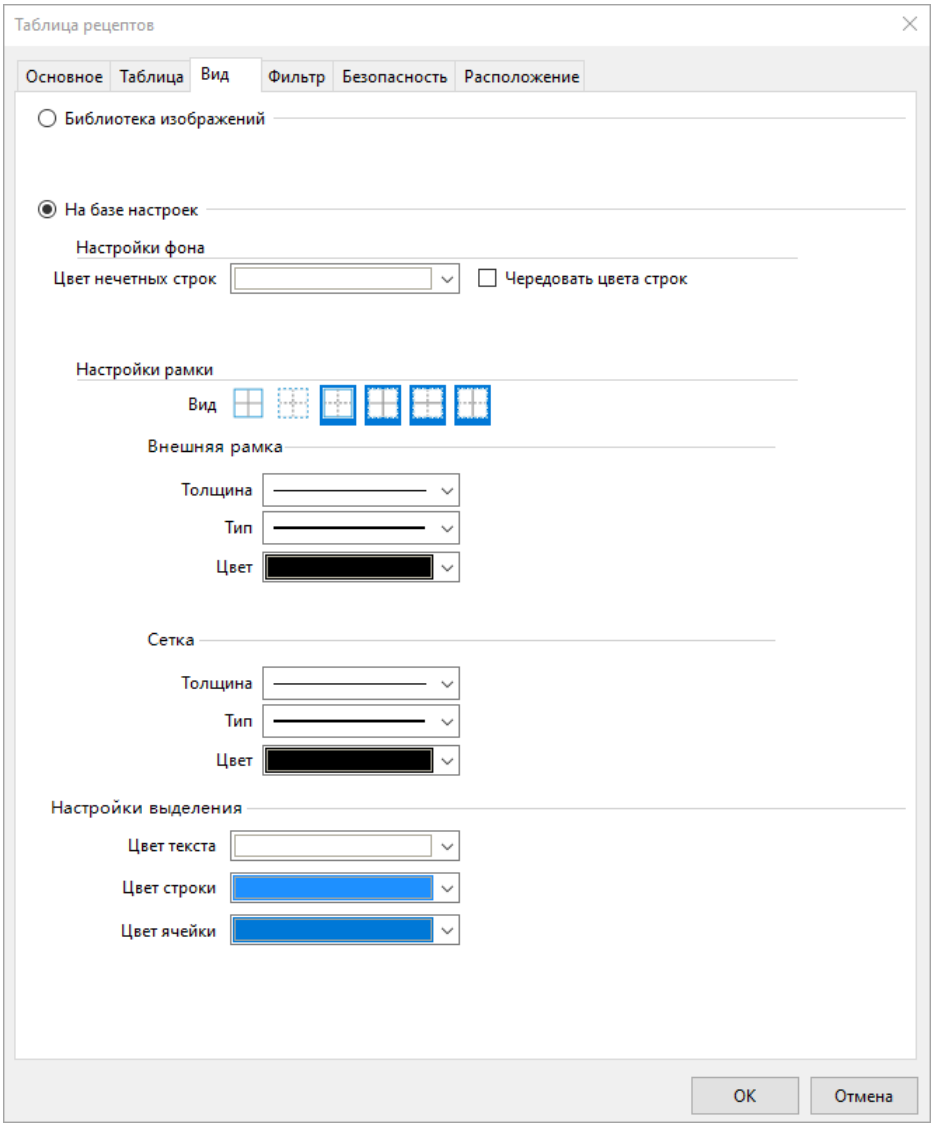


Рисунок 10.151 – Вкладка Вид

Библиотека изображений	Выбор изображения фона элемента из **Библиотеки изображений** (папка **DataForm**)
На базе настроек	В данном режиме внешний вид таблицы определяется заданными пользователем настройками, к которым относятся: • цвет фона строк таблицы (в случае установки галочки **Чередовать цвета строк** – указываются два цвета); • внешний вид рамки таблицы; • тип, толщина и цвет внешней рамки таблицы; • тип, толщина и цвет сетки таблицы.
Настройки выделения	Цвета текста и фона выделенной ячейки, цвет фона остальных ячеек соответствующей строки
Вкладка **Фильтр**	

Продолжение таблицы 10.56

Настройка	Описание
Скриншот	В случае установки галочки можно выбрать бит, по переднему или заднему фронту которого будет происходить экспорт видимого содержимого таблицы на USB-накопитель в виде файла формата .png. В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\png**
Использовать фильтр	В случае установки галочки пользователь выбирает тип фильтра, применяемый к таблице в течение времени, пока заданный бит имеет значение **TRUE**
Тип фильтра: по вхождению	Искомый фрагмент задается строкой или путём указания адреса регистра. Данный фильтр применяется как к параметрам строкового типа, так и к параметрам числового типа. При фильтрации по вхождению отображаются те строки таблицы, в состав значения хотя бы одного параметра которых входит данный текстовый фрагмент
Тип фильтра: по числу	Число задается константой или путём указания адреса регистра. Данный фильтр применяется только к параметрам числового типа

10.10.3 Чтение/запись рецепта

Элемент **Чтение/запись рецепта** используется для чтения значений файла рецепта в регистры панели и для записи значений регистров панели в файл рецепта.

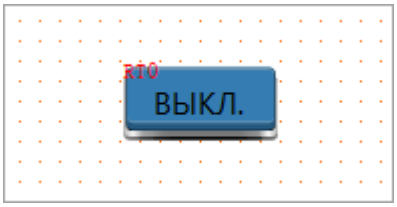
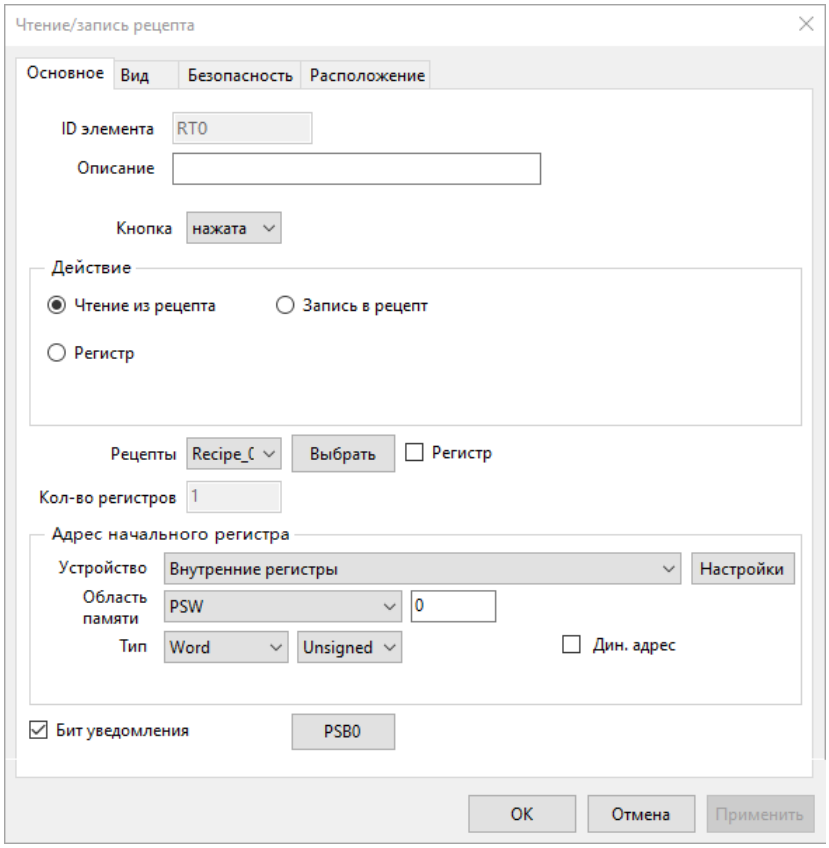


Рисунок 10.152 – Чтение/запись рецепта

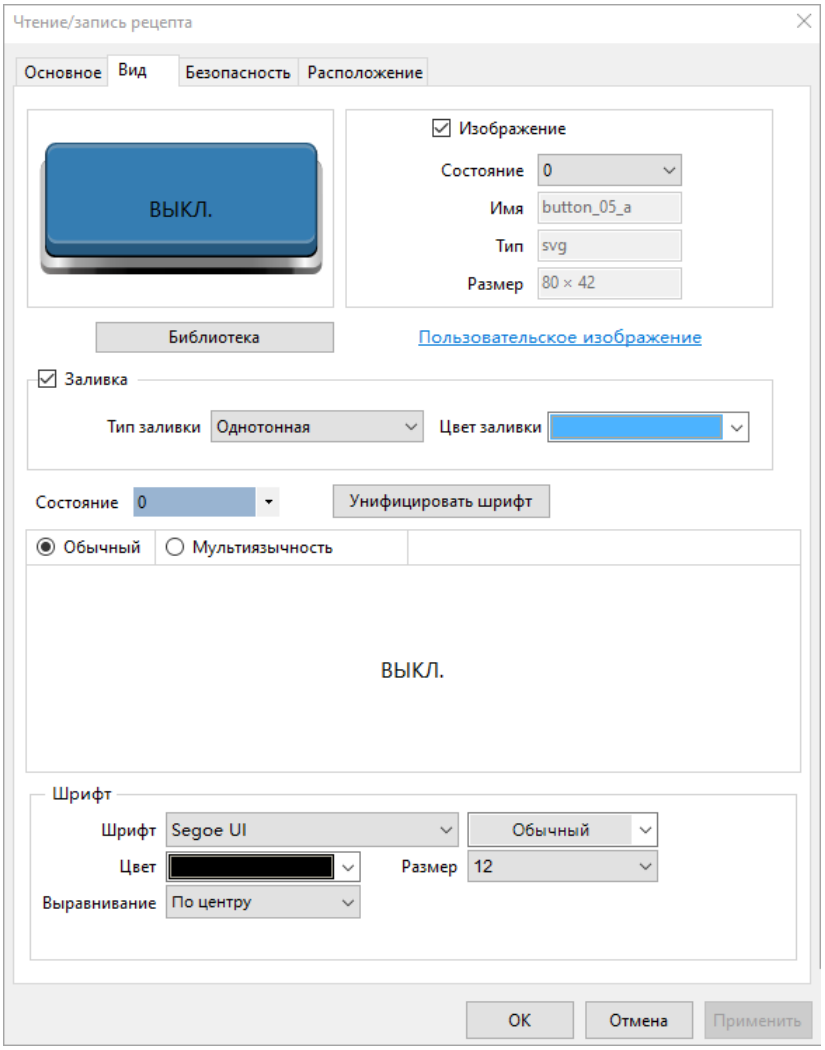
Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.57 – Настройки элемента Чтение/запись рецепта

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.153 – Вкладка Основное**	
Кнопка	Событие, при котором будет выполнено добавленное действие – нажатие или отпускание кнопки
Действие	Возможные варианты: • Чтение из рецепта; • Запись в рецепт; • Регистр – для каждой из операций можно указать отдельный бит, по переднему или заднему фронту которого она будет выполняться.
Группа	Выбор группы рецептов, с которой будет работать элемент. Номер рецепта определяется значением системного регистра **SPFW256** или, в случае установки в меню настройки рецептов галочки **Регистр выбора рецепта** – значением указанного пользователем регистра
Адрес начального регистра	Адрес начального из группы регистров, в которые будут записаны значения из файла рецепта (для действия **Чтение из рецепта**) или адрес начального из группы регистров, значения которых будут записаны в рецепт (для действия **Запись в рецепт**)
Бит уведомления	В случае установки галочки после завершения операции указанный бит устанавливается в значение **TRUE**

Продолжение таблицы 10.57

Настройка	Описание
Вкладка **Вид**  **Рисунок 10.154 – Вкладка Вид**	
Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния кнопки: **0** – отпущена, **1** – зажата Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущий шрифт применяется ко всем состояниям элемента
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.10.4 Управление таблицей рецептов

Элемент **Управление таблицей рецептов** используется для выполнения команды над выделенной строкой таблицы рецептов.

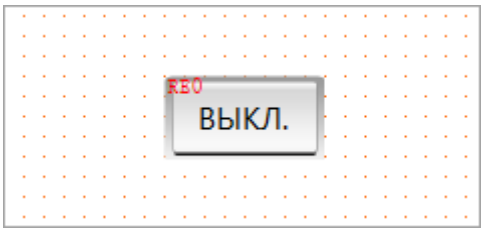
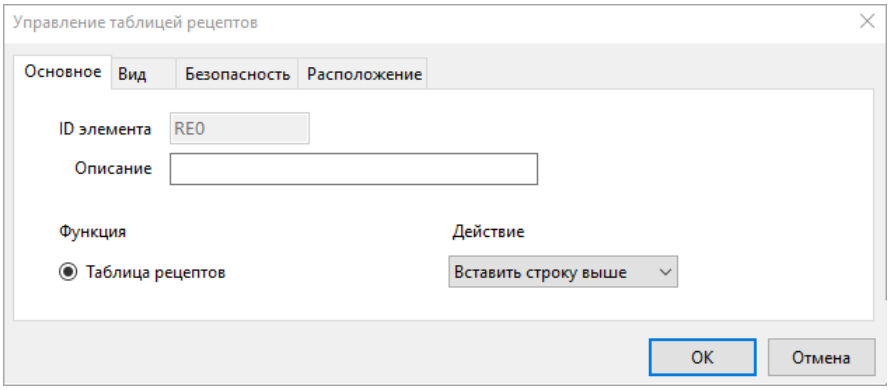


Рисунок 10.155 – Управление таблицей рецептов

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.58 – Настройки элемента Управление таблицей рецептов

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.156 – Вкладка Основное**	
Действие	Возможные варианты: • вставить новый рецепт выше выделенной строки; • вставить новый рецепт ниже выделенной строки; • удалить выделенный рецепт; • создать новый рецепт со значениями выделенной строки (скопировать рецепт).
Аудиофайл	Только для модификаций СП210-07-Р, СП210-10-Р и СП210-15-Р. Выбор аудиофайла, который будет воспроизводиться при нажатии на элемент

Продолжение таблицы 10.58

Настройка	Описание
Вкладка **Вид**	

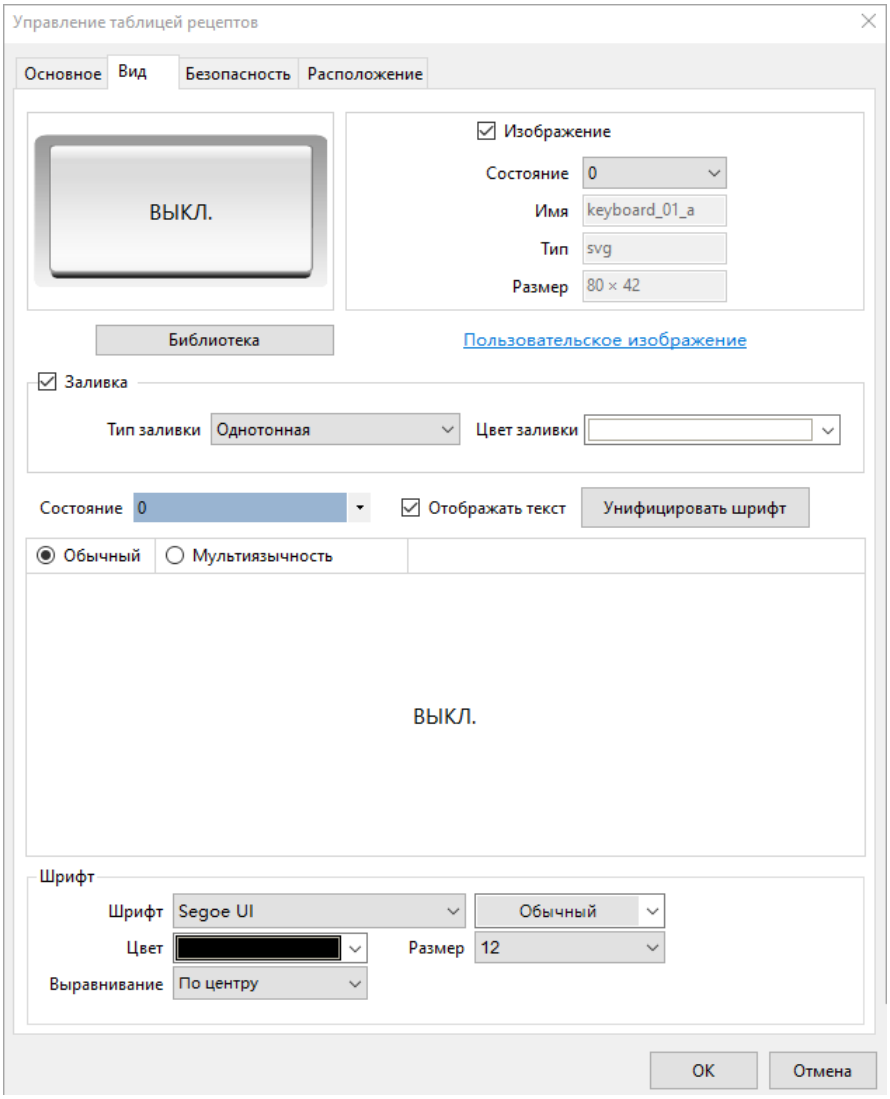


Рисунок 10.157 – Вкладка Вид

Изображение	В случае установки галочки в качестве фона элемента будет отображаться изображение, выбранное пользователем с помощью кнопки **Библиотека** из библиотеки изображений или с помощью кнопки **Пользовательское изображение** из изображений, доступных на ПК
Заливка	Тип (однотонная или градиент) и цвет заливки фона элемента. Доступно только для изображений формата .svg, поддерживающих соответствующий параметр
Состояние	Выбор состояния кнопки: **0** – отпущена, **1** – зажата Все остальные настройки вкладки применяются к выбранному в данный момент состоянию
Отображать текст	В случае установки галочки можно указать статический текст или текст локализации, который будет отображаться в элементе
Унифицировать шрифт	В случае нажатия кнопки текущий шрифт применяется ко всем состояниям элемента
Шрифт	Выбор шрифта, его типа, цвета, размера и выравнивания текста

10.11 Журнал действий

10.11.1 Настройка журнала действий

Данное меню используется для настройки журнала действий, в котором сохраняются действия оператора – авторизация, нажатия на элемент, ввод значений и т. д.

Данная информация может помочь при разборе причин возникновения в работе системы внештатных ситуаций.

Журнал действий сохраняется в файл.

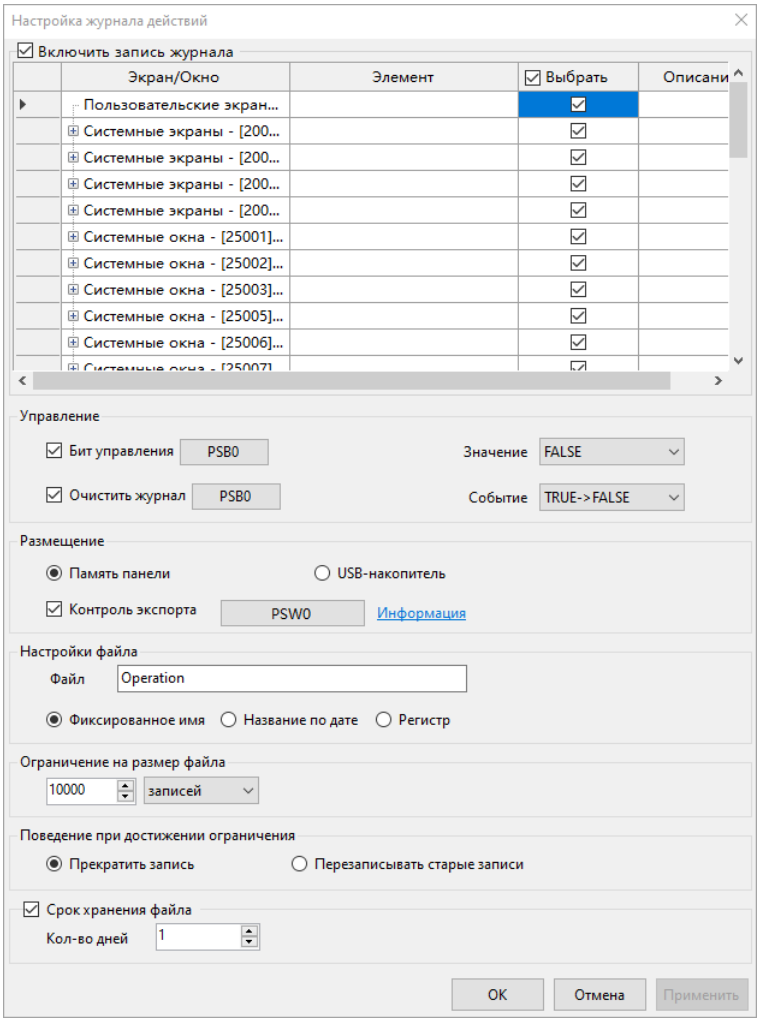


Рисунок 10.158 – Настройка журнала действий

Для его отображения используется элемент **Отображение журнала действий**.

Настройки журнала действий описаны в таблице 10.59.

Список сохраняемых действий описан в таблице 10.60.

Таблица 10.59 – Настройки журнала действий

Настройка	Описание
Включить запись журнала	В случае установки галочки панель сохраняет журнал действий оператора. В таблице галочками помечаются элементы, действия с которыми должны фиксироваться в журнале
Управление	В случае установки галочки **Бит управления** можно указать адрес бита и состояние, которое будет соответствовать записи журнала. Переключение бита в противоположное состояние прекратит запись в журнал В случае установки галочки **Очистить журнал** можно указать адрес бита, по переднему или заднему фронту которого будет происходить очистка файла журнала
Размещение	Выбор места сохранения файла журнала: в памяти панели или на подключенном USB-накопителе (рекомендуется использовать хранение на подключенном USB-накопителе для снижения износа встроенной памяти панели). В случае сохранения файла в памяти панели доступна галочка **Контроль экспорта**, которая позволяет указать адрес первого из 3 регистров, в которые будет записываться информация экспорта. Например, указан регистр **PSW0**, тогда: **PSW0** – команда экспорта. При записи значения **1** происходит экспорт журнала действий оператора на USB-накопитель. При записи значения **2** помимо экспорта выполняется удаление файла из памяти панели. Для выполнения новой команды в регистр предварительно должно быть записано значение **0**. **PSW1** – в этот регистр записывается значение прогресса операции экспорта в процентах. Значение **100** соответствует завершению экспорта. **PSW2** – в указанный регистр будет записан результат успешного выполнения экспорта или код ошибки: • 0 – экспорт в процессе; • 1 – экспорт завершен успешно; • 2 – USB-накопитель не подключен; • 3 – недостаточно памяти; • 4 – некорректный путь к файлу; • 5 – ошибка формата файла; • 6 – в процессе экспорта произошла ошибка. В режиме эмуляции: • для варианта **USB**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\history** с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf • для варианта **Память панели**: файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\db\history** в бинарном формате. Для возможности просмотра следует использовать команду экспорта (см. выше), которая сформирует в директории **\Temp\Run\storage\udisk\history** файл с расширением .xjdb. При нажатии на этот файл запускается утилита конвертации в формат .csv или .pdf.
Настройки файла	Название файла может быть задано как: • фиксированное; • формируемое на основе текущей даты (пример: **2025_10_08**) – соответственно, каждый день будет создаваться новый файл; • формируемое на основе содержимого указанных регистров (в регистрах должна быть размещена строка с именем файла в кодировке ASCII, без использования символов кириллицы и спецсимволов. Максимальное количество регистров – 256).

Продолжение таблицы 10.59

Настройка	Описание
Ограничение на размер файла	Ограничение на размер файла истории тревог, задаваемое в виде количества записей или мегабайт. Максимальное значение для каждого из вариантов – **65535** (при выборе мегабайт нужно убедиться, что в памяти панели или на USB-накопителе достаточно свободного места)
Поведение при достижении ограничения	Данная настройка определяет поведение при достижении файлом указанного ограничения: прекращение записи в файл или перезапись самых старых записей
Срок хранения файла	В случае установки галочки можно указать интервал времени с момента создания файла (в виде количества дней, не более **1000**), по истечению которого файл будет удален. Обычно используется совместно с названием файла, формируемым на основе текущей даты, чтобы сохранять файлы истории за последние N дней

Таблица 10.60 – Список сохраняемых действий

Действие	Описание
Изменение бита	В информации указывается прошлое и новое состояние бита
Ввод числа	В информации указывается прошлое и новое значение числа
Ввод строки	В информации указывается прошлое и новое значение строки в виде HEX-дампа
Нажатие	
Отпускание	
Чтение из рецепта	В информации указывается название группы рецептов
Запись в рецепт	В информации указывается название группы рецептов

10.11.2 Отображение журнала действий

Элемент **Отображение журнала действий** используется для отображения истории действий оператора в табличном виде.

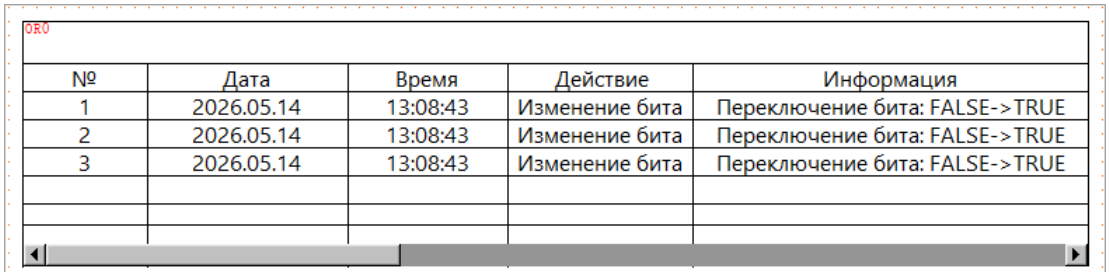
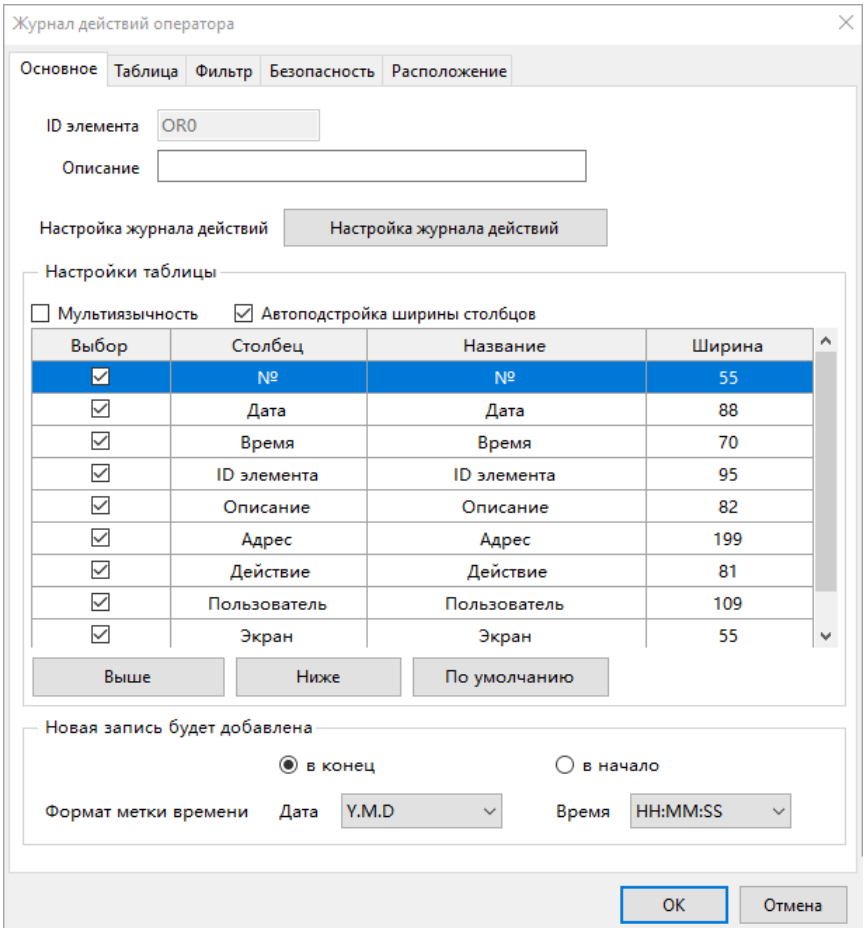


Рисунок 10.159 – Отображение журнала действий

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.61 – Настройки элемента Отображение журнала действий

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.160 – Вкладка Основное**	
Настройка журнала действий	Открытие соответствующего меню
Мультиязычность	В случае установки галочки в качестве названий столбцов могут быть использованы тексты локализации
Автоподстройка ширины столбцов	В случае установки галочки ширина столбцов таблицы настраивается автоматически на основании ширины заголовков
Настройки таблицы	С помощью галочек пользователь указывает столбцы, которые будут отображаться в таблице: • Номер записи; • Дата совершения действия; • Время совершения действия; • Идентификатор элемента, над которым выполнено действие; • Описание элемента, указанное в настройках элемента на вкладке **Основное**; • Адреса бита или регистра, указанный в настройках элемента на вкладке **Основное**; • Действие, совершенное над элементом; • Имя пользователя, совершившего действие • Название экрана или окна, на котором размещён элемент; • Информация, связанная с действием. Для каждого столбца настраивается заголовок и ширина

Продолжение таблицы 10.61

Настройка	Описание
Новая запись будет добавлена	Данная настройка определяет, в какую строку будет добавлена новая запись: перед всеми существующими (сверху) или после всех существующих (снизу)
Формат метки времени	Возможные варианты формата для даты: • M.D.Y; • D.M.Y; • Y.M.D. Возможные варианты формата для времени: • HH:MM:SS; • HH:MM; • HH:MM:SS:MS.

Вкладка **Таблица**

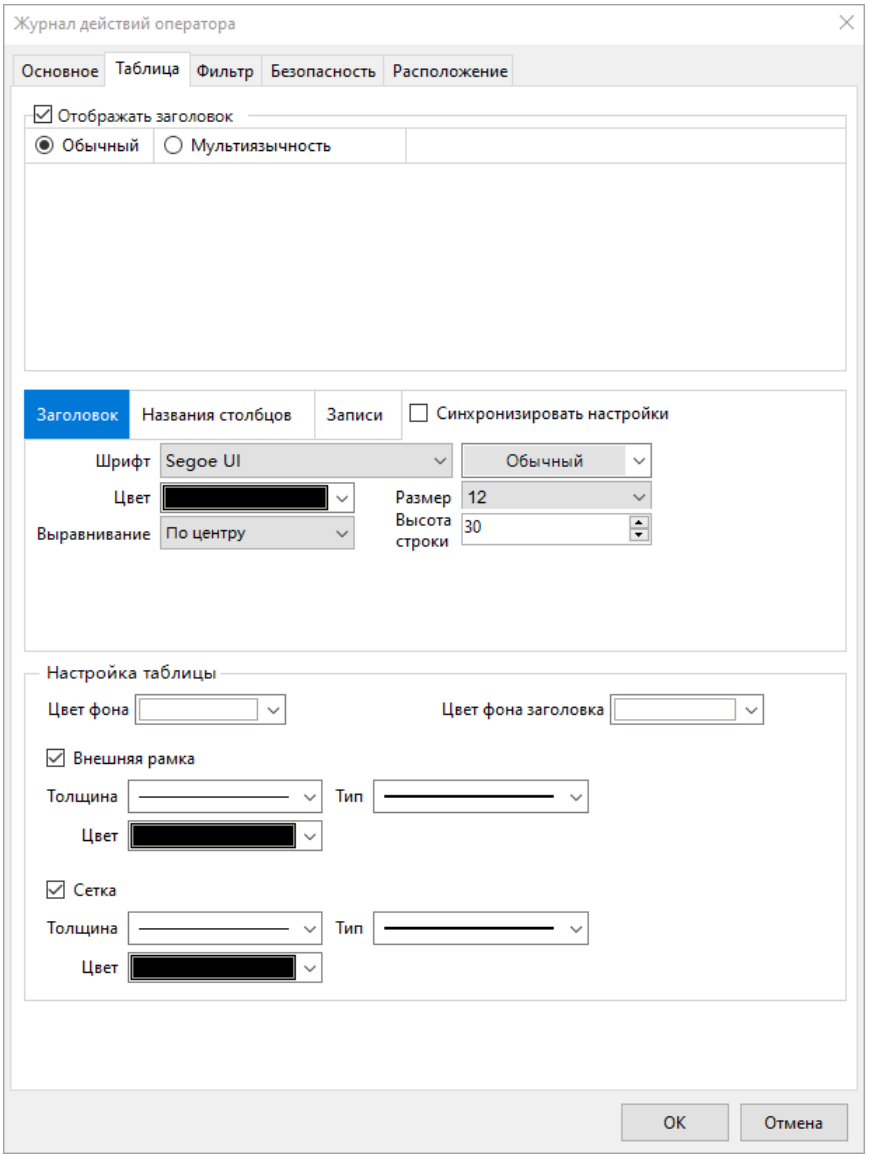


Рисунок 10.161 – Вкладка Таблица

Отображать заголовок	В случае установки галочки над таблицей отображается заголовок, который может быть задан статическим текстом или текстом локализации

Продолжение таблицы 10.61

Настройка	Описание
Шрифт	Выбор шрифта, типа, цвета, размера, выравнивания текста и высоты строки для заголовка таблицы, названий столбцов и текстов строк. Галочка **Синхронизировать настройки** позволяет сделать настройки всех текстов одинаковыми
Настройка таблицы	Цвета фона названия таблицы и строк
Внешняя рамка	Тип, толщина и цвет линий внешней рамки таблицы
Сетка	Тип, толщина и цвет линий сетки таблицы
Вкладка **Фильтр**	
Скриншот	В случае установки галочки можно выбрать бит, по переднему или заднему фронту которого будет происходить экспорт видимого содержимого таблицы на USB-накопитель в виде файла формата .png. В режиме эмуляции файл сохраняется в директории установки конфигуратора по пути **\Temp\Run\storage\udisk\png**
Использовать фильтр	В случае установки галочки пользователь выбирает тип фильтра, применяемый к таблице в течение времени, пока заданный бит имеет значение **TRUE**
Тип фильтра: дата	Дата задается константой или путём указания адреса регистра, начиная с которого будут размещены три регистра даты, содержащие год, месяц и день (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: интервал времени	Интервал времени задается двумя константами или путём указания адреса регистра, начиная с которого будут размещены 12 регистров – 6 для метки времени начала интервала и 6 для метки времени конца интервала. Порядок разрядов метки времени: год, месяц, день, часы, минуты, секунды (в виде беззнаковых значений в десятичной системе счисления)
Тип фильтра: регистр	В данном случае тип фильтра определяется динамически на основании значения указанного регистра: • 0 – дата; • 1 – интервал времени. Для любого из типов указывается адрес регистра, начиная с которого размещаются параметры фильтра (см. описание выше)

10.12 Мультимедия

10.12.1 Отображение видеопотока (RTSP)

Элемент **Отображение потока (RTSP)** используется для отображения видеопотока камер, поддерживающих протокол RTSP. Данный элемент поддерживается только расширенными модификациями панелей (-P).

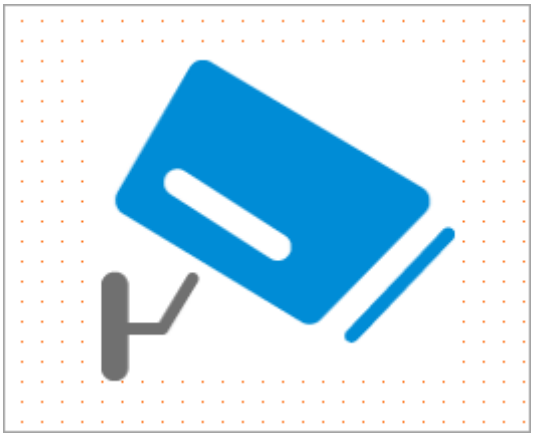


Рисунок 10.162 – Отображение видеопотока (RTSP)

На каждом экране/окне панели может быть размещён только один такой элемент, и на дисплее панели не может одновременно отображаться несколько окон с данными элементами.

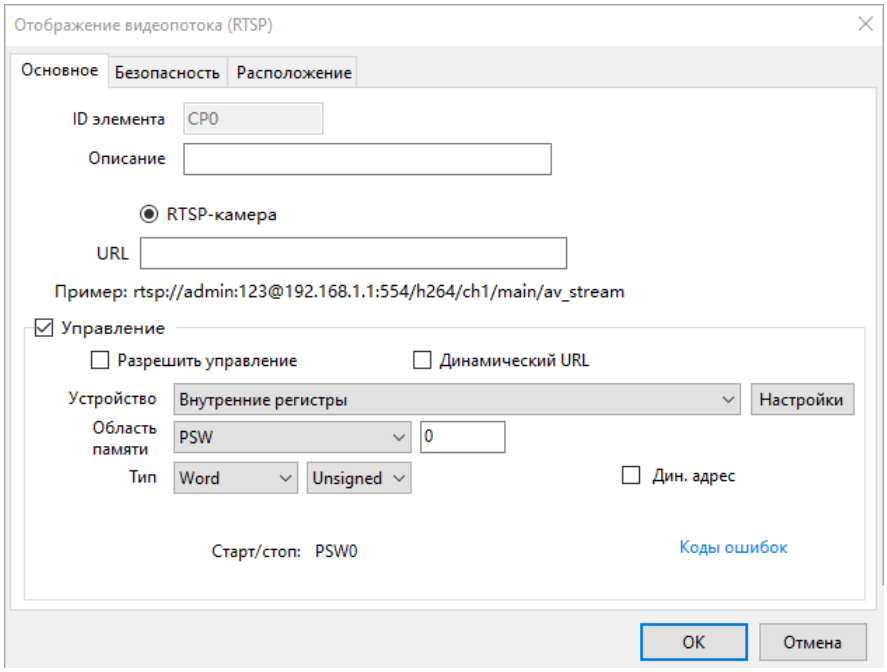
Максимальный размер элемента для модификации СП210-15-P составляет **1264x720**, для остальных модификаций – равен разрешению экрана панели.

Поддерживаемый видеокодек: **H264**.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.62 – Настройки элемента Отображение видеопотока (RTSP)

Настройка	Описание
Вкладка **Основное**  **Рисунок 10.163 – Вкладка Основное**	
URL	URL видеопотока. Пример: rtsp://admin:123456@10.2.11.181:554/H264?ch=1&subtype=1 где в рамках примера: • admin и 123456 – логин и пароль; • 10.2.11.181 – IP-адрес видеокамеры; • 554 – используемый порт (по умолчанию для протокола RTSP используется порт 554); • H264?ch=1&subtype=1 – уникальная для каждой камеры строка, в которой указан используемый кодек, канал, поток и т. д. Более подробная информация о ссылках должна быть указана в руководстве на IP-камеру. Из-за аппаратных и программных ограничений панели не гарантируется отображение любого видеопотока (например, отображение видеопотока высокого разрешения, видеопотока с высоким битрейтом и т. д.). Код ошибки записывается в системный регистр **SPSW688**
Управление	В случае установки галочки **Управление** можно указать адрес регистра, который будет использоваться для управления воспроизведением: • 1 – начать воспроизведение; • 0 – прекратить воспроизведение. Команда **Пауза** используется для приостановки воспроизведения: • 1 – остановить воспроизведение; • 0 – продолжить воспроизведение. Адрес регистра паузы равен адресу регистра управления + 1. Динамический URL позволяет изменить URL видеопотока в процессе работы панели. Под строку URL выделяется 64 регистра (кодировка – ASCII). Для изменения URL предварительно следует прекратить воспроизведение. Адрес начального регистра динамического URL равен адресу регистра управления + 2.

10.12.2 Воспроизведение видеофайлов

Элемент **Воспроизведение видеофайлов** используется для отображения видеофайлов. Данный элемент поддерживается только модификацией СП210-15-Р.

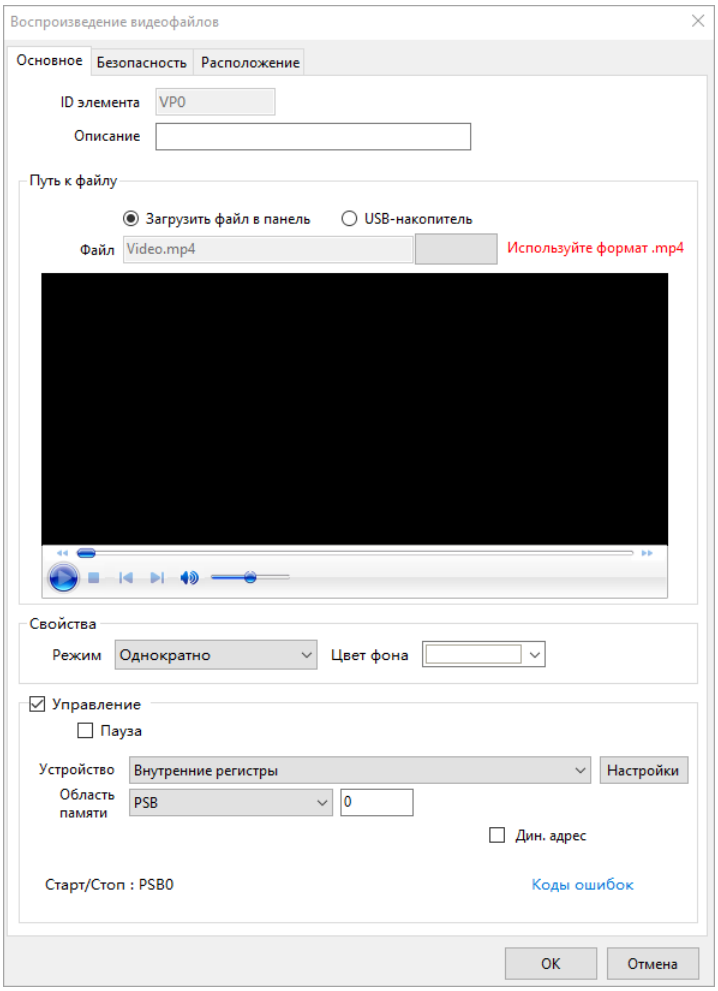


Рисунок 10.164 – Воспроизведение видеофайлов

Особенности использования:

- На каждом экране/окне панели может быть размещён только один элемент **Воспроизведение видеофайлов**;
- На дисплее панели не может одновременно отображаться несколько окон с элементами **Воспроизведение видеофайлов**;
- Данный элемент не может отображаться одновременно с элементом **Отображение видеопотока (RTSP)**.

Максимальный размер элемента составляет **1280x720**.

Требования к видеофайлам:

- формат **.mp4**;
- видеокодек **H264**;
- размер файла загружаемого в память панели не должен превышать 10 Мб;
- расчётное отображаемое количество кадров в секунду (FPS) – 20. Если разрешение файла превышает 800x480, то расчётное FPS – 10.

Типовые настройки элемента описаны в разделе 10.1.

Уникальные настройки элемента описаны в таблице:

Таблица 10.63 – Настройки элемента Воспроизведение видеофайлов

Настройка	Описание
Вкладка **Основное**	
Путь к файлу	Место размещения файла. Можно выбрать файл, который будет загружен в панель или указать директорию USB-накопителя, в которой файл должен быть размещен (ожидаемое имя файла формируется панелью автоматически на основе последнего выбранного в элементе файла; название по умолчанию – **Video.mp4**). Код ошибки воспроизведения записывается в системный регистр **SPSW689**
Свойства	Цвет фона и режим воспроизведения файла. Доступные варианты: • **Однократно** – выбранный файл будет воспроизведен однократно; • **Циклически** – выбранный файл будет воспроизводиться по кругу; • **Все файлы – однократно** – все файлы, размещенные в указанной директории USB-накопителя, будут последовательно воспроизведены однократно; • **Все файлы – циклически** – все файлы, размещенные в указанной директории USB-накопителя, будут последовательно воспроизведены, после чего воспроизведение начнется с первого файла и будет продолжаться циклически.
Управление	В случае установки галочки **Управление** можно указать адрес бита, который будет использоваться для управления воспроизведением: • TRUE – начать воспроизведение; • FALSE – прекратить воспроизведение. Команда **Пауза** используется для приостановки воспроизведения: • TRUE – остановить воспроизведение; • FALSE – продолжить воспроизведение. Адрес бита паузы равен адресу биту управления +1

10.12.3 Коды ошибок

В случае ошибки воспроизведения видеопотока код ошибки записывается в системный регистр **SPSW688** (для элемента **Отображение видеопотока**) или **SPSW689** (для элемента **Воспроизведение видеофайлов**).

Список кодов ошибок приведен в таблице:

Таблица 10.64 – Коды ошибок воспроизведения видеопотока

Значение	Описание
0xFF	Ошибка в адресе RTSP или имени видеофайла
0xFE	Ошибка получения потока медиаданных
0xFD	Ошибка поиска видеопотока
0xFC	Ошибка поиска аудиопотока
0xFB	Ошибка поиска декодера
0xFA	Ошибка открытия декодера
0xF9	Ошибка закрытия декодера
0xF8	Видео содержит B-кадры (**B-frames**)
0xF7	Исключение при декодировании
0xE0	Очередь воспроизведения пуста

11 Настройка обмена с другими устройствами

11.1 Поддерживаемые протоколы

Панели СП210 поддерживают следующие протоколы обмена с другими устройствами:

Таблица 11.1 – Протоколы обмена, поддерживаемые панелями СП210 по COM-порту

Протокол	Режим	Комментарий
Modbus RTU	Master/Slave	
Modbus ASCII	Master	
Пользовательский протокол	Master/Slave	Пользовательский протокол обмена реализуется с помощью макросов
Mitsubishi	Master	Поддерживаются линейки: • FX (FX0, FX1, FX1S/3S, FX0N/1N/2N, FX3SA-14MR-CM, FX2); • FX3 (FX3U, FX3G FX3GA); • FX5U; • Q/L; • Q02H (Q02, Q02H, L02). Для линеек FX, FX3, FX5 поддерживается только режим контроля четности **EVEN**
Delta	Master	Поддерживаются линейки: • AS (AS332T/P-A, AS324MT-A, AS320T/P-B, AS300N-A, AS228T/P/R-A, AS218TX/PX/RX-A); • DVP. Драйвер использует протокол **Modbus** и позволяет обращаться напрямую к областям памяти контроллеров (X, Y, D и т. д.)
Omron	Master	Поддерживаются линейки: • CP (CP1E-30N); • CJ (CJ1, CJ1G-CPU44, CJ1G-CPU45, CJ2M-CPU11); • CS1 (CS1G, CS1H).
ABB	Master	Поддерживается **AC500**. Драйвер использует протокол **Modbus** и позволяет обращаться напрямую к областям памяти контроллеров (MX, MW, MD и т. д.)
Сканеры штрихкодов	Slave	

Таблица 11.2 – Протоколы обмена, поддерживаемые панелями СП210 расширенных модификаций по интерфейсу Ethernet

Протокол	Режим	Комментарий
Modbus TCP	Master/Slave	
CODESYS Symbolic	Client	
OPC UA	Client/Server	В режиме **Client** поддерживается только подключение к контроллерам, программируемым в среде CODESYS V3.5
Siemens	Master	Поддерживаются модели: • S7-200 smart; • S7-300; • S7-1200, S-1500; • Logo. Драйвер **Logo** использует протокол **Modbus** и позволяет обращаться напрямую к областям памяти контроллеров (I, Q, M и т. д.). Остальные драйвера используют специфичные для Siemens способы обмена (например, PUT/GET для S7-1200 и S7-1500)

Продолжение таблицы 11.2

Протокол	Режим	Комментарий
Mitsubishi	Master	Поддерживаются линейки: • Q/L; • FX5U (протокол **SLMP**).
Delta	Master	Поддерживаются модели AS228T/P/R-A и AS218TX/PX/RX-A. Драйвер использует протокол Modbus и позволяет обращаться напрямую к областям памяти контроллеров (X, Y, D и т. д.)
OMRON	Master	Драйвер протокола **FINS**
Beckhoff ADS	Master	
MQTT	Client	

11.2 Общий принцип настройки обмена

В рамках обмена с другими устройствами панель может функционировать в двух режимах:

- **master (client)** – в этом режиме панель является инициатором обмена, отправляя запросы на чтение и запись данных другим устройствам;
- **slave (server)** – в этом режиме панель обрабатывает запросы, полученные от других устройств.

Панели базовой модификации (-Б) имеют только один коммуникационный интерфейс – **COM-порт**. Панели расширенной модификации (-Р) имеют **два COM-порта** (кроме СП210-05-Р) и интерфейс **Ethernet**.

Для каждого COM-порта может использоваться:

- только один протокол обмена;
- только один режим работы (master или slave);
- только одно сочетание сетевых настроек (скорости, режима контроля чётности и т. д.).

У панели СП210-05-Р интерфейс RS-485/RS-232 обозначается как COM2. У остальных панелей интерфейс RS-485/RS-232 (1) обозначается как COM1, и интерфейс RS-485/RS-232 (2) как COM2.

ПРИМЕЧАНИЕ
У панелей базовой модификации (-Б) интерфейс RS-232 обозначается как **COM1**, а интерфейс RS-485 – как **COM2**.

Для интерфейса **Ethernet** могут одновременно использоваться несколько протоколов обмена в разных режимах работы.

Общий принцип настройки обмена:

- в меню **Файл – Настройки проекта – Устройства** отображаются все интерфейсы панели;

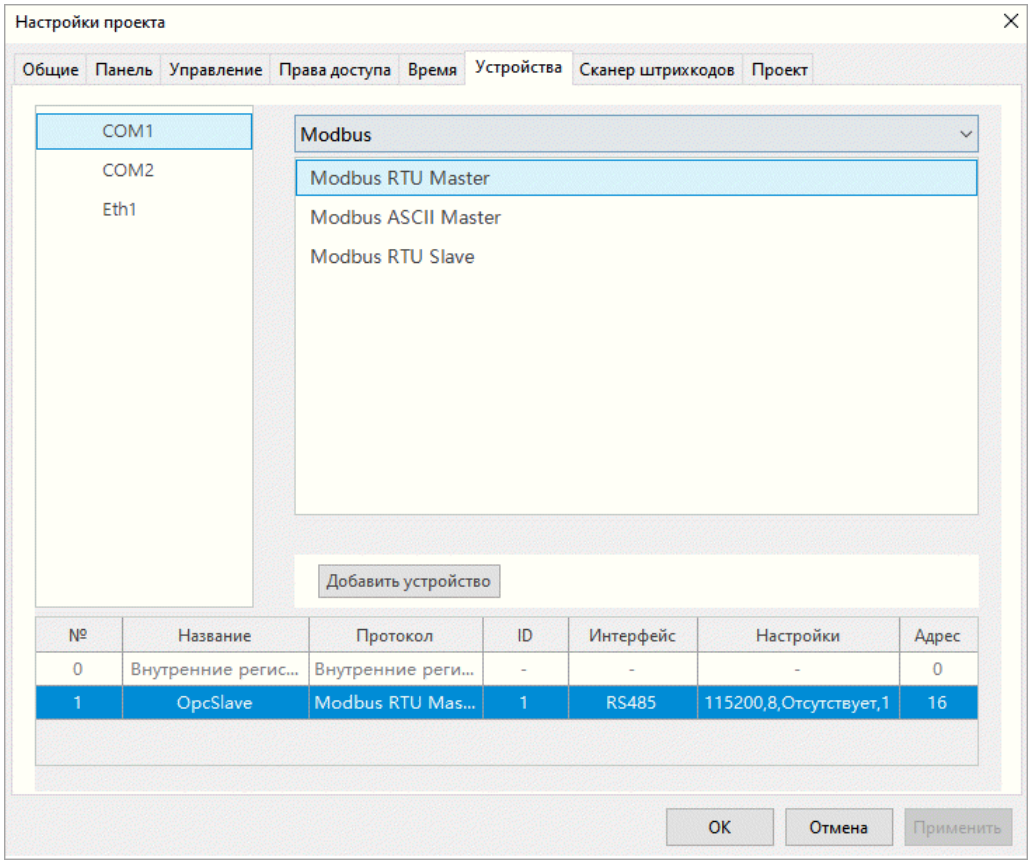


Рисунок 11.1 – Настройки проекта, вкладка Устройства

- для каждого интерфейса могут быть добавлены опрашиваемые slave-устройства (если панель работает в режиме **Master/Client**) или выбрана работа в режиме **Slave/Server**;
- при добавлении устройства указываемое имя должно являться уникальным;
- в настройках графических элементов указываются адреса битов/регистров опрашиваемых устройств (для режима **Master/Client**) или самой панели (для режима **Slave/Server**).

Особенности:

- настройка протокола **MQTT** выполняется в меню **Инструменты – VNC и MQTT**;
- настройка обмена со сканером штрих кодов выполняется в меню **Файл – Настройки проекта – Сканер штрихкодов**;
- реализация пользовательского протокола выполняется с помощью макросов;
- обмен по протоколу Modbus может быть реализован с помощью макросов;
- в макросах поддерживается доступ к переменным, опрос которых организован по протоколам CODESYS Symbolic и OPC UA.

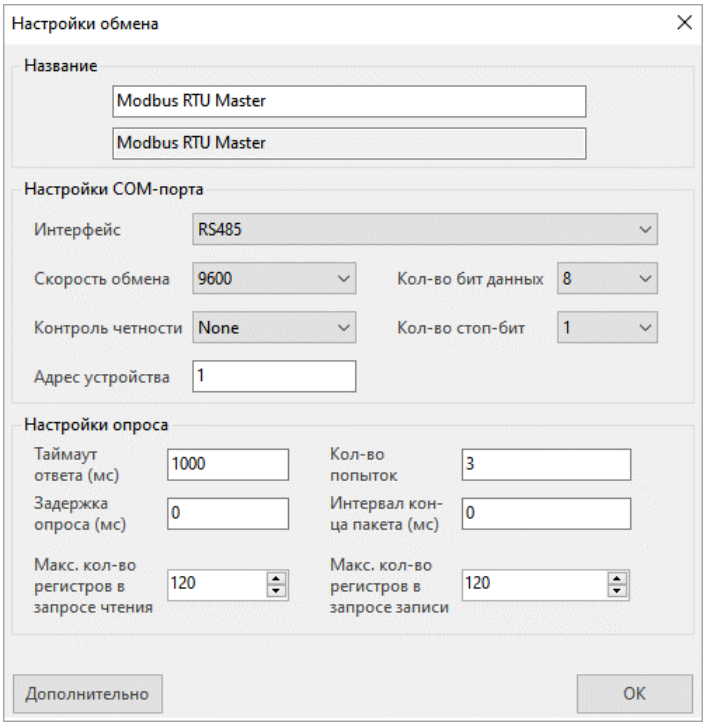
11.3 Настройка обмена по протоколу Modbus

11.3.1 Modbus RTU/ASCII Master

В пределах **COM-порта** может быть добавлено до **32** компонентов **Modbus RTU Master** или **Modbus ASCII Master**. Каждый компонент соответствует одному опрашиваемому slave-устройству. У всех устройств должны совпадать настройки COM-порта: скорость обмена, контроль четности, количество бит данных и количество стоп бит. Название и адрес каждого устройства должны быть уникальными.

В таблице ниже описаны настройки компонентов **Modbus RTU Master** и **Modbus ASCII Master**.

Таблица 11.3 – Настройка компонентов Modbus RTU Master и Modbus ASCII Master

Параметр	Описание
Основные настройки	
	
Рисунок 11.2 – Основные настройки опроса slave-устройства в режиме Modbus RTU Master	
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
Интерфейс	Данная настройка носит информационный характер. Фактически панель отправляет настроенные запросы сразу по обоим интерфейсам COM-порта – и по RS-485, и по RS-232
Скорость обмена	Возможные значения: 4800/9600/19200/38400/57600/115200
Контроль четности	Возможные значения: • None – проверка четности не выполняется; • Even – проверка на четность; • Odd – проверка на нечетность.
Кол-во бит данных	Для протокола **Modbus RTU** должно использоваться значение **8**, для протокола **Modbus ASCII** – **7**
Кол-во стоп-бит	Возможные значения: **1** или **2**
Адрес устройства	Адрес опрашиваемого устройства. Значение **0** соответствует адресу широковещательного запроса
Таймаут ответа	Время ожидания ответа от slave-устройства в мс (не более **32767**). Значение **0** соответствует минимально возможному времени ожидания

Продолжение таблицы 11.3

Параметр	Описание
Кол-во попыток	Общее количество попыток отправки запроса в том случае, если на него не был получен ответ от slave-устройства. После указанного количества попыток происходит отправка следующего запроса
Задержка опроса	Задержка между получением ответа от slave-устройства и отправкой следующего запроса в мс (не более **32767**; **0** – задержка отсутствует). Соответствует параметру **turnaround delay** из спецификации протокола **Modbus**
Интервал конца пакета	Интервал отсутствия новых байт, по которому определяется факт получения полного пакета данных в мс (не более **32767**; **0** – используется значение из спецификации протокола **Modbus**). Соответствует параметру **inter-frame delay** из спецификации протокола **Modbus**. Используется только для Modbus RTU, для Modbus ASCII конец пакета определяется по стоп-символам
Макс. кол-во регистров в запросе чтения	Максимальное количество регистров в групповом запросе чтения. Применительно к битам это значение домножается на **16** Пример: значение 120 означает, что за один запрос может быть считано не более 120 регистров или не более 1920 бит
Макс. кол-во регистров в запросе записи	Максимальное количество регистров в групповом запросе записи. Применительно к битам это значение домножается на **16** Пример: значение 120 означает, что за один запрос может быть записано не более 120 регистров или не более 1920 бит

Дополнительные настройки

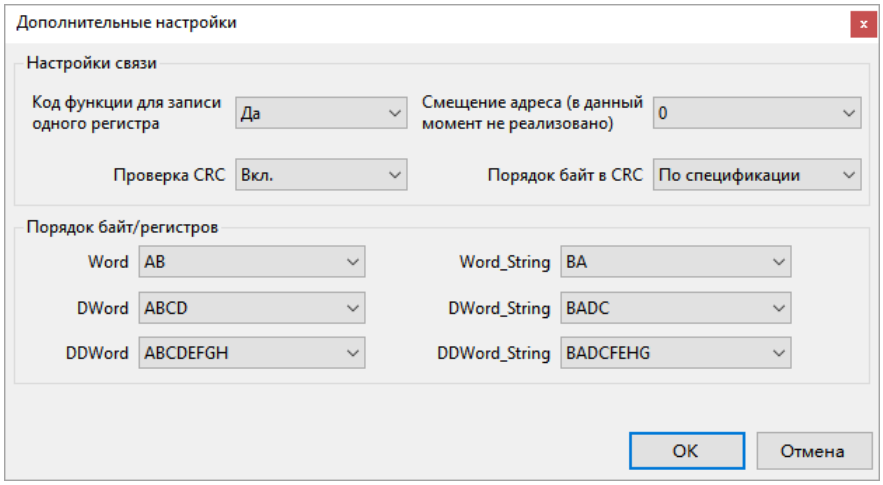


Рисунок 11.3 – Дополнительные настройки опроса slave-устройства в режиме Modbus RTU Master

Код функции для записи одного регистра	**Код функции Modbus**, используемой для записи значения, занимающего один регистр (типа **Word**): **Да (0x10)** – Write Multiple Registers; **Нет (0x06)** – Write Single Register. Влияет на все варианты опроса (через элементы, через функциональную область/функциональную кнопку и через макросы)
Проверка CRC	В случае установки значения **Вкл.** драйвер Modbus выполняет проверку контрольной суммы в полученных пакетах. Значение настройки **Порядок байт в CRC** определяет порядок байт в запросах панели
Порядок байт в CRC	Данный параметр обрабатывается только в том случае, если параметр **Проверка CRC** имеет значение **Вкл.**, и определяет порядок байт в контрольной сумме: • По спецификации – младшим байтом вперед; • Инвертированный – старшим байтом вперед.
Порядок байт/регистров	Порядок байт и регистров для числовых и строковых типов данных. Влияет только на опрос через элементы

ПРИМЕЧАНИЕ

После получения ответа от slave-устройства панель формирует следующий запрос с учётом параметра **Задержка опроса**. Задержка необходима для перехода COM-порта slave-устройства из режима передачи в режим приёма. Значение параметра подбирается опытным путём.

ПРИМЕЧАНИЕ

В **Modbus RTU** окончание пакета определяется по интервалу отсутствия новых байт, заданному в спецификации.

Однако некоторые slave-устройства могут делать более длинные паузы между байтами. Для корректного обмена с такими устройствами используется параметр **Интервал конца пакета**. Его значение подбирается опытным путём.

ПРИМЕЧАНИЕ

В спецификации Modbus указано, что порядок байт в контрольной сумме – младшим байтом вперед. Некоторые slave-устройства могут отступать от спецификации, и для корректного обмена с ними используются параметры **Проверка CRC** и **Порядок байт в CRC**.

ПРИМЕЧАНИЕ

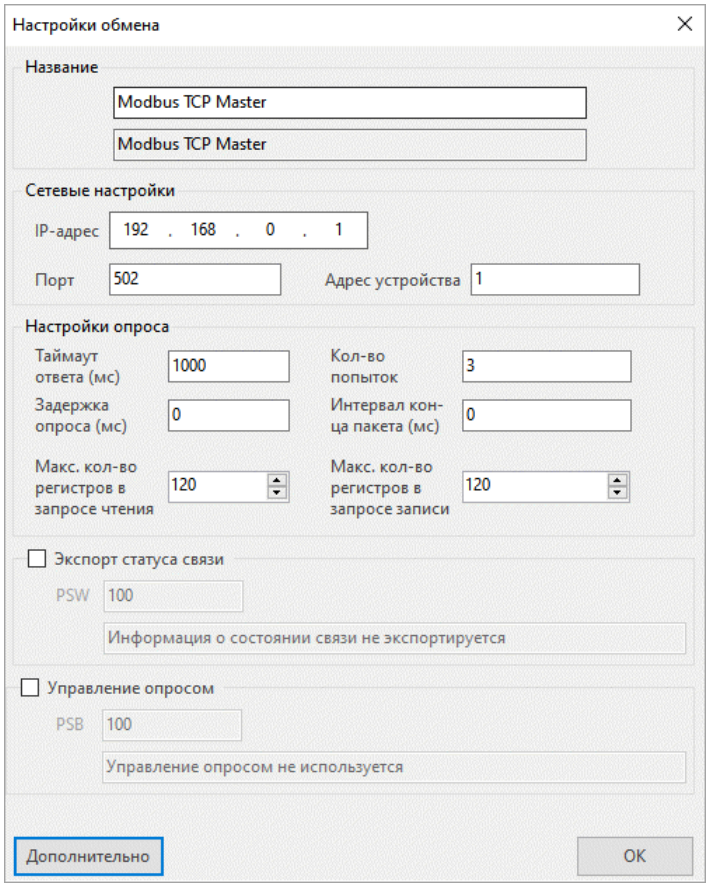
Спецификация Modbus не определяет порядок регистров при передаче параметров; порядок байт стандартизирован (старшим байтом вперед), но некоторые slave-устройства могут отступать от спецификации. Для корректной интерпретации данных, считываемых или записываемых в такие устройства, используются настройки группы **Порядок байт/регистров**.

11.3.2 Modbus TCP Master

В пределах интерфейса **Ethernet** может быть добавлено до **50** компонентов с различными протоколами, каждый из которых соответствует одному опрашиваемому slave-устройству.

В таблице ниже описаны настройки компонента **Modbus TCP Master**. Отсутствующие в таблице параметры унаследованы от компонента Modbus RTU Master и не используются.

Таблица 11.4 – Настройка компонента Modbus TCP Master

Параметр	Описание
Основные настройки	
	
Рисунок 11.4 – Основные настройки опроса slave-устройства в режиме Modbus TCP Master	
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
IP-адрес	IP-адрес slave-устройства
Порт	TCP-порт slave-устройства
Адрес устройства	Адрес (**Unit ID**) опрашиваемого устройства. Обычно используется только при опросе приборов, подключенных через шлюз **Modbus TCP/Modbus RTU**. Некоторые slave-устройства не обрабатывают значение этого параметра, а некоторые требуют, чтобы он имел конкретное значение (например, 255, 0, 1 или другое)
Таймаут ответа	Время ожидания ответа от slave-устройства в мс (не более **32767**). Значение **0** соответствует минимально возможному времени ожидания
Кол-во попыток	Общее количество попыток отправки запроса в том случае, если на него не был получен ответ от slave-устройства. После указанного количества попыток происходит отправка следующего запроса

Продолжение таблицы 11.4

Параметр	Описание
Задержка опроса	Задержка между получением ответа от slave-устройства и отправкой следующего запроса в мс (не более **32767**; **0** – задержка отсутствует). Соответствует параметру **turnaround delay** из спецификации протокола Modbus
Макс. кол-во регистров в запросе чтения	Максимальное количество регистров в групповом запросе чтения. Применительно к битам это значение умножается на **16** Пример: значение 120 означает, что за один запрос может быть считано не более 120 регистров или не более 1920 бит
Макс. кол-во регистров в запросе записи	Максимальное количество регистров в групповом запросе записи. Применительно к битам это значение умножается на **16** Пример: значение 120 означает, что за один запрос может быть записано не более 120 регистров или не более 1920 бит
Экспорт статуса связи	В случае установки галочки указывается адрес первого из **4** регистров, используемых для диагностики связи с данным slave-устройством. Например, для **PSW100**: • **PSW100** – кол-во запросов, на которые был получен корректный ответ; • **PSW101** – общее кол-во ошибок обмена (ошибкой считается получение пакета с некорректной структурой или кодом ошибки протокола Modbus); • **PSW102** – кол-во запросов, на которые не был получен ответ; • **PSW103** – адрес (Unit ID) последнего устройства, при обмене с которым возникла ошибка или сработал таймаут
Управление опросом	В случае установки галочки указывается адрес бита, который будет использоваться для управления опросом данного slave-устройства: • **TRUE** – отключить опрос; • **FALSE** – возобновить опрос
Дополнительные настройки	
Код функции для записи одного регистра	Код функции Modbus, используемой для записи значения, занимающего один регистр (типа **Word**): **0x10** – Write Multiple Registers; **0x06** – Write Single Register. Влияет на все варианты опроса (через элементы, через функциональную область/функциональную кнопку и через макросы)
Порядок байт/ регистров	Порядок байт и регистров для числовых и строковых типов данных. Влияет только на опрос через элементы

ВНИМАНИЕ

В конфигураторе версии **2.3.7** и соответствующей прошивке присутствует особенность работы драйвера Modbus TCP Master: Если в настройках панели указан и доступен сетевой шлюз, панель циклически отправляет **первому** в списке Modbus TCP Slave устройству избыточный запрос на чтение coil'a с адресом 0:

- Если slave-устройство не имеет такого coil'a и возвращает ответ с кодом ошибки **0x01 (ILLEGAL FUNCTION)** или **0x02 (ILLEGAL DATA ADDRESS)**, то на экране панели будет постоянно появляться системное окно **Ошибка обмена по Ethernet-порту** (номер окна - 25003).
- Если slave-устройство имеет coil с адресом 0, то данное окно появляться не будет.

Варианты устранения проблемы:

Если панель и связанные с ней устройства подключены **к одной локальной сети**: укажите в настройках шлюза панели адрес **0.0.0.0**.

Если панель и связанные с ней устройства подключены **к разным локальным сетям**: оставьте в сетевых настройках адрес шлюза сети. Дополнительно настройте панель на работу в режиме Modbus TCP Slave и первым опрашиваемым **Modbus TCP Slave** устройством в настройках проекта укажите саму панель (IP **127.0.0.1**).

Примечание: Описанное поведение не воспроизводится в режиме эмуляции. Исправление запланировано в следующей версии ПО и прошивки.

11.3.3 Modbus Master – способы настройки обмена

Для панели, работающей в режиме **Modbus Master**, предусмотрено три способа настройки обмена:

1. Опрос через элементы
2. Опрос через функциональную область (действие **Копирование данных**)
3. Опрос через функциональную область (действие **Вызов макроса**)

Таблица 11.5 – Способы настройки обмена в режиме Modbus Master

	Элементы	Копирование данных	Макросы
Сложность настройки	очень низкая	низкая	средняя
Установка периода опроса	отсутствует	имеется	имеется
Настройка количества бит/ регистров в групповом запросе	отсутствует	имеется	имеется
Управление опросом	только на уровне всего устройства	на уровне отдельных запросов	на уровне отдельных запросов
Настройка порядка байт/ регистров	через меню **Дополнительные настройки**	вручную	вручную

1. Опрос через элементы

В настройках любого динамизируемого параметра графического элемента можно указать slave-устройство и идентификатор области памяти **Modbus**.

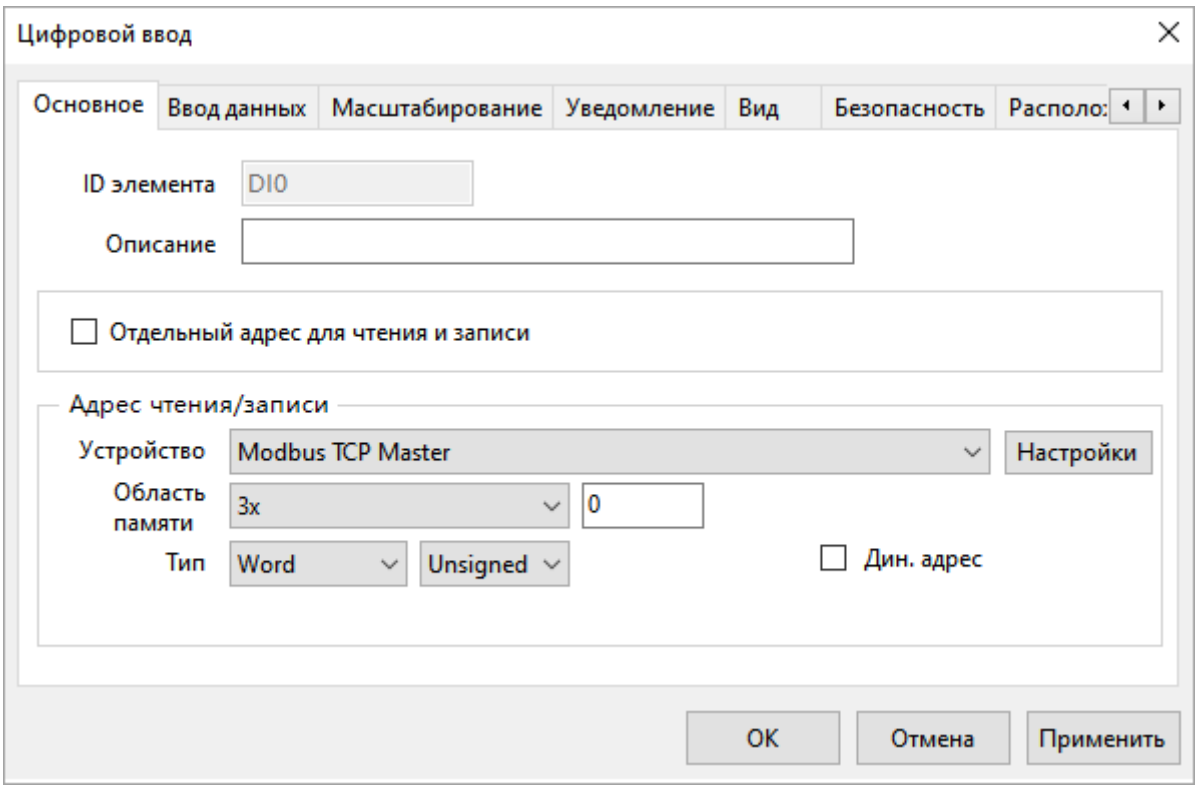


Рисунок 11.5 – Цифровой ввод

Если этот параметр используется только для отображения (например, бит элемента **Индикатор**), то он формирует запрос циклический запрос чтения.

Если параметр используется для ввода данных (например, бит элемента **Переключатель**), то он формирует только однократный запрос записи в момент изменения значения.

Если параметр используется и для чтения, и для ввода данных (например, бит элемента **Переключатель с индикацией**), то он формирует и запрос чтения, и запрос записи.

ПРИМЕЧАНИЕ
Если в настройках slave-устройства указан адрес **0** (широковещательный адрес), то данное устройство может быть выбрано только в элементах, не формирующих запросы чтения (например, в элементе **Функциональная кнопка**).

Таблица 11.6 – Области памяти Modbus

ID области памяти Modbus	Функция, используемая в запросе чтения	Функция, используемая в запросе записи
0x	0x01 (Read Coils)	**0x05** (Write Single Coil)
1x	0x02 (Read Discrete Inputs)	-
3x	0x04 (Read Input Registers)	-
4x	0x03 (Read Holding Registers)	**0x06** (Write Single Register) **0x10** (Write Multiple Registers) Для параметров, занимающих один регистр (типа Word), используемая функция зависит от настройки устройства **Код функции для записи одного регистра**

Для «битовых» элементов (**Индикатор**, **Переключатель** и т. д.) поддерживается привязка бита регистра (в стиле **4xR.B** и **3xR.B**, где **R** – адрес регистра, **B** – номер его бита).

В случае **чтения** элемент формирует запрос на чтение регистра и извлекает из него указанный бит.

В случае **записи** элемент считывает значение регистра, меняет в нем значение указанного бита и записывает его обратно.

Запросы на чтение отправляются циклически.

Период их отправки зависит от «нагруженности» экрана панели. Для «пустого» проекта с элементом, формирующим запрос на чтение одного регистра, период отправки по протоколу **Modbus RTU/ASCII** составляет **50** мс, по протоколу **Modbus TCP** – **15** мс. Запросы формируются только в том случае, если в данный момент на дисплее панели открыт экран, на котором размещен данный элемент (независимо от видимости/невидимости данного элемента). Запросы, настроенные в меню **Настройка тревог** и **Выборка данных**, отправляются независимо от открытого экрана.

Запросы на запись отправляются однократно в момент изменения значения.

Для **Modbus RTU/ASCII**:

Панель оптимизирует операции чтения, объединяя их в групповые запросы. Если разница между адресами «соседних» регистров не превышает **5**, то они будут объединены в групповой запрос.

Например, если к элементам привязаны адреса **4x0**, **4x5** и **4x10** – то панель сформирует один групповой запрос на чтение **11** регистров, начиная с адреса **0**. Если к элементам привязаны адреса **4x0**, **4x6** и **4x12**, то панель сформирует 3 отдельных запроса. Максимальное количество считываемых регистров в одном запросе ограничено значением параметра **Макс. кол-во регистров в запросе чтения** (не более **120**). Применительно к битам значение этого параметра умножается на **16**.

Если разница между адресами «соседних» бит не превышает **65**, то они будут объединены в групповой запрос.

Если к элементам привязаны адреса **0x0**, **0x65** и **0x130**, то панель сформирует один групповой запрос на чтение **131** бита, начиная с адреса **0**. Если к элементам привязаны адреса **0x0**, **0x66** и **0x132**, то панель сформирует **3** отдельных запроса.

Для **Modbus TCP**:

Панель оптимизирует операции чтения, объединяя их в групповые запросы. Если разница между адресами «соседних» регистров не превышает **17**, то они будут объединены в групповой запрос.

Например, если к элементам привязаны адреса **4x0**, **4x17** и **4x34** – то панель сформирует один групповой запрос на чтение **11** регистров, начиная с адреса **0**. Если к элементам привязаны адреса **4x0**, **4x18** и **4x36**, то панель сформирует 3 отдельных запроса. Максимальное количество считываемых регистров в одном запросе ограничено значением параметра **Макс. кол-во регистров в запросе чтения** (не более **120**). Применительно к битам значение этого параметра умножается на **16**.

Если разница между адресами «соседних» бит не превышает **257**, то они будут объединены в групповой запрос.

Если к элементам привязаны адреса **0x0**, **0x257** и **0x514**, то панель сформирует один групповой запрос на чтение **515** битов, начиная с адреса **0**. Если к элементам привязаны адреса **0x0**, **0x258** и **0x516**, то панель сформирует **3** отдельных запроса.

В некоторых случаях автоматическое формирование групповых запросов является неудобным, потому что в их состав могут войти адреса регистров, которые отсутствуют в slave-устройстве. В ответ на такой запрос будет отправлен ответ с кодом ошибки **0x02** (ILLEGAL DATA ADDRESS). Если регистр элемента не должен включаться в состав группового запроса, то в его настройках нужно установить галочку **Дин. адр.** и указать адрес регистра, который не используется в проекте панели (т. е. регистра, значение которого всегда равно **0**). В этом случае для чтения данного параметра будет формироваться отдельный **Modbus-запрос**, который будет отправляться два раза подряд.

Некоторые графические элементы формируют Modbus-запросы особым образом – например, значения, отображаемые в Таблице данных, считываются несколькими запросами, в рамках каждого из которых считывается не более **16** регистров.

Опрос через элементы имеет следующие ограничения:

- нельзя настроить период опроса;
- нельзя настроить циклическую запись данных;
- отсутствует возможность детальной настройки групповых запросов;
- отсутствует возможность использовать считанные данные в проекте панели. В ряде случаев требуется обработать считанные данные в макросе или сделать их доступными для master-устройства, подключённого к другому интерфейсу панели (например, панель в режиме Modbus RTU Master опрашивает модули ввода-вывода, и должна предоставить доступ к этим данным контроллеру, для которого она работает в режиме Modbus TCP Slave).

Некоторые из перечисленных выше ограничений можно обойти, если настроить опрос через функциональную область.

2. Опрос через функциональную область (действие Копирование данных)

В элементе **Функциональная область** можно настроить действие **Копирование данных**, которое обеспечивает копирование заданного количества битов/регистров с адреса источника по адресу назначения.

Таблица 11.7 – Варианты настройки копирования данных

Адрес источника	Адрес назначения	Выполняемая операция
Биты/регистры slave-устройства	Биты/регистры панели	Чтение битов/регистров slave-устройств
Биты/регистры панели	Биты/регистры slave-устройства	Запись битов/регистров slave-устройства
Биты/регистры первого slave-устройства	Биты/регистры второго slave-устройства	Чтение битов/регистров первого slave-устройства и их запись в биты/регистры второго slave-устройства
Биты/регистры панели	Биты/регистры панели	Копирование значений из одних регистров панели в другие, без выполнения Modbus-запросов

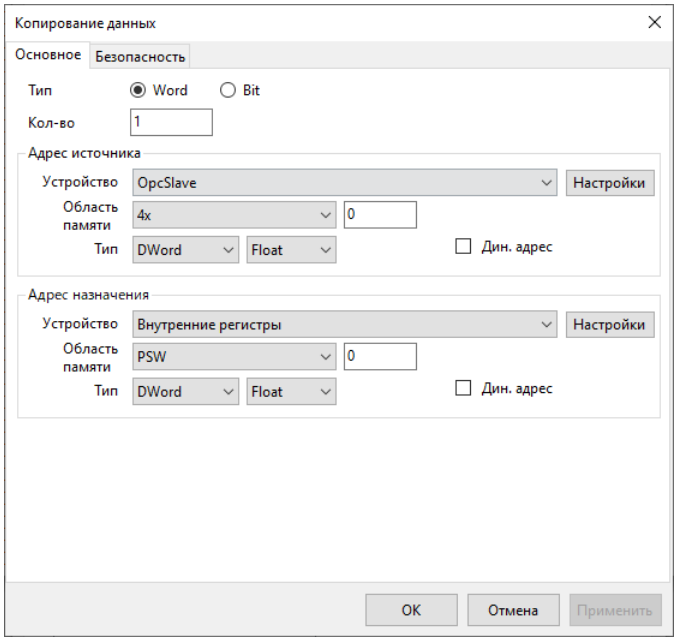


Рисунок 11.6 – Настройки действия Копирование данных

Довольно часто требуется, чтобы действия, настроенные в функциональной области, выполнялись независимо от того, какой экран в данный момент отображается на панели. В этом случае нужно сделать функциональную область **глобальной**, после чего она будет отображаться на всех экранах проекта.

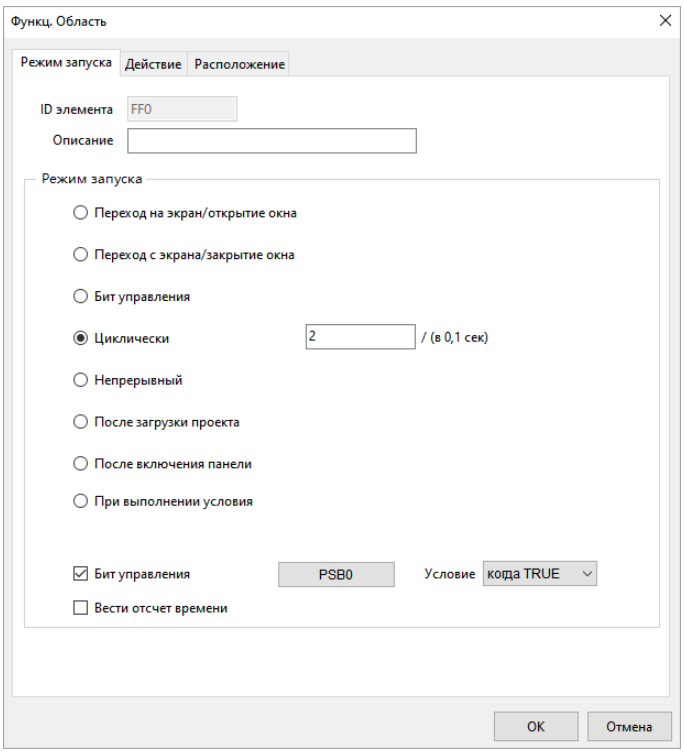


Рисунок 11.7 – Функциональная область

Минимально возможный период опроса (на примере «пустого» проекта с одним Modbus-запросом на чтение одного регистра):

- для режима запуска **Циклически** – 200 мс (Modbus RTU, Modbus ASCII) / 150 мс (Modbus TCP);
- для режима запуска **Непрерывно** – 100 мс (Modbus RTU, Modbus ASCII) / 50 мс (Modbus TCP).

С помощью **бита управления** можно останавливать/возобновлять отправку настроенных в функциональной области запросов. Допустимо использование и других режимов запуска – в этом случае Modbus-запрос будет однократно отправлен при наступлении заданного события (например, при переходе на данный экран).

Максимальное количество считываемых/записываемых регистров в одном запросе ограничено значениями параметров **Макс. кол-во регистров в запросе чтения** и **Макс. кол-во регистров в запросе записи** (не более **120**). Применительно к битам значение этих параметров умножается на **16**.

Действие **Копирование данных** может быть настроено в элементе **Функциональная кнопка**. В этом случае Modbus-запрос будет однократно отправлен при нажатии или отпускании кнопки.

3. Опрос через функциональную область (действие Вызов макроса)

В элементе **Функциональная область** можно настроить действие **Вызов макроса**. В рамках макроса можно использовать следующие коммуникационные функции для обмена по протоколу Modbus:

- **Read** – чтение одного параметра произвольного типа;
- **Reads** – чтение одного или нескольких параметров произвольного типа;
- **Write** – запись одного параметра произвольного типа;
- **Writes** – запись одного или нескольких параметров произвольного типа;
- **WriteF** – запись одного параметра типа **Float** или **Double**.

Одним из аргументов функции является область памяти Modbus, на основании которой определяется код функции, используемой в запросе (см. таблицу 11.6).

Вся остальная информация, приведённая в описании действия **Копирование данных**, актуальна и для обмена через макросы.

11.3.4 Modbus RTU Slave

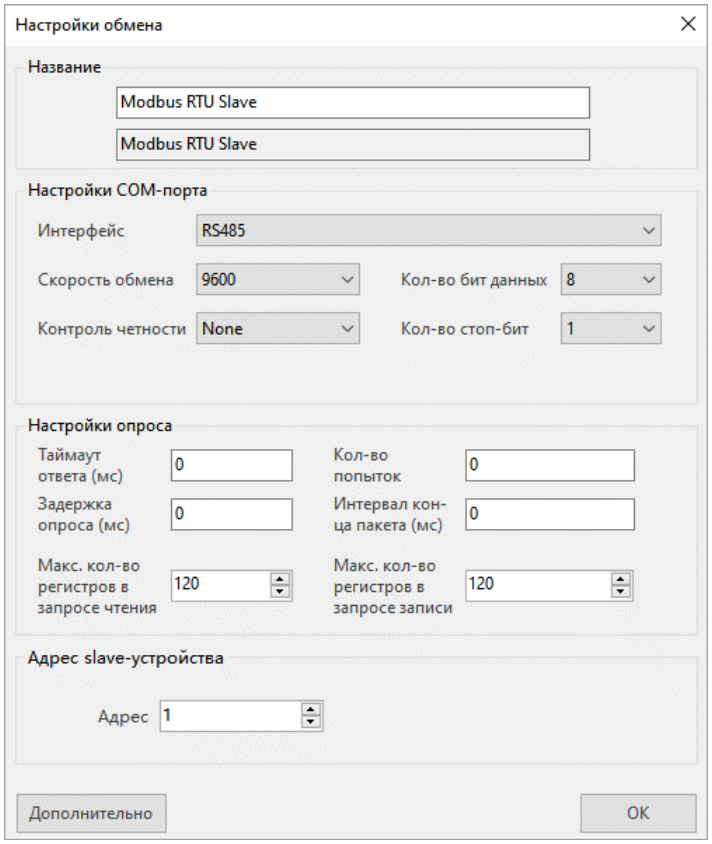
В пределах COM-порта может быть добавлен только один компонент **Modbus RTU Slave**, который будет «прослушивать» оба интерфейса этого порта – и RS-485, и RS-232. Панель поддерживает получение широковещательных запросов (отправленных на адрес **0**).

Поддерживаются групповые запросы чтения и записи – до **120** регистров / **1920** бит в одном запросе.

В таблице ниже описаны настройки компонента **Modbus RTU Slave**. Отсутствующие в таблице параметры унаследованы от компонента **Modbus RTU Master** и не используются.

Описание адресации битов и регистров панели в режиме **Slave** приведено в разделе 11.3.6.

Таблица 11.8 – Настройка компонента Modbus RTU Slave

Параметр	Описание
Основные настройки  **Рисунок 11.8 – Настройки панели в режиме Modbus RTU Slave**	
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
Интерфейс	Данная настройка носит информационный характер. Фактически панель будет «прослушивать» оба интерфейса COM-порта – и RS-485, и RS-232
Скорость обмена	Возможные значения: 4800/9600/19200/38400/57600/115200
Контроль четности	Возможные значения: • **None** – проверка четности не выполняется; • **Even** – проверка на четность; • **Odd** – проверка на нечетность.
Кол-во бит данных	Поддерживается только значение **8**
Кол-во стоп-бит	Возможные значения: **1** или **2**

Продолжение таблицы 11.8

Параметр	Описание
Задержка опроса	Задержка между получением запроса от master-устройства и отправкой ответа в мс (не более **32767**; **0** – задержка отсутствует). Задержка позволяет master-устройству успеть переключить свой COM-порт из режима передачи в режим приема
Адрес	Адрес панели в сети Modbus RTU
Дополнительные настройки	
Проверка CRC	В случае установки значения **Вкл.** драйвер Modbus выполняет проверку контрольной суммы в полученных пакетах. Значение настройки **Порядок байт в CRC** определяет ожидаемый порядок байт в запросах master-устройства
Порядок байт в CRC	Данный параметр определяет порядок байт в контрольной сумме, который панель ожидает в запросе и который использует в ответе, отправляемому master-устройству: • **По спецификации** – младшим байтом вперед; • **Инвертированный** – старшим байтом вперед. Значение **Инвертированный** следует использовать в том случае, если порядок байт контрольной суммы, которое ожидает в ответе master-устройство, отличается от спецификации Modbus

11.3.5 Modbus TCP Slave

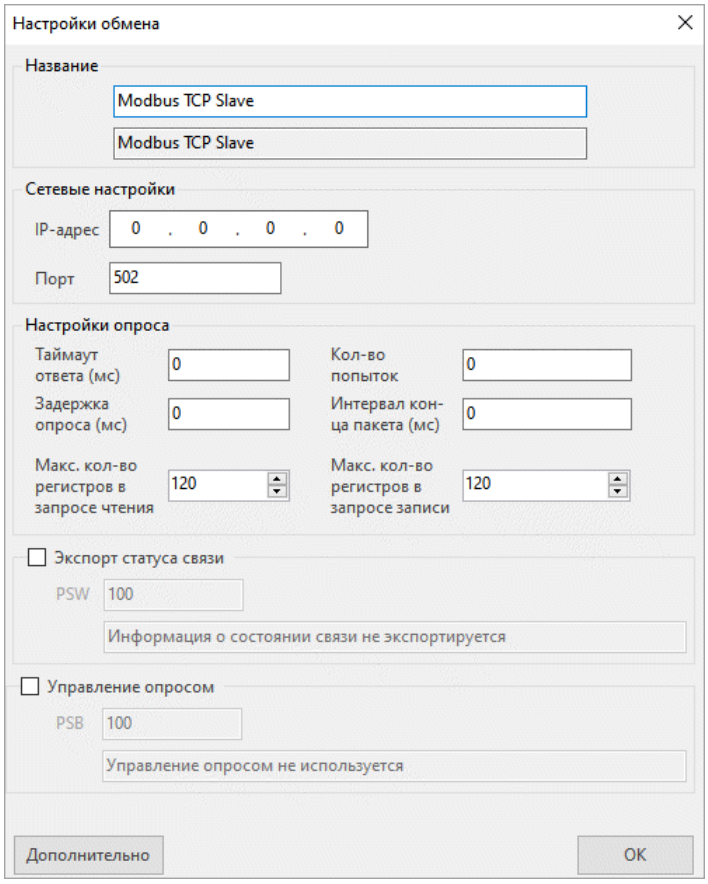
В пределах интерфейса **Ethernet** может быть добавлен компонент **Modbus TCP Slave**. Панель может одновременно работать в режиме **Modbus TCP Master** и **Modbus TCP Slave**, при этом для обоих режимов может использоваться один и тот же номер порта.

Рекомендуемое одновременное количество подключений – не более **64**. Фактическое общее количество одновременных подключений ограничено максимальным числом дескрипторов, используемых процессом операционной системы (**1024**; из них около **40** используется приложением панели).

Поддерживаются групповые запросы чтения и записи – до **120** регистров / **1920** бит в одном запросе.

В таблице ниже описаны настройки компонента **Modbus TCP Slave**. Отсутствующие в таблице параметры унаследованы от других Modbus-драйверов и не используются.

Таблица 11.9 – Настройки компонента Modbus TCP Slave

Параметр	Описание
Основные настройки	
	
Рисунок 11.9 – Настройки панели в режиме Modbus TCP Slave	
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
IP-адрес	Значение параметра не используется
Порт	TCP-порт панели, используемый для работы в режиме **Modbus TCP Slave**
Дополнительные настройки	
Адрес устройства	Адрес (**Unit ID**) панели. Панель отвечает только на запросы с данным **Unit ID**

11.3.6 Modbus Slave – адресация регистров панели

Если панель работает в режиме **Modbus Slave**, то для её памяти используется следующая модель адресации:

Таблица 11.10 – Адресация памяти панели в режиме Modbus Slave

Область памяти панели	Область памяти Modbus	Поддерживаемые функции Modbus
PSB0…9999	Coils 0…9999	0x01 (Read Coils) 0x05 (Write Single Coil) 0x0F (Write Multiple Coils)
PSW0…9999	Holding registers 0…9999	0x03 (Read Holding Registers) 0x06 (Write Single Register) 0x10 (Write Multiple Registers)
PFW0…55535	Holding registers 10000…65535	

Прямой доступ по Modbus к системным битам и регистрам панели (области **SPSB**, **SPSW**, **SPFW**) не поддерживается, но он может быть осуществлен косвенно, путём копирования данных через «пользовательские» области памяти (PSB, PSW, PFW) с помощью элемента **Функциональная область**.

11.4 Пользовательский протокол

Режим **Пользовательский протокол** доступен только для COM-порта и используется для работы по пользовательскому (нестандартному) протоколу обмена. Реализация протокола выполняется в макросе с помощью функций **COMSend** (отправка данных в COM-порт) и **COMReceive** (получение данных из COM-порта). См. примеры в пункте 12.6.4.

В компоненте указываются сетевые настройки интерфейса (см. таблицу 11.3).

11.5 Символьный протокол CODESYS V3

Символьный протокол CODESYS V3 (PLC Handler) используется для обмена с контроллерами, программируемыми в среде CODESYS V3.5. Поддерживается только подключение без использования логина/пароля.

В пределах интерфейса Ethernet может быть добавлено до **50** компонентов с различными протоколами, каждый из которых соответствует одному опрашиваемому устройству.

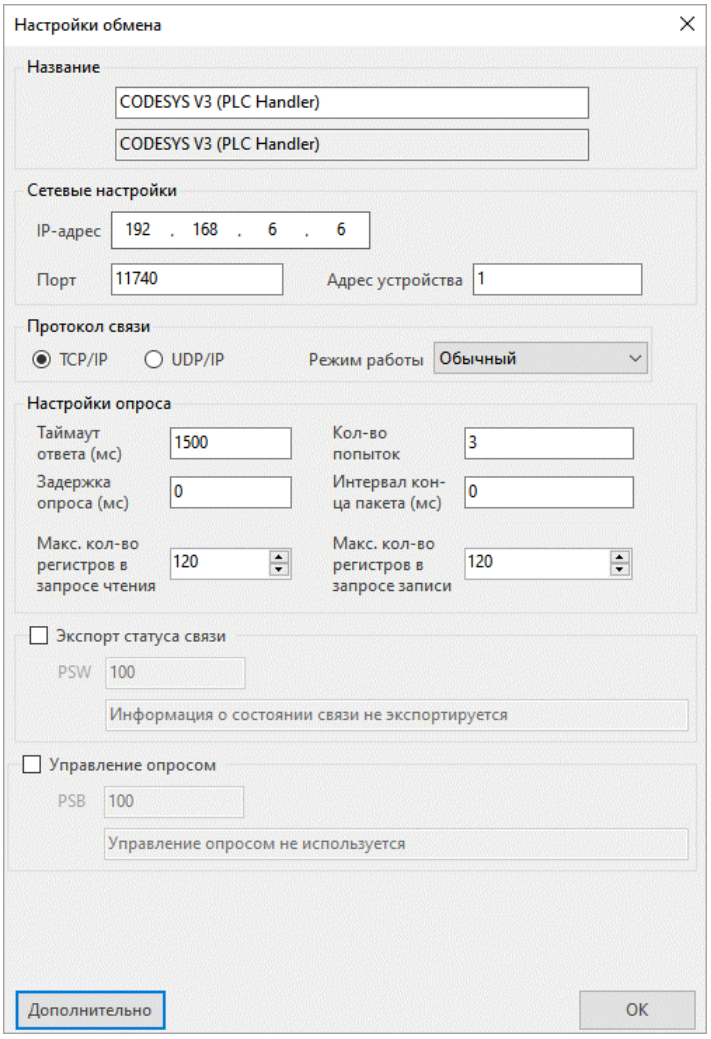


Рисунок 11.10 – Настройки обмена по символьному протоколу CODESYS V3

В таблице ниже описаны настройки компонента **CODESYS V3 (PLC Handler)**. Отсутствующие в таблице параметры не используются драйвером протокола.

Таблица 11.11 – Настройка компонента CODESYS V3 (PLC Handler)

Параметр	Описание
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
IP-адрес	IP-адрес опрашиваемого ПЛК
Порт	Номер порта опрашиваемого ПЛК. В большинстве случаев используется значение по умолчанию – **11740**
Протокол	**TCP** (обеспечивает гарантию доставки пакетов) или **UDP** (обеспечивает более высокую скорость обмена)

Продолжение таблицы 11.11

Параметр	Описание
Режим работы	Возможные значения: • Обычный – используется один коммуникационный канал, приоритет запросов чтения выше запросов записи; • Приоритет команд записи – используется один коммуникационный канал, приоритет запросов записи выше запросов чтения; • Двухканальный – используются два коммуникационных канала (для чтения и для записи). Количество доступных коммуникационных каналов ПЛК определяется параметром **MaxChannels** компонента **CmpChannelServer**. В большинстве случаев этот параметр по умолчанию имеет значение **4**. При этом подключение к ПЛК из среды CODESYS использует один из коммуникационных каналов. Более подробная информация об этом параметре применительно к контроллерам ОВЕН доступна в документе CODESYS V3.5. Настройка обмена с верхним уровнем
Таймаут ответа	Время ожидания ответа от ПЛК в мс (не более **32767**). Значение **0** соответствует минимально возможному времени ожидания
Экспорт статуса связи	В случае установки галочки указывается адрес первого из **4** регистров, используемых для диагностики связи с данным ПЛК. Например, для **PSW100**: • **PSW100** – кол-во запросов, на которые был получен корректный ответ; • **PSW101** – общее кол-во ошибок обмена; • **PSW102** – не используется; • **PSW103** – адрес устройства, при обмене с которым возникла ошибка или сработал таймаут (см. параметр **Адрес устройства** в настройках компонента).
Управление опросом	В случае установки галочки указывается адрес бита, который будет использоваться для управления опросом данного ПЛК: • **TRUE** – отключить опрос; • **FALSE** – возобновить опрос.

В проекте CODESYS следует:

- добавить компонент **Символьная конфигурация**;
- выделить в нём галочками нужные переменные (см. ниже информацию о поддерживаемых типах), которые должны быть доступны в проекте панели и определить тип доступа (только чтение, только запись, чтение и запись);
- сохранить файл проекта и выполнить команду **Компиляция**;
- в результате в директории сохранения проекта будет создан файл формата **.xml**.

Этот файл необходимо импортировать в конфигуратор с помощью кнопки **Импорт меток**. Импортированные переменные:

- отображаются в библиотеке переменных на вкладке **Переменные устройств**;
- могут быть привязаны к графическим элементам (с учётом соответствия типа переменной типу элемента);
- могут обрабатываться в макросах с помощью соответствующих функций.

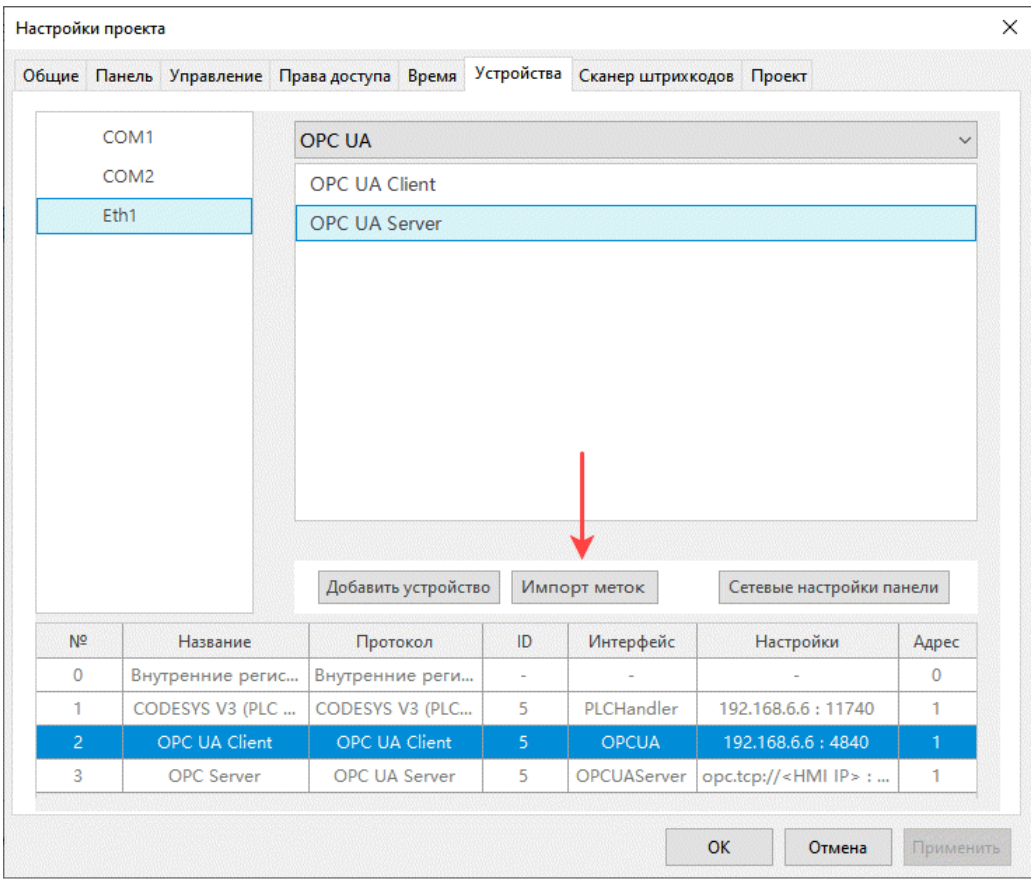


Рисунок 11.11 – Кнопка импорта меток

Таблица 11.12 – Поддерживаемые типы данных

Тип данных CODESYS	Примечание
BOOL	
SINT, INT, DINT, LINT USINT, UINT, UDINT, ULINT BYTE, WORD, DWORD, LWORD	Отдельные биты переменных могут быть привязаны к «битовым» элементам (**Индикатор**, **Переключатель** и т. д.) и параметрам (бит видимости, бит управления и т. д.)
REAL, LREAL	
DATE, TIME, DT, TOD	Отображаются в виде форматированных строк с **IEC-нотацией** без префиксов типов: • DATE: 2026-01-01; • TIME: 49d17h2m47s295ms; • DT: 2026-01-01-11:25:00; • TOD: 23:59:59.999
STRING	Поддерживается только кодировка **ASCII**
WSTRING	Не могут быть привязаны к элементам, доступны только в макросах
Массивы	Поддерживаются одно-, двух- и трехмерные массивы
Структуры и функциональные блоки	Поддерживается произвольное количество уровней вложенности (например, структура, включающая в себя массив структур, полями которых являются другие структуры и массивы)

11.6 OPC UA Client

В режиме **OPC UA Client** панель поддерживает только обмен с контроллерами, программируемыми в среде CODESYS V3.5 и способными работать в режиме **OPC UA Server**. Поддерживается только анонимное подключение без аутентификации и шифрования.

В пределах интерфейса **Ethernet** может быть добавлено до **50** компонентов с различными протоколами, каждый из которых соответствует одному опрашиваемому устройству.

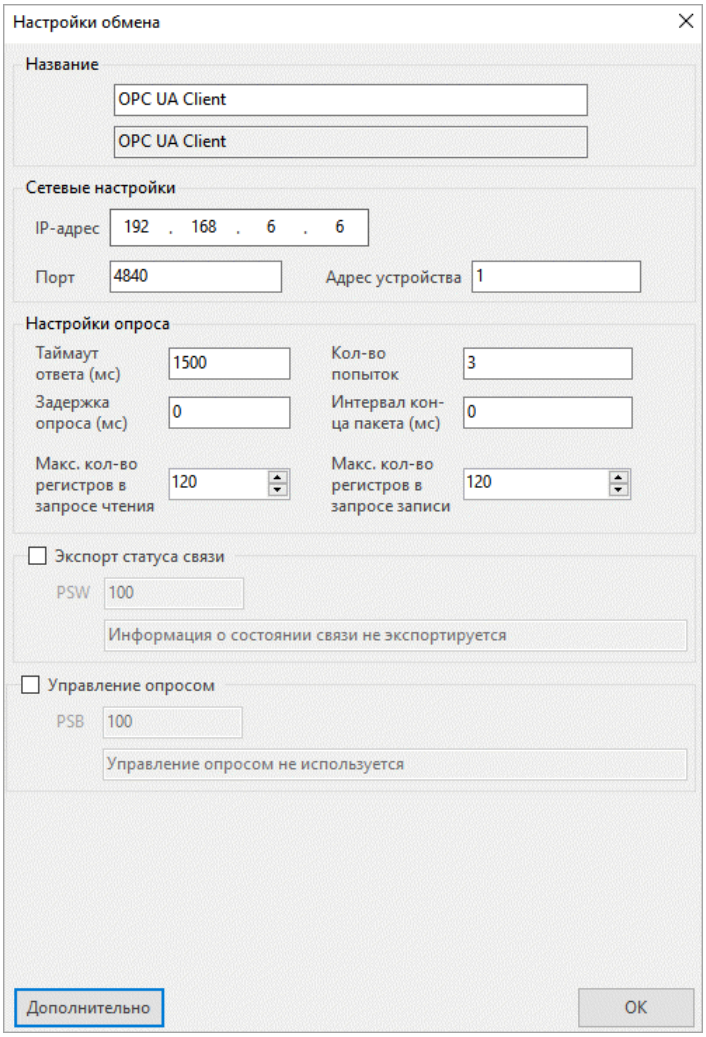


Рисунок 11.12 – Настройки обмена в режиме OPC UA Client

В таблице ниже описаны настройки компонента **OPC UA Client**. Отсутствующие в таблице параметры не используются драйвером протокола.

Таблица 11.13 – Настройка компонента OPC UA Client

Параметр	Описание
Название	Название устройства. Должно быть уникальным в пределах проекта. Используется в макросах
IP-адрес	IP-адрес опрашиваемого устройства
Порт	TCP-порт опрашиваемого устройства
Таймаут ответа	Время ожидания ответа от ПЛК в мс (не более **32767**). Значение **0** соответствует минимально возможному времени ожидания

Продолжение таблицы 11.13

Параметр	Описание
Экспорт статуса связи	В случае установки галочки указывается адрес первого из **4** регистров, используемых для диагностики связи с данным ПЛК. Например, для **PSW100**: • **PSW100** – кол-во запросов, на которые был получен корректный ответ; • **PSW101** – общее кол-во ошибок обмена; • **PSW102** – не используется; • **PSW103** – адрес устройства, при обмене с которым возникла ошибка или сработал таймаут (см. параметр **Адрес устройства** в настройках компонента).
Управление опросом	В случае установки галочки указывается адрес бита, который будет использоваться для управления опросом данного ПЛК: • **TRUE** – отключить опрос; • **FALSE** – возобновить опрос.

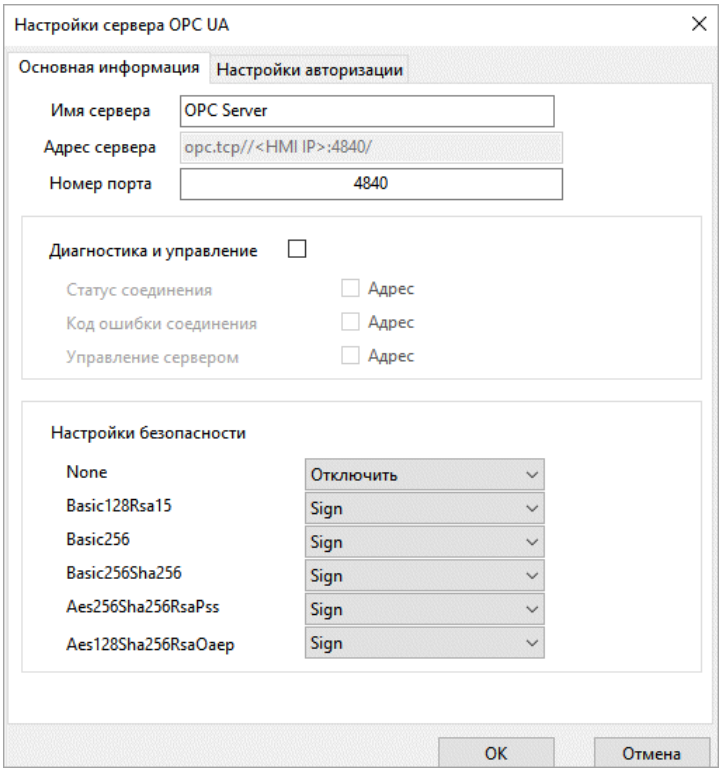
Принцип настройки обмена и поддерживаемые типы данных совпадают с символьном протоколом CODESYS (с тем уточнением, что в настройках компонента **Символьная конфигурация** должна быть установлена галочка **Поддержка OPC UA**).

11.7 OPC UA Server

В рамках интерфейса Ethernet может быть добавлен только один компонент **OPC UA Server**.

В таблице ниже описаны настройки этого компонента.

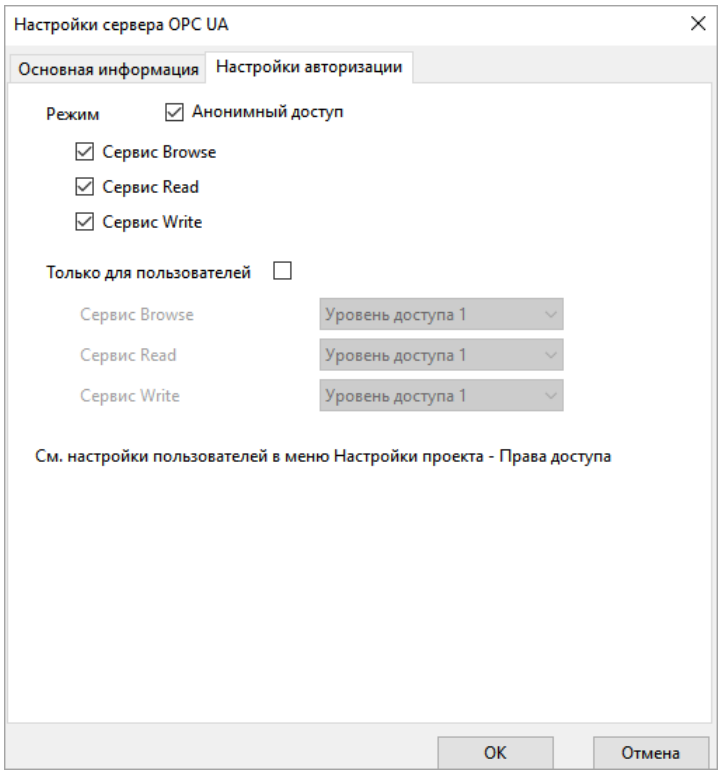
Таблица 11.14 – Настройка компонента OPC UA Server

Параметр	Описание
Основные настройки  **Рисунок 11.13 – Настройки обмена в режиме OPC UA Server**	
Имя сервера	Информационный параметр, не обрабатывается
Адрес сервера	Адрес сервера, который должен быть указан в устройстве-клиенте
Номер порта	Номер порта панели, используемый в режиме **OPC UA Server**

Продолжение таблицы 11.14

Параметр	Описание
Диагностика и управление	В случае установки галочки появляется возможность указать адреса регистров, в которые будут записаны статус соединения и код ошибки, а также адрес регистра, который будет использоваться для управления сервером
Статус соединения	Возможные значения: • **0** – нет активного клиентского подключения; • **1** – есть как минимум одно активное клиентское подключение.
Код ошибки соединения	Возможные значения: • **0** – нет ошибок; • **1** – не удалось установить соединение; • **2** – некорректные настройки; • **3** – сервер остановлен; • **4** – соединение закрыто; • **5** – тестирование; • **6** – ошибка связи; • **7** – неизвестная ошибка.
Управление сервером	Возможные значения: • **0** – инициализация (состояние до первой записи в регистр); • **1** – запуск сервера; • **2** – остановка сервера.
Настройки безопасности	Поддерживаемые точки подключения с соответствующими им алгоритмами шифрования: • None – эта точка позволяет организовать подключение без подписи и шифрования; • Basic128Rsa15; • Basic256; • Basic256Sha256; • Aes256Sha256RsaPss; • Aes128Sha256RsaOaep. Для каждой из точек подключения выбирается один из режимов безопасности сообщений: • **Отключить** – данная точка подключения не будет доступна; • **Sign** – данная точка подключения будет доступна с использованием цифровой подписи; • **Sign&Encrypt** – данная точка подключения будет доступна с использованием цифровой подписи и одновременным шифрованием сообщений; • **Sign и Sign&Encrypt** – данная точка подключения будет доступна как с использованием цифровой подписи без шифрования сообщений, так и с использованием цифровой подписи и одновременным шифрованием сообщений. В панели используется самоподписанный сертификат безопасности

Продолжение таблицы 11.14

Параметр	Описание
Настройки авторизации  **Рисунок 11.14 – Настройки авторизации в режиме OPC UA Server**	
Анонимный доступ	В случае установки галочки допускается анонимное подключение к серверу (без использования логина и пароля). В зависимости от установленных галочек будут доступны сервисы: • Browse (просмотр списка тегов); • Read (чтение значений тегов); • Write (запись значений тегов).
Только для пользователей	В случае установки галочки для подключения к серверу потребуется указать **логин** и **пароль** пользователя, имеющего указанный уровень доступа (см. Настройки проекта – Права доступа). В зависимости от настроек пользователю будут доступны сервисы: • Browse (просмотр списка тегов); • Read (чтение значений тегов); • Write (запись значений тегов). В случае изменения сетевых настроек или логинов/паролей пользователей будет отображено системное окно **25111**. Потребуется перезагрузить панель, чтобы подключиться к серверу с использованием новых значений настроек

После установки настроек сервера отрывается окно **настройки тегов**. В случае установки галочки **Включить** можно добавлять, удалять и редактировать теги и папки – с помощью команд контекстного меню, открываемого по нажатию **ПКМ** на узел дерева, или же с помощью расположенных справа кнопок.

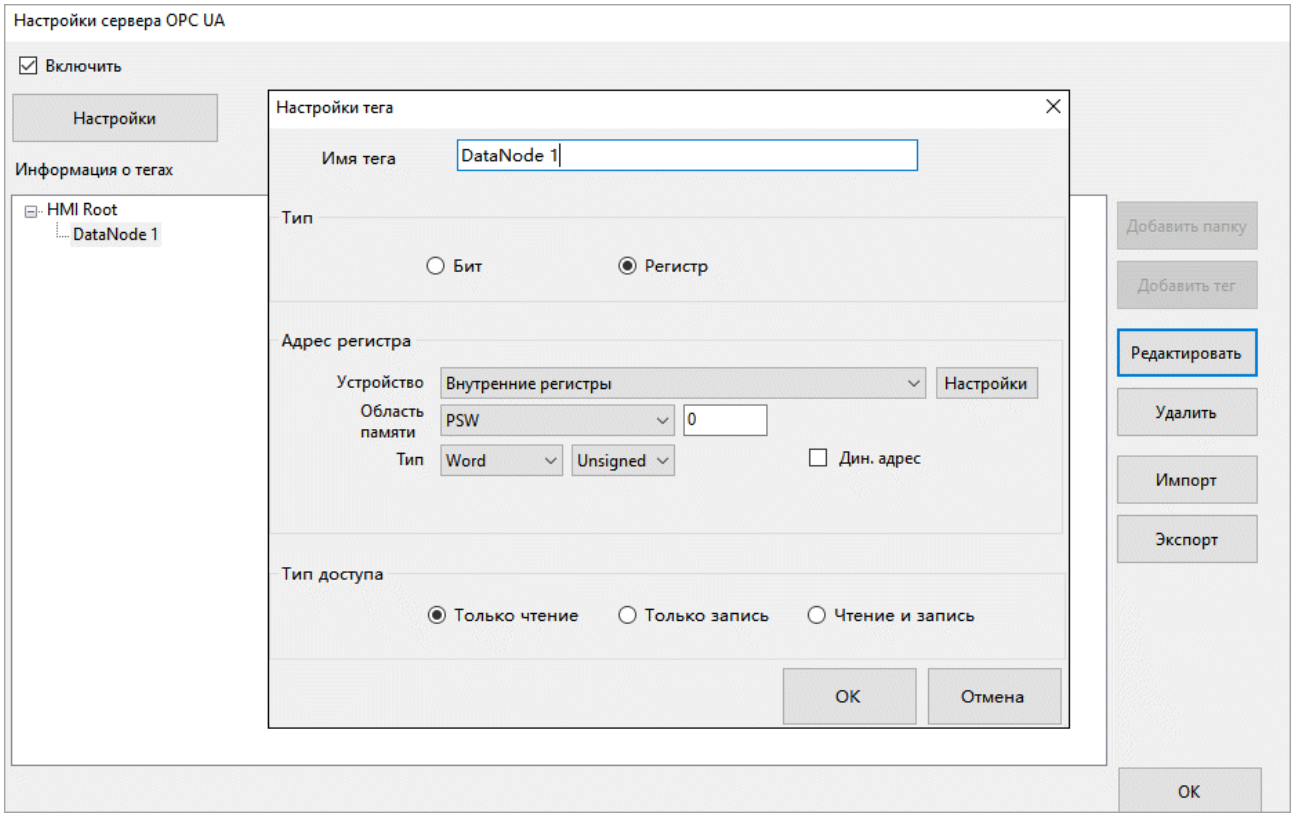


Рисунок 11.15 – Добавление тегов в режиме OPC UA Server

Нажатие на кнопку **Настройки** открывает окно **настроек сервера** (см. таблицу 11.14).

Команда **Экспорт** позволяет выгрузить список тегов в виде файла формата **.csv** или **.xml**. Команда **Импорт** позволяет загрузить файл формата **.csv**.

Доступные настройки тега:

- название (может включать символы кириллицы);
- тип тега – битовый или числовой;
- устройство, область памяти, адрес, тип данных и формат тега;
- тип доступа – чтение, запись или чтение и запись.

Сервер также предоставляет диагностические теги, описанные в спецификации протокола OPC UA.

Таблица 11.15 – Ограничения компонента OPC UA Server

Параметр	Значение
Поддерживаемые сервисы	Browse (просмотр списка тегов) Read (циклическое чтение значений тегов) Write (запись значений тегов) Subscription (получение значений тегов по подписке)
Максимальное количество тегов	10 000
Максимальное количество папок	128
Максимальное количество тегов в папке	255
Максимальное количество одновременных подключений	100

Продолжение таблицы 11.15

Параметр	Значение
Максимальное количество подписок (**subscription**) в рамках клиентского подключения	64
Минимально возможный интервал предоставления данных	100 мс

11.8 Протоколы производителей ПЛК

Настройки протоколов производителей ПЛК (Siemens, Mitsubishi, Omron и т. д.) совпадают с настройками компонентов Modbus RTU/ASCII Master и Modbus TCP Master. Основным отличием является то, что в настройках опрашиваемых параметров указывается не идентификатор области памяти Modbus (который определяет код используемой функции Modbus), а идентификатор области памяти соответствующего контроллера.

11.9 Системные биты/регистры диагностики и управления обменом

1. Режим Master: системные биты и регистры диагностики

В случае отсутствия ответа от slave-устройства в течение таймаута или получения ответа с кодом ошибки системный бит **SPSB48** кратковременно устанавливается в значение **TRUE**. Одновременно с этим устанавливается в **TRUE** системный бит, соответствующий используемому интерфейсу:

- **SPSB49** – интерфейс **COM1**;
- **SPSB50** – интерфейс **COM2**;
- **SPSB52** – интерфейс **Ethernet**.

По переднему фронту бита **SPSB48** происходит открытие системного окна **25003 Ошибки обмена**.

Если системный регистр **SPFW453** имеет значение **0**, то перечисленные биты кратковременно устанавливаются в **TRUE** при каждой ошибке обмена (ошибке таймаута или получения ответа с кодом ошибки). Если системный **SPFW453** имеет значение **1**, то перечисленные биты кратковременно устанавливаются в **TRUE** только спустя **N** последовательных ошибок обмена, где **N** = равно значению параметра **Кол-во попыток**, заданного в настройках опроса. Кроме того, в этом случае в системные регистры **SPSW205-206** (COM1) и **SPSW207-208** (COM2) записывается код ошибки драйвера обмена; эти коды используются разработчиками конфигуратора для отладки.

Для COM-портов доступны следующие параметры диагностики:

Таблица 11.16 – Параметры диагностики COM-портов

Параметр / Интерфейс	COM1	COM2
Кол-во запросов, на которые был получен корректный ответ	SPSW44	SPSW48
Кол-во запросов, на которые был получен ответ с некорректной структурой пакета или кодом ошибки протокола Modbus	SPSW45	SPSW49
Кол-во запросов, на которые не был получен ответ	SPSW46	SPSW50
Адрес последнего устройства, при обмене с которым возникла ошибка или сработал таймаут	SPSW47	SPSW51

Для интерфейса Ethernet аналогичные счетчики могут быть экспортированы в указанные пользователем регистры панели (см. параметр **Экспорт статуса связи**).

2. Режим Master: управление опросом

Для управления опросом slave-устройств, подключенных к COM-порту, используются системные регистры **SPFW400… 415** (COM1) и **SPFW416… 431** (COM2). Для каждого COM-порта выделено **16** регистров; каждый бит этой группы используется для управления опросом slave-устройства с конкретным адресом.

На примере интерфейса COM1:

- **SPFW400.1** – slave-устройство с адресом **1**;
- **SPFW400.2** – slave-устройство с адресом **2**;
- ...
- **SPFW400.15** – slave-устройство с адресом **15**;
- **SPFW401.0** – slave-устройство с адресом **16**;
- ...
- **SPFW415.15** – slave-устройство с адресом **255**.

Для определения номера регистра и бита, используемого для управления обменом со slave-устройством с адресом **X**, можно воспользоваться формулами:

- **Адрес регистра = Offset + (X / 16)**, где **Offset** = **400** (COM1) или **416** (COM2)
- **Адрес бита = остаток от деления (X / 16)**

Запись в бит управления обменом значения **TRUE** останавливает опрос данного slave-устройства, запись значения **FALSE** – возобновляет обмен.

Для интерфейса Ethernet адрес бита управления обменом указывается в настройках slave-устройства в параметре **Управление опросом**.

3. Настройки интерфейсов

Если требуется обеспечить возможность изменения настроек COM-портов с дисплея панели, можно воспользоваться следующими системными регистрами. Новые значения настроек применяются после перезагрузки панели.

Таблица 11.17 – Параметры настроек COM-портов

Параметр / Интерфейс	Возможные значения	COM1	COM2
Адрес опрашиваемого slave-устройства (только для режима Master)		SPFW336	SPFW344
Скорость обмена	48 – 4800 бит/с 96 – 9600 бит/с 192 – 19200 бит/с 384 – 38400 бит/с 576 – 57600 бит/с 1152 – 115200 бит/с 1875 – 187500 бит/с	SPFW337	SPFW345
Кол-во бит данных	8 – 8 бит 7 – 7 бит	SPFW338	SPFW346
Кол-во стоп-бит	1 – 1 бит 2 – 2 бита	SPFW339	SPFW347
Режим контроля четности	0 – None 1 – Odd 2 – Even	SPFW340	SPFW348
Задержка отправки	0…32767 мс	SPFW341	SPFW349

Если требуется обеспечить возможность изменения сетевых настроек панели, то следует воспользоваться следующими системными регистрами:

Таблица 11.18 – Системные регистры сетевых настроек

Параметр	Оперативные системные регистры сетевых настроек	Энергонезависимые системные регистры сетевых настроек
IP-адрес панели	SPSW258…SPSW261	SPFW318…321
Маска подсети панели	SPSW262…SPSW265	SPFW322…325
IP-адрес шлюза	SPSW266…SPSW269	SPFW326…329
DNS-сервер	SPSW270…SPSW273	SPFW331…334

Каждый параметр представлен группой из 4 регистров, каждый регистр определяет один октет параметра.

Если требуется изменять значения сетевых настроек с помощью элементов визуализации, то следует привязать к ним **энергонезависимые** системные регистры. Применение новых настроек происходит после изменения значения любого из регистров.

Если требуется изменить сетевые настройки панели из макроса, то следует:

- записать в регистр **SPSW480** значение **1**, чтобы избежать включения режима DHCP-клиента;
- записать нужные сетевые настройки в **оперативные** системные регистры;
- записать те же самые настройки в **энергонезависимые** системные регистры;
- сгенерировать передний фронт в системном бите **SPSB124.**

ПРИМЕЧАНИЕ
Системный регистр **SPSW480** используется для управления режимом **DHCP-клиента** (**1** – отключить, **0** – включить). Применение значения этого регистра происходит по переднему фронту бита **SPSB124**

Если требуется обеспечить возможность изменения сетевых настроек опрашиваемых slave-устройств с дисплея панели, то можно воспользоваться следующими системными регистрами:

- **SPFW1-4** – IP-адрес Ethernet устройства 1;
- **SPFW5** – номер порта Ethernet устройства 1;
- **SPFW6-9** – IP-адрес Ethernet устройства 2;
- **SPFW10** – номер порта Ethernet устройства 2;
- ...
- **SPFW246-259** – IP-адрес Ethernet устройства 50;
- **SPFW250** – номер порта Ethernet устройства 50.

Новые значения настроек применяются после перезагрузки панели.

11.10 MQTT

Панели СП210-Р могут работать в режиме MQTT-клиента.

Поддерживаемая версия протокола – **3.1.1**.

Для настройки обмена следует подключиться к панели с помощью команды **Инструменты – VNC и MQTT**. Для подключения нужно указать **ID панели** (можно выбрать из сохранённых с помощью кнопки «↓») и пароль удалённого подключения (по умолчанию – **12345678**).

Нажатие на кнопку **Поиск устройств** производит поиск панелей в локальной сети. ПК отправляет UDP-пакет на широковещательный адрес сети (порт **3333**), в результате чего его получают все панели. Результаты сканирования отображаются в виде таблицы с указанием имени панели, IP-адреса, идентификатора и модификации.

Нажатие на кнопку **Тест связи** производит проверку связи с панелью (отправку ping-запроса), после чего отображается соответствующее информационное сообщение.

Нажатие на кнопку **Подключение** осуществляет подключение к панели.

На вкладке **MQTT** следует указать настройки подключения к MQTT-брокеру: адрес брокера (IP или URL), ID клиента (которое будет использоваться в качестве первого уровня имен топиков – см. информацию далее), имя пользователя/пароль (если они используется в брокере) и QoS (запрашиваемый тип гарантии доставки сообщений).

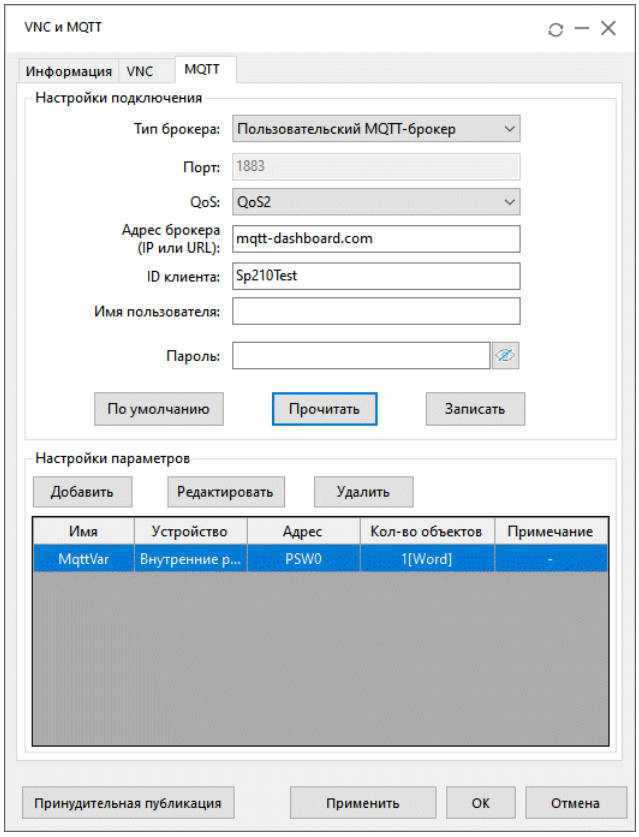


Рисунок 11.16 – Внешний вид вкладки MQTT

Подключение к брокеру всегда осуществляется по стандартному для протокола **MQTT** номеру порта – **1883**.

Возможные варианты **QoS**:

- **QoS0** – без гарантий доставки сообщения брокеру. Панель однократно отправляет сообщение и не ждёт подтверждения получения.

- **QoS1** – с гарантией доставки сообщения брокеру и возможностью дублирования сообщения для подписчиков. Панель отправляет сообщение и ждёт подтверждения получения. В случае отсутствия подтверждения панель повторяет отправку сообщения. При получении каждого сообщения брокер пересылает его подписчикам – т. е. одно и то же сообщение может быть продублировано;
- **QoS2** – с гарантией доставки сообщения брокеру и исключением возможности дублирования сообщения для подписчиков.

В таблице **Настройки параметров** происходит добавление, редактирование и удаление параметров панели, которые будут доступны по MQTT. Кроме того, добавление параметров в эту таблицу происходит при установке галочек **Передавать по MQTT** или **Передавать всё по MQTT** в библиотеке переменных на вкладке **Пользовательские переменные**.

Пользователь может добавить до **500** параметров, при этом параметр может являться массивом.

Таблица 11.19 – Настройки параметров, передаваемых по MQTT

Параметр	Описание
Имя	Имя параметра
Устройство	Выбор источника параметра – память панели или одного из опрашиваемых slave-устройств
Тип	**Bit** (булевское значение) или **Word** (число или строка)
Добавление	В виде одиночного параметра или массива. Для массива указывается количество объектов (не более **126**)
Область памяти	Область памяти, в которой хранится параметр
Начальный адрес	Адрес, начиная с которого размещается значение одиночного параметра или первого параметра массива
Тип данных	Для типа **Bit** поддерживается только тип данных **BOOL**. Для типа **Word** поддерживаются: • Целочисленные типы данных: **Int16, Uint16, Int32, Uint32, Int64**; • Типы данных с плавающей точкой: **Float**, **Double**; • Строковый тип данных: **String** (кодировка ASCII). Для строкового типа указывается длина строки в байтах
Режим публикации	Возможные варианты: • при изменении значения параметра; • периодически в случае выполнения условия (см. ниже; недоступно для массивов и для переменных типа **Bit**); • периодически.
Условие	Доступные условия: • меньше константы Мин.; • больше константы Макс.; • в границах диапазона констант Мин. и Макс. (включая значения границ); • за границами диапазона констант Мин. и Макс.; • не равно константе Мин и константе Макс.
Мин. и Макс.	Константы, используемые в условиях
Период	Интервал публикации в секундах для режимов публикации **По условию и Периодически**
Примечание	Опциональный комментарий к параметру

При нажатии кнопки **Редактировать** открывается окно, в котором доступен поиск по списку добавленных параметров, а также экспорт их настроек в виде файла формата **.xls** или **.xlsx**.

Команда **Импорт** позволяет загрузить этот файл.

Кнопка **Прочитать** позволяет считать текущие настройки из панели. Для записи отредактированных настроек используется кнопка **Записать**. Кнопка **По умолчанию** возвращает все настройки брокера в значения по умолчанию. В случае редактирования параметров помимо нажатия на кнопку **Запись** следует загрузить проект в панель для применения новых настроек.

Кнопка **Принудительная публикация** используется для публикации настроек параметров в топик **pub_configlist** (см. ниже).

Панель использует **5** топиков. В **3** их них она осуществляет публикацию, а ещё на **2** производит подписку. Топики имеют двухуровневные имена.

Первый уровень представляет собой **MQTT-идентификатор**, который совпадает со значением параметра **ID клиента**. Второй уровень определяется типом и назначением топика.

Таблица 11.20 – Список MQTT-топиков

Тип	Имя топика	Описание
Публикация	MQTT-идентификатор/ pub_configlist	Передача информации о настроенных тегах при запуске проекта и по нажатию кнопки **Принудительная публикация** в конфигураторе. Используется флаг **Will Retain**
Публикация	MQTT-идентификатор/ pub_data	Передача значений текущих настроенных тегов
Публикация	MQTT-идентификатор/ write_reply	Подтверждение записи в **/write_data**
Подписка	MQTT-идентификатор/ write_data	Топик для записи значений в теги панели
Подписка	MQTT-идентификатор/ access_data	Топик для внеочередного запроса значений всех параметров (они будут опубликованы в топик **/pub_data**)

Payload топиков представляет собой JSON-объекты в кодировке **UTF-8**. Ниже приведены примеры payload'ов для ситуации, когда в проекте настроено три MQTT-переменные:

- переменная типа Float, размещенная в памяти панели;
- массив из 3 переменных типа Word, размещенных в памяти панели;
- переменная типа Bool, считываемая из slave-устройства, подключенного к панели.

Для всех переменных выбран периодический режим публикации с одинаковым интервалом, поэтому публикация происходит единым «пакетом».

MQTT-идентификатор/pub_configlist

- Unix – метка времени сообщения в формате Unixtime;
- Version – версия протокола панели;
- Configlist – набор устройств.

Каждое устройство содержит следующие параметры:

- Order_name – имя параметра, заданное в конфигураторе в меню **Настройки проекта - Устройства**;
- Order_ID – идентификатор параметра;
- Order_type – тип параметра.

Пример


```
{
 "Unix": "1763081721505",
 "Version": "V1.0",
 "Configlist": {
  "Modbus TCP Master": [
   {
    "Order_name": "ModbusBoolVar",
    "Order_ID": "000002",
    "Order_type": "Bool"
   }
  ],
  "Внутренние регистры": [
   {
    "Order_name": "HmiFloatVar",
    "Order_ID": "000000",
    "Order_type": "Float"
   },
   {
    "Order_name": "HmiWordArrayVar[3]",
    "Order_ID": "000001",
    "Order_type": "INT16U"
   }
  ]
 }
}
```

MQTT-идентификатор/pub_data

- Unix – метка времени сообщения в формате Unixtime;
- Version – версия протокола панели;
- Pub_data – набор устройств.

Каждое устройство содержит названия параметров и соответствующие им значения.

Пример


```
{
 "Variant": [
  {
   "Unix": "1763085137991",
   "Version": "V1.0",
   "Pub_Data": {
    "Modbus TCP Master": {
     "ModbusBoolVar": 1
    },
    "Внутренние регистры": {
     "HmiFloatVar": 11.22,
     "HmiWordArrayVar[3]": [0, 0, 0]
    }
   }
  }
 ]
}
```

MQTT-идентификатор/write_data

- Unix – метка времени сообщения в формате Unixtime (не обрабатывается панелью);
- Version – версия протокола панели;
- Write_data – устройство или набор устройств.

Каждое устройство содержит названия параметров и соответствующие им записываемые значения.

Пример записи одного параметра:

Пример

```
{
 "Unix": "1755237844",
 "Version": "V1.0",
 "Write_Data": {
  "Внутренние регистры": {
   "HmiFloatVar": 22.33
  }
 }
}
```

Пример записи нескольких параметров:

Пример

```
{
 "Unix": "1755237844",
 "Version": "V1.0",
 "Write_Data": {
     "Modbus TCP Master": {
       "ModbusBoolVar": 0
     },
     "Внутренние регистры": {
       "HmiFloatVar": 33.44,
       "HmiWordArrayVar[0]": 11,
       "HmiWordArrayVar[1]": 22,
       "HmiWordArrayVar[2]": 33
     }
 }
}
```

MQTT-идентификатор/write_reply

- Unix – метка времени сообщения в формате Unixtime;
- Version – версия протокола панели;
- Write_reply – устройство или набор устройств.

Каждое устройство содержит названия параметров и соответствующие им статусы:

- OK – запись прошла успешно;
- ERROR1 – не удалось записать значения;
- ERROR2 – переменная с таким именем не найдена;
- ERROR3 – ошибка обработки запроса.

Пример подтверждения записи нескольких параметров:

Пример

```
{
  "Unix": "1755237844",
  "Version": "V1.0",
  "Write_Reply": {
    "Modbus TCP (HMI is master)": {
      "ModbusBoolVar": "OK"
    },
    "Внутренние регистры": {
      "HmiFloatVar": "OK",
      "HmiWordArrayVar": "OK"
    }
  }
}
```

Публикация в этот топик не происходит, если структура объекта, записанного в топик **/write_data** является некорректной – например, когда значение хотя бы одного параметра задано в некорректном формате:

- если в качестве разделителя целой и дробной части используется «,» вместо корректного «.»;
- если значение переменной булевского или числового типа представлено строкой;
- и т. п.

MQTT-идентификатор/access_data

- Unix – метка времени сообщения в формате **Unixtime** (не обрабатывается панелью);
- Version – версия протокола панели;
- Content – команда. В данный момент поддерживается только команда **alldata** – запрос на публикацию всех параметров.

Пример

```
{
  "Unix": "1755237844",
  "Version": "V1.0",
  "Content": "alldata"
}
```

11.11 Сканер штрихкодов

Панели СП210 поддерживают подключение сканера штрихкодов по интерфейсу RS-232 или USB A. В случае использования интерфейса RS-232 данный COM-порт не должен использоваться в меню **Файл – Настройки проекта – Устройства**.

Для настройки обмена следует перейти в меню **Файл – Настройки проекта – Сканер штрихкодов** и установить галочку **Включить**.

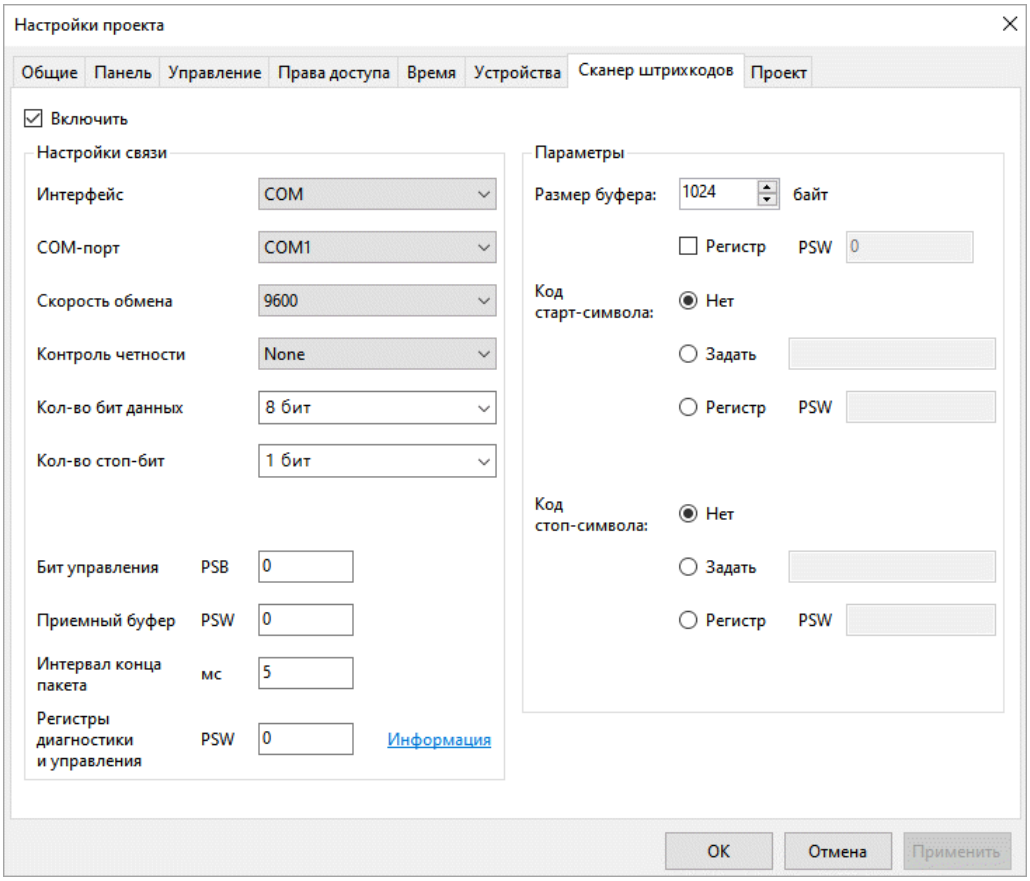


Рисунок 11.17 – Настройки проекта, вкладка Сканер штрихкодов

Таблица 11.21 – Сканер штрихкодов

Параметр	Описание
Настройки связи	
Тип	COM или USB
Интерфейс	Для типа COM нужно указать номер COM-порта панели и выбрать настройки интерфейса RS-232 (скорость, режим контроля четности, количество бит данных, количество стоп-бит)
Параметры	
Размер буфера	Размер приемного буфера в байтах (не более **1024**). Может быть задан константой или путём указания адреса регистра
Код старт-символа (HEX) Код стоп-символа (HEX)	Значения байтов стартового и стопового символа, заданные в HEX константами или путём указания адресов регистров. Могут использоваться совместно или по одному. Панель игнорирует байты полученного пакета, размещенные до старт-символа и после стоп-символа. Если в пакете отсутствует стартовый и/или стоповый символ, то содержимое игнорируется
Управление и диагностика	
Бит паузы	Адрес бита, используемого для остановки приема: • **TRUE** – остановить прием данных • **FALSE** – возобновить

Продолжение таблицы 11.21

Параметр	Описание
Приемный буфер	Адрес начального регистра приемного буфера, в котором будут размещены выделенные из полученного пакета данные. Количество используемых регистров определяется параметром **Размер буфера**, задаваемого в байтах (см. выше)
Интервал конца пакета	Интервал отсутствия новых байтов, по которому определяется факт получения полного пакета данных в мс (не более **65535**)
Регистры диагностики и управления	Адрес первого из **5** регистров, в которые будет записываться диагностическая информация. Предположим, указан регистр **PSW0**, тогда: • 0 – ожидание; • 1 – получен новый пакет. **PSW0** – флаг получения нового пакета: • 0 – ожидание; • 1 – получен новый пакет. **PSW1** – состояние драйвера: • 0 – ожидание; • 1 – некорректное значение одного из параметров; • 2 – пауза (см. бит паузы); • 3 – не удалось открыть порт. **PSW2** – количество полученных байт. **PSW3** – при записи значения 1 происходит очистка буфера и обнуление значений регистров диагностики **PSW4** – код статуса: • 0 – нет ошибок; • 1 – ошибка переполнения буфера; • 2 – старт-символ не обнаружен; • 3 – стоп-символ не обнаружен

12 Макросы

12.1 Основная информация

Макросы – это программы на языке Си, используемые для создания алгоритмов управления и обработки данных. Термин «макрос» для обозначения подобных программ в панелях оператора используется по исторически сложившимся причинам и следует отделять от макросов препроцессора, описанных в стандарте языка Си.

Основной задачей панели оператора является сбор данных с подключенных к ней устройств и предоставление графического интерфейса для их отображения, изменения и архивирования. Тем не менее, в некоторых проектах может потребоваться решать более специфические задачи, например:

- обрабатывать полученные данные – активировать на экране индикатор тревоги в случае превышения контролируемым параметром заданного значения и т. п.;
- иметь полный контроль (на уровне отдельных запросов) над обменом данными с другими устройствами;
- реализовывать нестандартные протоколы обмена для специфических приборов;
- сохранять данные в файл в произвольном формате или вычитывать данные из таких файлов;
- отображать на экране панели геометрические объекты динамической формы и цвета;
- и т.д.

Именно для этого нужны макросы.

Следует учесть, что выполнение макросов является дополнительным функционалом панели, но не основным. Код макросов должен быть как можно более компактным и решать лишь те связанные с работой панели задачи, выполнение которых не удаётся реализовать в ПЛК, ПР или другом подключенном устройстве. В случае необходимости выполнения больших и сложных программ рекомендуется использовать вместо панели оператора линейку панельных контролеров СПК210.

12.2 Работа с макросами в интерфейсе конфигуратора

Для работы с макросами используется одноименный узел дерева проекта. Он содержит две вкладки – **Заголовочные файлы** и **Файлы исходного кода**. Для создания нового файла следует нажать на соответствующую вкладку **ПКМ**, использовать команду **Создать** и указать:

- имя файла (должно быть уникальным в рамках каждой из папок);
- автора;
- описание;
- дату создания.

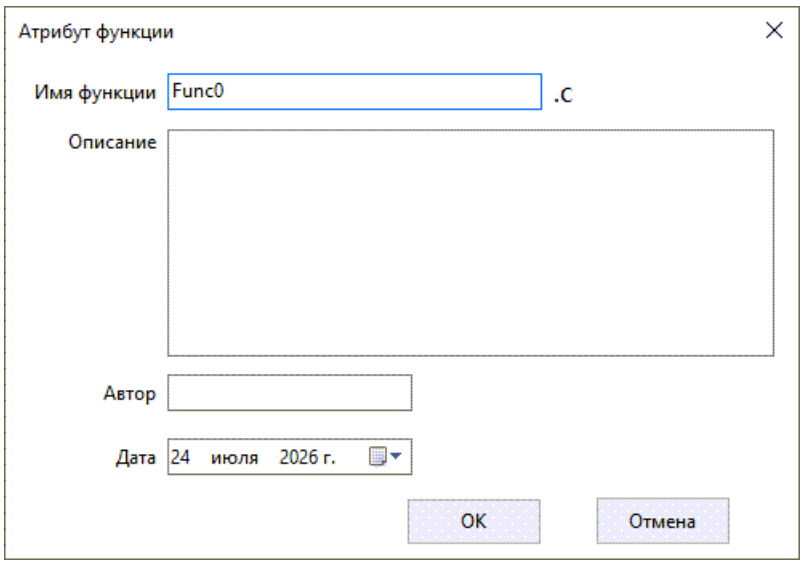


Рисунок 12.1 – Атрибут функции

С помощью команды **Вставить** можно вставить во вкладку файл, ранее скопированный из дерева проекта в буфер обмена.

При нажатии **ПКМ** на файл доступны следующие команды контекстного меню:

- Вырезать (копировать файл в буфер обмена и удалить его из дерева проекта);
- Копировать в буфер обмена;
- Удалить;
- Зашифровать;
- Расшифровать;
- Свойства (переход в редактор кода).

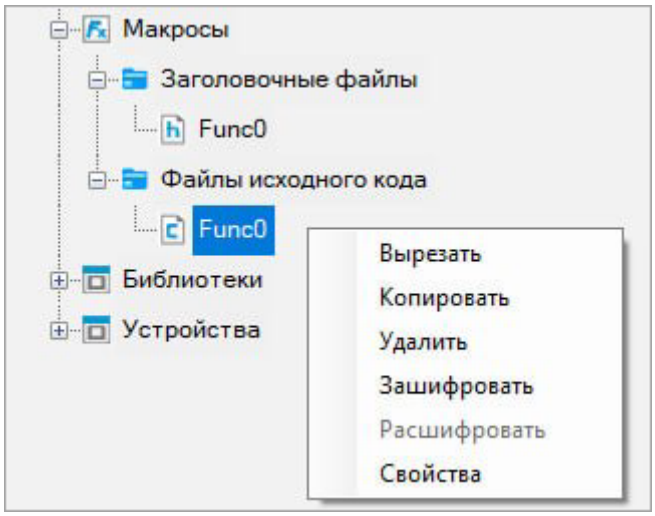


Рисунок 12.2 – Контекстное меню файла макроса

Функционал шифрования позволяет установить пароль, который потребуется ввести для открытия файла. Расшифровка позволяет отключить требование ввода пароля.

Двойное нажатие **ЛКМ** на файл приводит к открытию этого файла в редакторе кода. Редактор состоит из следующих областей:

- панель команд (сверху);
- панель навигации (слева);
- кнопки переключения содержимого панели навигации (слева снизу);

- текстовый редактор (по центру);
- панель сообщений компиляции (снизу).

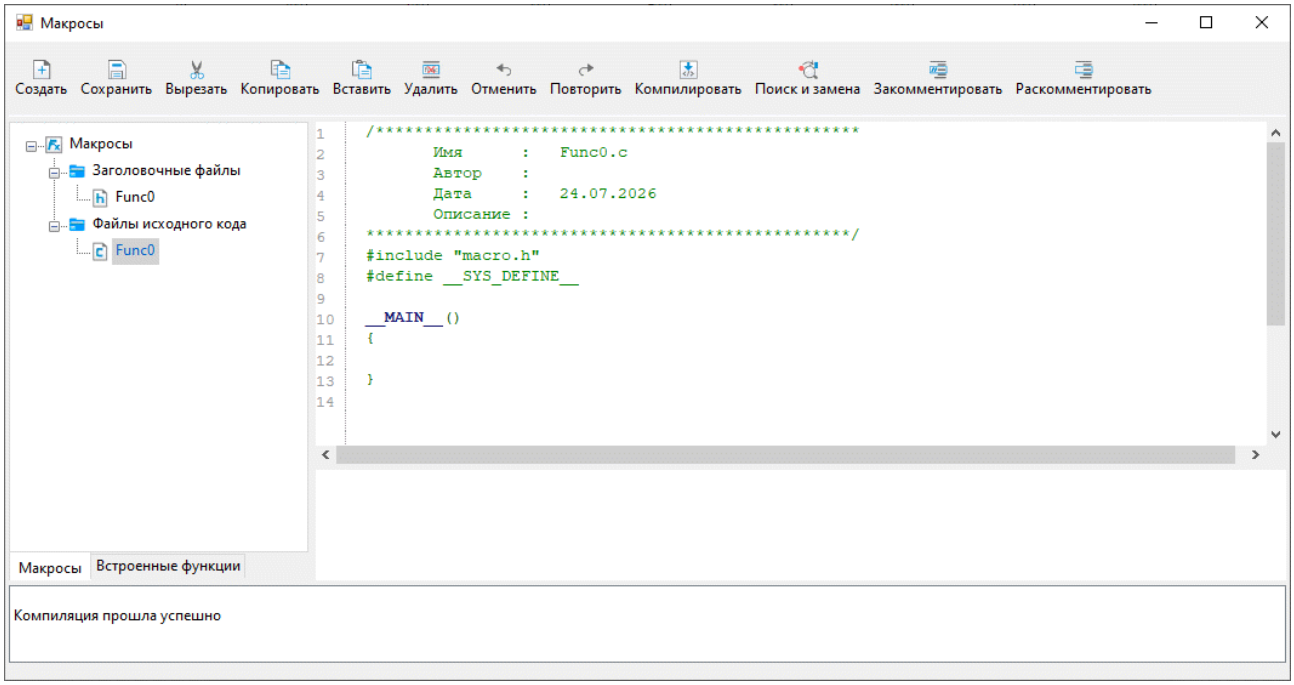


Рисунок 12.3 – Макросы

В панели навигации в зависимости от выбранной вкладки отображается дерево макросов проекта или список встроенных функций и макросов.

Ярлыки панели команд позволяют:

- создать новый файл;
- сохранить файл;
- вырезать файл;
- копировать файл;
- вставить файл;
- удалить файл;
- отменить последнее изменение в выбранном файле (список изменений очищается при открытии другого файла);
- восстановить последнее изменение в выбранном файле (список изменений очищается при открытии другого файла);
- компилировать все файлы;
- осуществить поиск или поиск с заменой по выбранному файлу;
- закомментировать в выбранном файле выделенные строки;
- раскомментировать в выбранном файле выделенные строки.

Компиляция – это процесс трансляции кода, написанном на языке высокого уровня, в объектный код. Объектный код обрабатывается линкером (компоновщиком), который превращает его в машинный код, исполняемый процессором панели.

Для компиляции макросов используется описанная выше команда. В результате компиляции в панели сообщений выводится информация об успешном завершении компиляции или описания возникших в её процессе ошибок.

Кроме того, компиляция макросов происходит в процессе компиляции проекта. В этом случае информация выводится в панель сообщений.

12.3 Вызов макросов

Вызов макросов возможен с помощью следующих элементов:

- Функциональная область (действие **Вызов макроса**);
- Функциональная кнопка (действие **Вызов макроса**);
- Переключатель (вкладка **Макрос**);
- Многопозиционный переключатель (вкладка **Макрос**).

Функциональная область позволяет выполнять макросы циклически или однократно при наступлении определённых событий (например, при переходе на экран, на котором расположена функциональная область). Остальные элементы позволяют однократно выполнить макрос при нажатии и/или отпускании.

В настройках вызова указывается имя файла исходного кода данного макроса и режим выполнения (синхронно или асинхронно).

12.4 Особенности поддержки языка Си в панелях СП210

Таблица 12.1 – Особенности поддержки языка Си в панелях СП210

Параметр	Значение
Версия стандарта языка Си	C99
Компилятор, используемый в панели	arm-linux-gnueabihf-gcc version 4.9.4 (32-bit)
Компилятор, используемый в режиме эмуляции	MinGW (32-bit) with gcc version 4.5.2
Флаги компиляции	O0 -std=c99
Поддержка стандартной библиотеки языка Си	Полная
Директория монтирования USB-накопителя	**/mnt/udisk/** В режиме эмуляции можно работать с файлами на ПК (при условии, что эти файлы доступны для пользователя, от имени которого запускается конфигуратор). В этом случае пути к файлам должны экранироваться: **C:\\test_folder\\test_file.txt**

12.5 Базовая информация о языке C99

Цель пункта – познакомить читателя с основными элементами языка и привести примеры их использования; но он не является полноценным справочным руководством по языку. В данном пункте приведена базовая информация о языке C99 применительно к его реализации в панелях оператора СП210. В пункте не рассмотрены некоторые возможности языка, которые в большинстве случаев не потребуется применять при создании макросов для панелей оператора:

- рекурсивный вызов функций;
- работа с комплексными числами;
- функционал препроцессора (кроме самых базовых команд);
- создание функций с переменным количеством параметров;
- динамическое выделение памяти;
- и т. д.

Если вы не знакомы с языком Си, то рекомендуется использовать дополнительные справочные материалы – книги (например, *Стивен Прата. Язык программирования С. Лекции и упражнения*), интернет-ресурсы и т. п.

Последняя черновая версия стандарта C99 (отличающаяся от релизной только незначительными изменениями в оформлении) доступна по ссылке. По ней же доступен документ C99 Rationale, в

котором объясняются причины, мотивы и обоснования особенностей реализации различных элементов языка (в том числе, изменений и нововведений, добавленных в версии C99).

Список специфических функций и макросов, доступных для панелей СП210, приведен в пункте 12.6.8.

12.5.1 Файлы исходного кода

В языке Си единицей трансляции является файл исходного кода, который имеет расширение **.c**. Созданный в конфигураторе файл выглядит по умолчанию следующим образом:

Пример

```
/*************************************************
   Имя    : Func0.c
   Автор  :
   Дата   : 11/17/2025
   Описание:
*************************************************/
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{


}
```

Файл включает в себя:

- заголовок, сформированный на основе информации, указанной при создании файла;
- директиву **#include**, подключающую системный заголовочный файл **macro.h**;
- директиву **#define**, определяющую макрос препроцессора **__SYS_DEFINE__** (не представляет интереса для пользователя);
- функцию **__MAIN__**, по умолчанию не имеющую реализации (аналог функции **main**, описанной в стандарте языка Си). Её код создается пользователем.

ПРИМЕЧАНИЕ
Шаблон создаваемого файла можно изменить путём редактирования файла \Tool\Macro \Template\UFunHeader.txt в директории установки конфигуратора.

При вызове макроса фактически происходит выполнение функции __MAIN__, являющейся точкой входа в программу. Таким образом, каждый файл исходного кода в дереве проекта соответствует одной программе. Помимо директив и функции __MAIN__ файл исходного кода может содержать:

- объявления и определения других функций;
- объявления и определения глобальных переменных;
- определения структур, объединений, перечислений и псевдонимов.

Таким образом, в рамках функции **__MAIN__** могут вызываться другие функции, созданные пользователем, или размещённые в подключаемых через директиву **#include** библиотеках.

12.5.2 Заголовочные файлы

Объявление и определение – это две важные концепции языка Си. Объявление – это способ сообщить компилятору о существовании объекта, а определение – способ создать объект и выделить память под его хранение.

Заголовочные файлы в основном содержат объявления (интерфейс), а файлы исходного кода – определения (реализацию).

Расширение заголовочных файлов - **.h**.

При создании языка Си введение концепции заголовочных файлов позволило обеспечить возможность раздельной компиляции файлов исходного кода; кроме того, их использование позволяет повысить степень модульности проекта.

Заголовочные файлы могут включать:

- директивы препроцессора;
- объявления внешних (extern) переменных;
- объявление и определение структур, объединений, перечислений и псевдонимов;
- объявление (прототипы) функций.

При создании макросов вы можете обойтись без использования заголовочных файлов. В случае их использования – предпримите усилия, чтобы в рамках файла исходного кода один и тот же заголовочный файл подключался не более одного раза (с помощью директив **#ifndef** или **#pragma once**).

12.5.3 Типы данных

Типы данных языка Си можно разделить на следующие категории:

- тип bool;
- целочисленные;
- значения с плавающей точкой;
- указатели;
- массивы (и их частный вид – строки);
- структуры;
- объединения;
- перечисления;
- псевдонимы;
- тип void.

В следующих подпунктах рассмотрены основные свойства этих типов. Для наглядности их описание сопровождается синтетическими примерами работы с переменными данных типов. Различные аспекты этих примеров (инициализация переменных, операторы, функции и т. д.) будут рассмотрены в следующих пунктах. В комментариях иногда размещен псевдокод – неформальный язык, применяемый для пояснения выполняемых операций и использующий ключевые слова популярных языков программирования.

1. Тип bool

Логический тип данных используется для представления значения, которое имеет только два возможных состояния – true (1) или false (0). Этот тип применяется для описания бинарных событий и команд (например, отсутствие/наличие тревоги, отключение/включение насоса и т. д.). Значения данного типа поддерживают логические операции.

Переменная типа bool занимает 1 байт памяти.

ПРИМЕЧАНИЕ
Для использования макросов **true** и **false** требуется подключить заголовочный файл **stdbool.h**.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdbool.h"

__MAIN__()
{
   bool BoolVar;

   BoolVar = GetPSB(0) && GetPSB(1);
   SetPSB(2, BoolVar);
   // Эквивалентный псевдокод: PSB2 = PSB0 AND PSB1

   SetPSB(3, true);
   // Эквивалентный псевдокод: PSB3 = true
}
```

2. Целочисленные типы

Целочисленные типы отличаются друг от друга наличием/отсутствием знака и диапазоном значений. Значения этих типов поддерживают арифметические и побитовые операции, а также операции сравнения.

В таблице указаны размеры типов применительно к панели и её эмулятору; на других платформах размер может отличаться. Наличие нескольких вариантов именования для каждого типа связано с некоторыми историческими особенностями языка Си и наличием в панели псевдонимов для этих типов.

При выполнении арифметических операций возможен эффект переполнения – например, если переменная типа **uint16_t** имеет значение **65535** (это верхний предел ее диапазона) и мы прибавим к ней **1**, то переменная получит значение **0** (т. е. ее значение станет равным нижней границе диапазона ее значений). Для панелей оператора СП210 переполнение и беззнаковых, и знаковых типов обрабатывается одним и тем же способом.

Деление на **0** приведет к исключению, в результате чего выполнение всех макросов прекращается до перезагрузки панели или перезапуска эмулятора.

ПРИМЕЧАНИЕ

Типы с постфиксом **_t** является типами фиксированного размера, для которых гарантируется совпадение размера на всех платформах. Для их использования требуется подключить заголовочный файл **stdint.h**.

Таблица 12.2 – Целочисленные типы данных

Тип	Диапазон	Размер в байтах
char INT8 int8_t	-128…127	1
unsigned char BYTE UINT8 uint8_t	0…255	
short INT16 int16_t	-32768… 32767	2
unsigned short WORD UINT16 uint16_t	0…65535	
int long INT32 int32_t	$-2^{31}…2^{31}-1$	4
unsigned int unsigned long DWORD UINT32 uint32_t	$0…2^{32}$	
long long INT64 int64_t	$-2^{63}…2^{63}-1$	8
unsigned long long DDWORD UINT64 uint64_t	$0…2^{64}$	

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
   uint16_t u16_var1;
   uint16_t u16_var2 = 65535;
   int16_t  i16_var;

   uint32_t u32_Var1;
   uint32_t u32_Var2 = 123456;



   u16_var1 = PSW[0] + PSW[1];
   PSW[2] = u16_var1;
   // Можно сократить до PSW[2] = PSW[0] + PSW[1];

   PSW[3] = u16_var2 + 1;
   // Беззнаковое переполнение: в PSW[3] будет записано значение 0

   i16_var = 32767 + 1;
   PSW[4] = i16_var;
   // Знаковое переполнение: в PSW[3] будет записано значение -32768
   // (нижняя граница типа int16_t)

   u32_Var1 = *(uint32_t*)(PSW + 10);
   *(uint32_t*)(PSW + 12) = u32_Var2;
   // Чтение и запись параметра типа DWord
}
```

3. Типы данных с плавающей точкой

Значения этих типов поддерживают арифметические операции и операции сравнения. В реализации панелей оператора тип **long double** по диапазону и точности соответствует типу **double**, так что его использование не имеет смысла.

Значение с плавающей точкой является лишь приближением к реальному значению с ограниченной точностью. Поэтому такие значения нет смысла проверять на строгое равенство – вместо этого следует вычислять их разность, и если в рамках конкретной задачи она пренебрежимо мала, то считать их «равными».

Значения с плавающей точкой могут иметь особые состояния – такие, как **NaN** и бесконечность. Для проверки, является ли значение особым, используются макросы заголовочного файла **math.h**

Таблица 12.3 – Целочисленные типы данных

Тип	Диапазон	Точность	Размер в байтах
float	$10^{-38}…10^{38}$	7-8 знаков после точки	4
double	$10^{-308}…10^{308}$	15-17 знаков после точки	8
long double			12

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  float FloatVar1;
  float FloatVar2;

  // Чтение и запись параметра формата Float
  FloatVar1 = *(float*)(PSW + 20);
  FloatVar2 = FloatVar1 + 0.0001;
  *(float*)(PSW + 22) = FloatVar2;

  // Начиная с FloatVar1 = 2048 приращение 0.0001...
  // ... не будет происходить из-за ограничения точности.
  // Это называется «эффект расходимости мантиссы».
  // Существуют различные алгоритмы для его компенсации.
  // См., например, алгоритм Кэхэна

}
```

4. Указатели

Переменная типа «указатель» содержит адрес другой переменной (*или функции – но об этом лучше прочитать в учебной литературе*). Указатели обеспечивают гибкую и эффективную работу с памятью, и поэтому активно используются в коде стандартной библиотеки Си (и других библиотек). Они позволяют создавать функции, которые способны обрабатывать и модифицировать наборы данных произвольного типа и размера.

Для панелей оператора СП210 размер указателей составляет 4 байта, так как аппаратная платформа является 32-битной.

Указатели поддерживают арифметические операции и специфические операции получения адреса (&) и разыменования – доступа к значению, расположенному по адресу (*). Частой ошибкой является разыменование неинициализированного указателя («нулевого» или имеющего некорректное значение, не совпадающее с доступным адресом памяти). Результатом подобной операции является возникновение исключения, которое делает невозможным продолжение нормальной работы панели до её перезагрузки.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  // Синтетический пример использования указателя

  uint16_t value = 123;
  // Указатель на тип uint16_t
  uint16_t *pointer;

  // Получение адреса переменной value
  pointer = &value;
  // Доступ к значению, размещенному под указателем
  // В PSW[30] будет записано значение переменной value
  PSW[30] = *pointer;

  // Можно сделать и наоборот – записать в переменную value значение регистра PSW[30]
  *pointer = PSW[30];
}
```

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  uint16_t ui16_var;
  // Функция Read читает значение из памяти панели или slave-устройства...
  // ...и размещает его в переменной
  // Так как функция должна поддерживать чтение значения произвольного типа и размера...
  // ...то на её последний вход следует передавать адрес переменной...
  // ...в которую это значение будет записано
  Read(_T("Внутренние регистры"), 0, TP_PFW, DT_Word, 30, 0, &ui16_var);
}
```

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  float FloatVar1;

  // Для доступа к параметру типа Float, размещенного в регистрах панели...
  // ...используется каламбур типизации

  FloatVar1 = *(float*)(PSW + 20);

  // PSW + 20 – это адрес регистра панели PSW20 (фактически – указатель)
  // (float*) заставляет компилятор считать, что это указатель на значение типа Float
  // (тот же синтаксис, что при объявлении переменной типа указатель)
  //
  // *(float*) – разыменовывает это указатель, чтобы получить лежащее в регистрах панели...
  // ... значение типа Float
}
```

5. Массивы

Массив объединяет набор объектов одного и того же типа. Данные массива размещаются в памяти последовательно. Для доступа к элементу массива используется оператор []. Индексация массивов ведется с **0**.

Массивы часто обрабатываются в цикле **for**:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"
__MAIN__()
{
  uint16_t Data[4];

  // Размещаем в массиве значения регистров панели PSW100, PSW200, PSW300 и PSW400...
  //  ...для какой-то последующей обработки
  for (int i = 0; i < 4; i++)
  {
    Data[i] = PSW[100 * (i + 1)];
  }
}
```

Размеры массивов удобно задавать через макроопределения или константы – иначе в случае их изменения легко забыть внести исправление в какой-нибудь фрагмент кода, где они используются.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Макроопределение
#define ARR_SIZE 4

__MAIN__()
{
  uint16_t Data[ARR_SIZE];

  // Размещаем в массиве значения регистров панели PSW100, PSW200, PSW300 и PSW400...
  //  ...для какой-то последующей обработки
  for (int i = 0; i < ARR_SIZEi++)
  {
    Data[i] = PSW[100 * (i + 1)];
  }
}
```

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Константа
const int16_t ARR_SIZE = 4;

__MAIN__()
{
  uint16_t Data[ArrSize];

  // Размещаем в массиве значения регистров панели PSW100, PSW200, PSW300 и PSW400...
  //  ...для какой-то последующей обработки
  for (int i = 0; i < ARR_SIZE; i++)
  {
    Data[i] = PSW[100 * (i + 1)];
  }
}
```

Если при инициализации массива не указать его размер – то компилятор определит его автоматически на основании числа инициализаторов.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  // Будет выделена память под 3 значения
  uint16_t Data[] = {1, 2, 3};
}
```

Можно инициализировать лишь нужные элементы массива – остальные автоматически будут инициализированы нулями.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Макроопределение
#define ARR_SIZE 4

__MAIN__()
{
  // После инициализации массив примет вид {1, 0, 0, 4}
  uint16_t Data[ARR_SIZE] = {[0] = 1, [3] = 4};
}
```

Доступны массивы переменной длины (variable-length array; **VLA**) – их размер может меняться в процессе работы программы. В этом случае при доступе к элементу массива следует контролировать, что индекс соответствует текущему размеру массива. Для **VLA** не поддерживается инициализация и использование модификатора **static**.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  uint16_t ArrSize;
  uint16_t VLA_Data[ArrSize];
  uint16_t CurrentIndex;

  ArrSize = PSW[51];
  CurrentIndex = PSW[52];

  if (ArrSize != 0 && CurrentIndex <= ArrSize)
  {
    VLA_Data[CurrentIndex] = 42;
  }

}
```

Массивы могут быть многомерными:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  // массив 2x3x4
  uint16_t arr3d[2][3][4] =
  {
    {  // Первый «блок»
      {1, 2, 3, 4},     // 3 «строки» по 4 значения
      {5, 6, 7, 8},
      {9, 10, 11, 12}
    },
    {  // Второй «блок»
      {13, 14, 15, 16},
      {17, 18, 19, 20},
      {21, 22, 23, 24}
    }
  };

  // В регистр PSW53 будет записано значение 7
  // (напомним, нумерация массивов ведется с 0)
  PSW[53] = arr3d[0][1][2];
}
```

Имя массива является указателем на его начальный элемент. Поэтому при использовании функций Reads/Writes (и других функций, принимающих на вход указатель) необязательно использовать оператор получения адреса (&).

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

#define ARR_SIZE 4

__MAIN__()
{
  uint16_t ui16_var;
  uint16_t Data[ARR_SIZE];

  // Для обычной переменной оператор & необходим
  Read(_T("Local Device"), 0, TP_PFW, DT_Word, 30, 0, &ui16_var);

  // А вот для массива подойдет любой из вариантов:
  Reads(_T("Local Device"), 0, TP_PSW, DT_Word, 0, 0, ARR_SIZE, &Data);
  Reads(_T("Local Device"), 0, TP_PSW, DT_Word, 0, 0, ARR_SIZE, Data);
}
```

Зная о родстве массивов и указателей – можно оценить, как в языке Си реализованы арифметические операции над последними.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

#define ARR_SIZE 4

__MAIN__()
{
  uint16_t arr[ARR_SIZE] = {10, 20, 30, 40};
  uint16_t *ptr = arr;

  // ptr указывает на arr[0]
  PSW[54] = *ptr;

  ptr = ptr + 1;

  // ptr указывает на arr[1]
  PSW[55] = *ptr;

}
```

То есть увеличение значения указателя на 1 «смещает» его на количество байт, равное типу указателя: «ptr + n» эквивалентно «ptr + n * sizeof(*ptr)».

Кроме того, указатели, как и массивы, поддерживают индексный доступ.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

#define ARR_SIZE 4

__MAIN__()
{
  uint16_t arr[ARR_SIZE] = {10, 20, 30, 40};
  uint16_t *ptr = arr;

  // ptr[0] соответствует arr[0]
  PSW[54] = ptr[0];

  // ptr[1] соответствует arr[1]
  PSW[55] = ptr[1];

}
```

6. Строки

В языке Си нет отдельного типа данных для строк. Вместо этого строки реализованы как массивы символов. В стандарте C99 доступны два вида символов:

Таблица 12.4 – Виды символов в C99

Тип	Источник	Кодировка	Размер символа
char	Встроенный тип	ASCII	1 байт
wchar_t	Заголовочный файл wchar.h	UTF-16	2 байта

Строка в кодировке ASCII представляет собой массив char, а строка в кодировке UTF-16 – массив wchar_t. Строки в языке Си являются нуль-терминированными – каждая строка должна заканчиваться символом, имеющим код «**0**» (NULL-терминатором). При инициализации строки этот символ добавляется автоматически. Строковые литералы обрамляются двойными кавычками (в отличие от отдельных символов, обрамляемых одинарными кавычками). Для UTF-16 значение литерала предваряется префиксом L.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "wchar.h"

__MAIN__()
{
  // Выделенный размер – 1 байт
  char AsciiChar = 'v';
  // Выделенный размер – 5 байт (4 байтовых символа + NULL-терминатор)
  char AsciiStr[] = "test";
  // Выделенный размер – 2 байта
  wchar_t UnicodeChar = L'я';
  // Выделенный размер – 10 байт (четыре 2-байтовых символа + 2-байтовый NULL-терминатор)
  wchar_t UnicodeStr[] = L"тест";
}
```

С помощью символа «\» можно экранировать управляющие последовательности и HEX-коды символов. Это, в частности, позволяет инициализировать строки с использованием кодировки Win-1251.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Строка «тест» (кодировка Win-1251)
  char AsciiStr[] = "\xF2\xE5\xF1\xF2";
  // Перенос строки
  char AsciiStr[] = "line 1\nline 2";
}
```

Для работы с символами и строками используются следующие заголовочные файлы:

Таблица 12.5 – Заголовочные файлы для работы с символами и строками

Категория	Тип	Заголовочные файлы
Символ	char	ctype.h
	char[]	string.h stdlib.h
Строка	wchar_t	wctype.h
	wchar_t[]	wchar.h

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"
#include "string.h"

__MAIN__()
{
  char FullStr[20] = "";
  char str1[10] = "Hello, ";
  char str2[10] = "World!";
  uint16_t FullStrLen;

  // Записываем в FullStr результат объединения FullStr и str1
  // Из str1 используется не более 10 символов
  strncat(FullStr, str1, 10);
  // Записываем в FullStr результат объединения FullStr и str2
  // Из str2 используется не более 10 символов
  strncat(FullStr, str2, 10);
  // Вычисляем количество символов строки FullStr
  FullStrLen = strlen(FullStr);

  // Записывем строку FullStr в регистры панели начиная с PSW60
  Writes(_T("Внутренние регистры"), 0, TP_PSW, DT_Word, 60, 0, 1 + FullStrLen / 2, FullStr);

}
```

В заголовочном файле **string. h** также объявлены функции для работы с памятью, которые часто используются применительно к массивам – memcpy, memmove, memcmp, memset и др.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "string.h"

__MAIN__()
{
  // Записываем в регистры PSW56-59 значение 123
  // Регистр занимает 2 байта, поэтому 2 * 4
  memset(&PSW[56], 123, 2 * 4);
}
```

7. Структуры

Структура объединяет набор объектов разных типов. Каждый объект размещается в отдельных ячейках памяти.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Объявление структуры
struct MessageBox
{
  char title[30];
  char message[200];
  uint8_t IconCode;
  uint16_t x;
  uint16_t y;
};



__MAIN__()
{
  // Определение структуры с полной инициализацией полей
  struct MessageBox ErrorMessageBox1 = {"error", "Some error text", 0, 100, 100};

  // Определение структуры с частичной инициализацией полей
  // Остальные поля будут инициализированы нулями
  struct MessageBox ErrorMessageBox2 = {.title = "Error",
    .message = "Some error text"};

  // Доступ к полям структуры в коде
  ErrorMessageBox2.x = 100;
  ErrorMessageBox2.y = 100;

}
```

Можно совместить объявление структуры с её определением:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Объявление структуры
struct MessageBox
{
  char title[30];
  char message[200];
  uint8_t IconCode;
  uint16_t x;
  uint16_t y;
} ErrorMessageBox2;



__MAIN__()
{
  // Доступ к полям структуры в коде
  ErrorMessageBox2.title = "Error";
  ErrorMessageBox2.message = "Some error text";
  ErrorMessageBox2.x = 100;
  ErrorMessageBox2.y = 100;
}
```

Для доступа к полю структуры через указатель на структуру используется специфический оператор «->»:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Объявление структуры
struct MessageBox
{
  char title[30];
  char message[200];
  uint8_t IconCode;
  uint16_t x;
  uint16_t y;
} ErrorMessageBox2;



__MAIN__()
{
  struct MessageBox *PtrToMessageBox = &ErrorMessageBox2;

  // Доступ к полям структуры через указатель
 // Для полей-массивов использование оператора «->» не требуется...
 // ...потому что имя массива - указатель на его начальный элемент
  PtrToMessageBox.title = "Error";
  PtrToMessageBox.message = "Some error text";
  PtrToMessageBox->x = 100;
  PtrToMessageBox->y = 100;
}
```

В языке Си присутствует поддержка битовых полей (bitfield) – разновидностей структур, содержащих только целочисленные поля, для каждого из которых указано количество занимаемых бит.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Объявление структуры с битовыми полями
struct MERCURY23x_MODEL_PARAMS_2
{
  uint8_t CounterConstant : 4;
  uint8_t PhaseCount : 1;
  uint8_t IsAveragePowerProfile : 1;
  uint8_t TempRange : 1;
  uint8_t DirectionCount : 1;
};
```

4.4.16 Вариант исполнения счетчика

4.4.16.1 Устаревший

Команда предназначена для чтения варианта исполнения счетчика.
Номер параметра – 12h. Поле параметров отсутствует.
Уровень доступа – 1, 2.
Поле данных ответа состоит из 6 или 8 двоичных байт, формат которых приведен на рисунке 4.33.

№ байта ответа	7	6	5	4	3	2	1	0
1-й	Cl A – класс точности по активной энергии: 0 – 0,2S; 1 – 0,5S; 2 – 1,0; 3 – 2,0		Cl R – класс точности по реактивной энергии: 0 – 0,2; 1 – 0,5; 2 – 1,0; 3 – 2,0		Uн – номинальное напряжение: 0 – 57,7 В; 1 – 230 В		Iн – номинальный ток: 0 – 5 А; 1 – 1 А; 2 – 10 А	
2-й	Число направлений: 0 – 2; 1 – 1	Температурный диапазон: 0 – -20°С; 1 – -40°С	Учет профиля средних мощностей: 0 – нет; 1 – да	Число фаз: 0 – 3; 1 – 1	Постоянная счетчика: 0 – 5000 имп/кВт·ч; 1 – 25000 имп/кВт·ч; 2– 1250 имп/кВт·ч; 3 – 500 имп/кВт·ч; 4 – 1000 имп/кВт·ч; 5 – 250 имп/кВт·ч			
3-й	Суммирование фаз:	Тарификатор:	Тип счетчика: 0 – AR;		№ варианта исполнения (см. рисунок 4.34)			

Рисунок 12.4 – Фрагмент спецификации протокола Меркурий

При использовании структур можно столкнуться с выравниванием памяти (memory aligment). Рассмотрим влияние выравнивания на простом примере. Пусть у нас определена структура с двумя полями, одно из которых имеет тип **uint8_t**, а второе – **uint32_t**.

Пример

```
struct TestStuct
{
  uint8_t  var1;
  uint32_t var2;
} ts
```

Первое поле занимает 1 байт, второе 4 – и логично предположить, что вся структура будет занимать 5 байт. Но с помощью оператора **sizeof** можно убедиться, что в реальности в панели структура занимает 8 байт (для других платформ результат может быть иным):

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

struct TestStuct
{
  uint8_t  var1;
  uint32_t var2;
} ts



__MAIN__()
{

  // В PSW80 будет записано значение 8
  PSW[80] = sizeof(ts);

}
```

Наблюдаемый эффект связан с выравниванием памяти. В статье на Википедии приведен необходимый минимум для понимания этой концепции (а в англоязычной версии статьи есть конкретные примеры):

«Центральные процессоры в качестве основной единицы при работе с памятью используют машинное слово, размер которого может быть различным. Однако размер слова всегда равен нескольким байтам (байт является наименьшей единицей, в которой отсчитываются адреса). Как

правило, машинное слово равно 2k байтам, то есть состоит из одного, двух, четырёх, восьми и т. д. байтов.

При сохранении какого-то объекта в памяти может случиться, что некое поле, состоящее из нескольких байтов, пересечёт “естественную границу” слов в памяти. Некоторые модели процессоров не могут обращаться к данным в памяти, нарушающим границы машинных слов. Некоторые могут обращаться к невыровненным данным дольше, нежели к данным, находящимся внутри целого “машинного слова” в памяти».

В некоторых ситуациях требуется управлять выравниванием памяти (например – отключить его); это, например, может быть удобным при реализации протоколов, когда структуру требуется «наложить» на байтовый буфер запроса или ответа.

Для этого существуют различные средства, но проще всего воспользоваться директивой **pack**:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Сохраняем текущее выравнивание и устанавливаем 1-байтовое
#pragma pack(push, 1)
struct TestStuct
{
  uint8_t  var1;
  uint32_t var2;
} ts
// Восстанавливаем сохраненное ранее выравнивание
#pragma pack(pop)

// На эту структуру директива pack уже не действует
struct TestStuct2
{
  uint8_t  var1;
  uint32_t var2;
} ts2;

__MAIN__()
{

  // В PSW80 будет записано значение 5
  PSW[80] = sizeof(ts);
  // В PSW81 будет записано значение 8
  PSW[81] = sizeof(ts2);
}
```

Структуры могут иметь произвольную вложенность и свободно сочетаются с массивами – например, можно объявить массив структур, одним из полей которой является структура, одним из полей которой является массив и т. д.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

struct TestStuct
{
  uint8_t  var1;
  uint32_t var2;
};

__MAIN__()
{
  struct TestStuct ArrTs[4];
  ArrTs[0].var1 = 42;
}
```

8. Объединения

Объединения очень похожи на структуры, за одним существенным отличием – все их объекты размещаются в одних и тех же ячейках памяти. Это обеспечивает разные варианты представления одних и тех же данных.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

union IPAddress
{
  uint32_t HexIp;
  uint8_t octets[4];
};

__MAIN__()
{
  union IPAddress ip;
  // Соответствует 192.168.0.10
  ip.HexIp = 0xC0A8000A;

  // Старший октет фактически хранится в младшем байте

  // В регистр PSW90 будет записано значение 192
  PSW[90] = ip.octets[3];
  // В регистр PSW91 будет записано значение 168
  PSW[91] = ip.octets[2];
  // В регистр PSW92 будет записано значение 0
  PSW[92] = ip.octets[1];
  // В регистр PSW93 будет записано значение 10
  PSW[93] = ip.octets[0];
}
```

9. Перечисления

Перечисление позволяет указать для конкретных значений целочисленной переменной символьные названия, что повышает читабельность кода. В языке Си все перечисления трактуются как значения типа **int** и имеют соответствующий диапазон доступных значений. Часто перечисления используются в качестве меток оператора **switch**.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__
enum Days
{
  NONE, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY
};

__MAIN__()
{
 enum Days WeekDay;
 // Чтение системного регистра с номером дня недели
 WeekDay = SPSW[22];

 switch (WeekDay)
 {
   case MONDAY:
   case TUESDAY:
   case WEDNESDAY:
   case THURSDAY:
   case FRIDAY:
      // разместите здесь код управления системой в будние дни
      break;
   case SATURDAY:
   case SUNDAY:
     // разместите здесь код управления системой в выходные дни
     break;
   default:
     // разместите здесь код вывода сообщения об ошибке
  }
}
```

Значения элементов перечислений могут быть указаны в явном виде:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

enum Days
{
  NONE = 0,
  MONDAY = 1,
  TUESDAY = 2,
  WEDNESDAY = 3,
  THURSDAY = 4,
  FRIDAY = 5,
  SATURDAY = 6,
  SUNDAY = 7
};
```

При этом для некоторых элементов значения могут быть опущены:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

enum Colors
{
   RED = 1,
   GREEN,    // 2
   BLUE,     // 3
   WHITE = 7,
   BLACK     // 8
};
```

Значения элементов перечисления могут быть «значимыми»:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

// Используется цветовая модель BGR, применяемая в функциях работы с элементом Холст
enum Colors
{
   RED = 0x0000FF,
   GREEN = 0x00FF00,
   BLUE = 0xFF0000
};
```

10. Псевдонимы (typedef)

Ключевое слово **typedef** создает псевдоним для типа данных, что повышает читабельность кода.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

typedef struct
{
  uint16_t x;
  uint16_t y;
} Point;



__MAIN__()
{
 // Использование typedef позволило обойтись без ключевого слова struct
 Point Line[2] = {10, 10, 100, 100};
}
```

11. Тип void

Void – это специфический тип данных. Он является неполным – создать переменную данного типа нельзя. Но **void** можно использовать:

- в качестве типа, возвращаемого функцией – это будет означать, что данная функция не возвращает никакого значения (см., например, Beep);
- при создании функций, которые не принимают никаких параметров (см., например, Beep);
- при указании параметров функции, которые могут иметь произвольный тип (см., например, функцию WriteF);
- указатель на **void** соответствует указателю на произвольный тип данных. Это может использоваться при создании функций, которые должны работать с данными произвольного типа (см., например, функции семейства Read/Write).

12.5.4 Переменные

Переменная – это именованный участок памяти, в котором хранится значение. Для использования переменной её необходимо определить в файле исходного кода, указав тип. Определение может быть проведено в любом месте файла, что влияет на область видимости и существования переменной.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__
#include "stdint.h"

// Глобальная переменная
// Сохраняет свое значение между вызовами программы
// Может использоваться в пределах файла исходного кода
uint16_t global_var1;

// Пример одновременного определения нескольких переменных общего типа
uint16_t global_var2, global_var3;

__MAIN__()
{
  // Локальная переменная
  // Не сохраняет свое значение между вызовами программы
  // (переинициализурется при каждом вызове)
  // Может использоваться только в пределах блока
  // (в данном случае – в пределах функции __MAIN__)
  uint16_t local_function_var;

  // какой-то код
  {
    // Локальная переменная
     // Не сохраняет свое значение между вызовами программы
     // (переинициализурется при каждом вызове)
     // Может использоваться только в пределах блока
     // (в данном случае – до символа закрывающей фигурной скобки)
     uint16_t local_scope_var;

     // какой-то код
  }

  // использовать здесь local_scope_var не получится
}
```

Определение переменной должно быть сделано перед её использованием (т. е. в расположенных выше строках файла).

Переменные могут быть глобальными и локальными.

Глобальные переменные определяются вне функций. Они доступны для использования в рамках всего файла исходного кода. Кроме того, их можно использовать в других файлах (см. пункт 12.5.5). Если для глобальной переменной не указано значение инициализации – то она инициализируется значением **0**. Значение глобальной переменной **сохраняется** между вызовами программы.

Локальные переменные объявляются внутри функций. Они доступны для использования только в рамках блочного оператора, в котором определены (блочный оператор представлен парой фигурных скобок). Если для локальной переменной не указано значение инициализации – то при старте программы она будет содержать произвольное значение («мусор»). Значение локальной переменной **не сохраняется** между вызовами программы. Частным случаем локальных переменных являются параметры функции.

Инициализация – это процесс однократной записи в переменную предустановленного значения при запуске программы.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdbool.h"
#include "stdint.h"
__MAIN__()
{
   bool BoolVar = true;
   uint16_t u16_var = 123;
   char str[] = "test";
}
```

Способы инициализации для других типов данных приведены в примерах пункт 12.5.3.

12.5.5 Спецификаторы и квалификаторы

При объявлении и определении переменных и функций могут использоваться спецификаторы. Основными из них являются **static** и **extern**. Спецификатор размещается перед типом переменной или перед типом значения, возвращаемого функций.

Локальные переменные со спецификатором **static** сохраняют свои значения между вызовами программы. Кроме того, если для них не указаны значения инициализации – то они инициализируются значением **0**.

Для глобальных переменных и функций спецификатор **static** ограничивает область видимости файлом исходного кода, в котором они определены.

Пример

```
// Func0.c

#include "macro.h"
#define __SYS_DEFINE__
#include "stdint.h"

// Эта глобальная переменная доступна в других файлах
uint16_t global_var;
// Эта глобальная переменная недоступна в других файлах
static uint16_t static_global_var;

__MAIN__()
{
  // Эта локальная переменная не сохранит свое значение между вызовами программы...
  // ...и будет инициализироваться начальным значением при каждом вызове
  uint16_t local_var = 0;
  // Эта локальная переменная сохранит свое значение между вызовами программы
  static uint16_t static_local_var = 0;
}

// Func1.c

#include "macro.h"
#define __SYS_DEFINE__
#include "stdint.h"

// Импорт глобальной переменной из файла Func0.c
// Импортировать static_global_var не получится...
// ...так как она помечена модификатором static
extern uint16_t global_var;

__MAIN__()
{
  global_var = 42;
}
```

Как показано в примере выше, спецификатор **extern** используется для объявления (импорта) переменных, которые определены в другом файле.

Переменные с квалификатором **const** расцениваются как константы, значения которых не должны изменяться в процессе работы программы. Попытки каким-либо образом изменить эти значения (например, с помощью использования указателей) приводятся к неопределённому поведению (undefined behavior) программы.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

__MAIN__()
{
  // Переменная (может быть массивом, структурой, полем структуры и т. д.)
  float val = 11.22;
  // Константное значение, не может быть изменено
  const float readonly_val = 22.33;
  // Указатель на константу. Значение адреса может быть изменено...
  // ...а вот расположенные по адресу данные - нет
  const float *ptr1 = &val
  // Эквивалент варианта выше
  float const *ptr2 = &val
  // Константный указатель. Значение адреса не может быть изменено...
  // ...а вот расположенные по адресу данные - могут
  float *const ptr3 = &val
  // Константый указатель на константу. Ни адрес, ни расположенные по нему данные...
  // ...не могут быть изменены
  const float *const ptr4 = &val
  // Эквивалент варианта выше
  float const *const ptr5 = &val
}
```

12.5.6 Литералы

Литерал – это элемент исходного кода, который соответствует фиксированному значению. Например, в строке

Пример
```
float val = 11.22;
```

значение **11.22** является литералом.

Для литералов определённых категорий доступны префиксы и постфиксы, влияющие на их формат и тип.

Префиксы для целочисленных типов:

- **0x** (шестнадцатеричная система счисления; HEX);
- **0b** (двоичная система счисления).

Префиксы для строк:

- префикс **L** указывает на кодировку **UTF-16**.

Постфиксы для целочисленных типов:

- **u (U)** – признак «беззнаковости» литерала. Может сочетаться с другими постфиксами (например, **ul** – unsigned long);
- **l (L)** – литерал является значением типа **long**;
- **ll (LL)** – литерал является значением типа **long long**.

По умолчанию (без постфикса) целочисленный литерал интерпретируется компилятором как значение типа **int**.

Постфиксы для типов с плавающей точкой:

- **f (f)** – литерал является значением типа **float**;
- **L (L)** – литерал является значением типа **long double**.

По умолчанию (без постфикса) литерал с плавающей точкой интерпретируется компилятором как значение типа double.

Постфиксы позволяют:

- добиться нужной интерпретации литерала, что в некоторых случаях может быть очень важно (см. пример с делением ниже);
- сэкономить память;
- повысить читабельность кода.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"
#include "wchar.h"

__MAIN__()
{
  // DEC
  uint16_t val1 = 16;
  // HEX
  uint16_t val2 = 0x10;
  // Двоичная систем, разделитель ' является опциональным
  uint16_t val3 = 0b0001'0000;

  // Результат: 3 (так как выполняется целочисленное деление)
  float val4 = 10 / 3;
  // Результат: 3.333333
  float val5 = 10 / 3f; // или 3.0

  wchar_t UnicodeStr[] = L"тест";

}
```

12.5.7 Идентификаторы

Идентификатор – это имя, указываемое при объявлении/определении переменных, функций, псевдонимов типов данных, макросов и других сущностей.

В языке Си идентификаторы являются регистрозависимыми, т. е. **var** и **Var** – это два разных идентификатора.

На идентификатор накладываются следующие ограничения:

- может включать в себя буквы английского языка, цифры и символы нижнего подчеркивания (_);
- первый символ не может быть цифрой;
- не может совпадать с ключевым словом языка Си или идентификатором, используемым в стандартной библиотеке языка Си;
- длина идентификатора не может превышать 256 символов.

12.5.8 Комментарии

В языке Си поддержаны однострочные и многострочные комментарии:

Пример

```
// однострочный комментарий


/* многострочный
комментарий */
```

Кнопки редактора макросов позволяют закомментировать/раскомментировать выделенные строки с помощью однострочных комментариев.

12.5.9 Операторы и выражения

Выражение — это упорядоченный набор операций над переменными, литералами и функциями.

Таблица 12.6 – Таблица основных операторов языка Си

Оператор	Описание
Оператор присваивания	
=	
Арифметические операторы	
+ - * / %	Сложение Вычитание Умножение Деление Остаток от деления *Применимы к арифметическим типам (оператор % применим только к целочисленным типам)*
Логические операторы	
&& || !	Логическое И (AND) Логическое ИЛИ (OR) Логическое отрицание (NOT) *Применимы к логическому типу. Заголовочный файл iso646.h содержит макроопределения-псевдонимы (например, and для &&)*
Побитовые операторы	
& | ^ ~	Побитовое И Побитовое ИЛИ Побитовое исключающее ИЛИ (XOR) Побитовая инверсия *Применимы к целочисленным типам*
Операторы сравнения	
< > <= >= == !=	«Меньше» «Больше» «Меньше или равно» «Больше или равно» *Применимы к арифметическими логическим типам* Проверка на равенство Проверка на неравенство *Применимы к арифметическим (включая перечисления) и логическим типам*
Операторы битового сдвига	

Продолжение таблицы 12.6

Оператор	Описание
<< >>	Битовый сдвиг влево Битовый сдвиг вправо *Применимы к арифметическим типам. Первый аргумент оператора – это значение, в котором выполняется сдвиг, второй – количество сдвигаемых бит*
Остальные операторы	
[] . ->& * , sizeof	Доступ к объекту массива или указателя Доступ к полю структуры Доступ к полю структуры через указатель на структуру Получение адреса объекта Разыменование указателя Разделитель аргументов функции Получение адреса объекта или типа *(возвращаемое значение имеет тип size_t, который определен в заголовочном файле stddef.h)*

Для преобразования типов используется специфический оператор:

Пример

```
int ivar;
float fvar = 11.22;


// float to int
// ivar получит значение 11
ivar = (int)fvar;
```

В выражении могут одновременно использоваться несколько операторов:

Пример

```
uint16_t var;
// Что выполнится сначала – сложение 5 и 1, или умножение 10 на 5?
var = 10 * 5 + 1;
```

Порядок их вычисления зависит от приоритета. Для управления приоритетом используются скобки – они имеют высший приоритет:

Пример

```
uint16_t var;
// Теперь мы точно знаем, что сначала выполнится сложение
var = 10 * (5 + 1);
```

В языке Си поддерживается множественное присваивание:

Пример

```
uint16_t var1, var2, var3;
var1 = var2 = var3 = 42;
```

Арифметические, логические и «сдвиговые» операторы могут комбинироваться с оператором присваивания:

Пример
var1 += 5; // эквивалентно var = var + 5;

Характерными для языка Си являются операторы инкремента и декремента, которые могут быть префиксными и постфиксными:

Пример

var2 = ++var1; // увеличивает var1 на +1, после чего записывает результат в var2

var3 = var2++; // записывает var2 в var3, после чего увеличивает var2 на +1

Некоторые характерные для других языков операторы в Си представлены в виде функций заголовочного файла **math.h**:

- abs (получение абсолютной величины числа);
- pow (возведение в степень);
- sin, cos;
- и др.

12.5.10 Функции

При создании простых макросов можно расположить весь их код в функции **__MAIN__**. Но при решении более сложных задач некоторые действия будут повторяться многократно, и разумно реализовать их в виде отдельных функций.

Ниже приведен пример определения и вызова функции, которая имеет 2 параметра:

- **T** типа **float**;
- **IsAirSaturation** типа **bool**.

Эти параметры принимают значения аргументов функции, переданные ей при вызове.

В результате выполнения функция возвращает значение типа **float**.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__
#include "math.h"
#include "stdbool.h"

#define a0 999.83952
#define a1 16.952577
#define a2 -7.9905127E-3
#define a3 -4.6241757E-5
#define a4 1.0584601E-7
#define a5 -2.8103006E-10
#define b  0.0168872

// Уравнение Келла для определения плотности воды при температуре T
// Параметр IsAirSaturation определяет необходимость учета влияния растворенного...
// ...в воде воздуха
// http://www.bioconsult.ch/Inovatech/W-Lehre/J%20Che%20Eng%20Dat20,%2097.pdf
// Размерность: г/мл
// Точность: 4 знака после запятой
float WaterDensity(float T, bool IsAirSaturation)
{
  float wd;
  wd = (a0 + a1*T + a2*pow(T,2) + a3*pow(T,3) + a4*pow(T,4) + a5*pow(T,5) ) /
    (1.0 + b*T);
  // http://www.bioconsult.ch/Inovatech/W-Lehre/J%20Che%20Eng%20Dat20,%2097.pdf
  // Учет влияния растворенного воздуха
  if (IsAirSaturation)
  {
    wd = wd - 0.004612 + 0.000106 * T
  }

  // Перевод из кг/м3 в г/мл
  wd = 0.001 * wd;

  return wd;
}
```

```
__MAIN__()
{
  *(float*)(PSW + 20) = WaterDensity(*(float*)(PSW + 10), false);
  *(float*)(PSW + 22) = WaterDensity(*(float*)(PSW + 12), false);
  // и т. д.
  // чтобы избежать копипасты – можно разместить вызов функции в цикле for
}
```

Оператор **return** традиционно размещается в конце функции, но может располагаться в любых ее фрагментах (в т. ч. таких операторов в функции может быть несколько, чтобы организовать выход из функции при различных условиях).

В некоторых ситуациях в качестве аргументов функции используются указатели – см., в частности, примеры использования функций **Reads/Writes**.

Если требуется указать, что функция не имеет явного возвращаемого значения (например, она как раз манипулирует данными, полученными по указателю), то в качестве типа функции следует использовать **void**.

Если функция не имеет параметров, то вместо них указывается **void**.

Характерный пример – функция **Beep**, которая используется для активации зуммера панели. Она не принимает и не возвращает никаких значений, и объявлена следующим образом:

Пример
```
void Beep(void);
```

12.5.11 Препроцессор

Препроцессор – это программа, подготавливающая код на языке Си к компиляции. К основным задачам препроцессора относятся:

- вставка в обрабатываемый файл содержимого других заголовочных файлов с помощью директивы **#include**;
- выполнение макроподстановок для директив **#define**;
- выполнение директив условной компиляции (не рассматриваются в рамках данного документа);
- выполнение других директив.

Директива **#include** используется для подключения заголовочных файлов – в частности, заголовочных файлов стандартной библиотеки языка Си. Это требуется для возможности вызова функций этой библиотеки. Например, для использования функций заголовочного файла **math.h** требуется добавить такую строку:

Пример
```
#include "math.h"
```

или такую:

Пример
```
#include <math.h>
```

Разница между этими двумя формами заключается только в принципе поиска файлов, что не имеет значения при работе с панелями оператора.

Директива **#define** используется для определения констант и макросов. Препроцессор заменяет первый аргумент директивы вторым, выполняя макроподстановку.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Макроопределение
#define ARR_SIZE 4

__MAIN__()
{
  uint16_t Data[ARR_SIZE];

  // Размещаем в массиве значения регистров панели PSW100, PSW200, PSW300 и PSW400...
  //  ...для какой-то последующей обработки
  for (int i = 0; i < ARR_SIZE; i++)
  {
    Data[i] = PSW[100 * (i + 1)];
  }
}
```

См. также список предопределённых констант #define, автоматически создаваемых препроцессором.

В качестве констант также можно использовать переменные с квалификатором **const** – по сравнению с макроконстантами этот подход уменьшает вероятность ошибки, так как для таких переменных компилятор сможет проверить соответствие типов.

Пример макроса с параметрами:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Ограничиваем числовое значение в диапазоне [min, max]
#define LIMIT(min, x, max) (((x) < (min)) ? (min) : (((x) > (max)) ? (max) : (x)))

__MAIN__()
{
  uint16_t var1 = 20;
  uint16_t var2 = 40;
  uint16_t var3 = 60;

  // В регистр PSW100 будет записано значение 30
  PSW[100] = LIMIT(30, var1, 50);
  // В регистр PSW101 будет записано значение 40
  PSW[101] = LIMIT(30, var2, 50);
  // В регистр PSW102 будет записано значение 50
  PSW[102] = LIMIT(30, var3, 50);
}
```

Такой макрос можно заменить функцией. Но подобные макросы полезны тем, что могут быть добавлены на вкладку Встроенные функции редактора макросов.

Если требуется, чтобы макрос выполнял несколько отдельных выражений, то используется идиома **do {…} while(0)**:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "stdint.h"

// Макрос меняет между собой значения аргументов типа int
// Символ «\» используется для переноса строк внутри кода макроса
#define SWAP(a, b) \
    do { \
        int temp = (a); \
        (a) = (b); \
        (b) = temp; \
    } while(0)

__MAIN__()
{
  int x = 10, y = 20;
  SWAP(x, y);
  // Теперь x = 20, y = 10
}
```

В макросах с параметрами могут использоваться специфические операторы **#** (обрамление параметра макроса двойными кавычками) и **##** (объединение двух лексем). См. примеры использования этих операторов в статье на Википедии.

ПРИМЕЧАНИЕ

Макросы с параметрами требуют тщательной расстановки скобок, поэтому использовать их следует с осторожностью.

Директивы **#warning** и **#error** позволяют вывести в лог компилятора предупреждение или сообщение об ошибке. Это, например, позволяет расставлять в исходном коде различного рода напоминания.

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  #warning Добавить проверку деления на 0
  PSW[106] = PSW[105] / PSW[104];


}
```

Директива **#pragma** позволяет выполнить указанное после имени директивы действие. Список поддерживаемых действий зависит от компилятора. Например, в пункте 12.5.4 использовалась **#pragma pack** для управления выравниванием в структуре.

12.5.12 Стандартная библиотека языка Си

Стандартная библиотека языка Си позволяет использовать в программе широкий набор функций, определённых в стандарте языка. Для использования конкретной функции требуется с помощью директивы **#include** подключить к программе заголовочный файл, в котором объявлена данная функция.

Ниже приведен список заголовочных файлов, которые могут потребоваться при создании программ для панелей оператора.

Таблица 12.7 – Список некоторых заголовочных файлов стандартной библиотеки языка Си

Заголовочный файл	Что содержит
ctype.h	Функции для обработки символов кодировки ASCII
errno.h	Коды ошибок, возвращаемых функциями стандартной библиотеки, и глобальную переменную **errno**, в которую они записываются
inttypes.h	Функции для работы с целочисленными переменными фиксированного размера
iso646.h	Макроопределения-псевдонимы для логических операторов (например, **and** для **&&**)
math.h	Арифметические и тригонометрические функции
stdbool.h	Макросы **true** и **false**
stdint.h	Целочисленные типы данных фиксированного размера (**uint16_t** и т. д.)
stdio.h	Функции для работы с файлами
stdlib.h	Функции общего назначения (выделение и освобождение памяти, генерация псевдослучайных чисел, сортировка и поиск, преобразование чисел в строки и строк в числа и т. д.)
string.h	Функции для работы с ASCII-строками и памятью (memcpy, memmove, memcmp, memset)
time.h	Функции конвертации различных форматов даты и времени
wchar.h	Функции для работы со строками в кодировке UTF-16 (UCS-2)
wctype.h	Функции для обработки символов кодировки UTF-16 (UCS-2)

12.6 Встроенные функции и макросы

В редакторе кода на вкладке **Встроенные функции** расположен список встроенных функций и макросов.

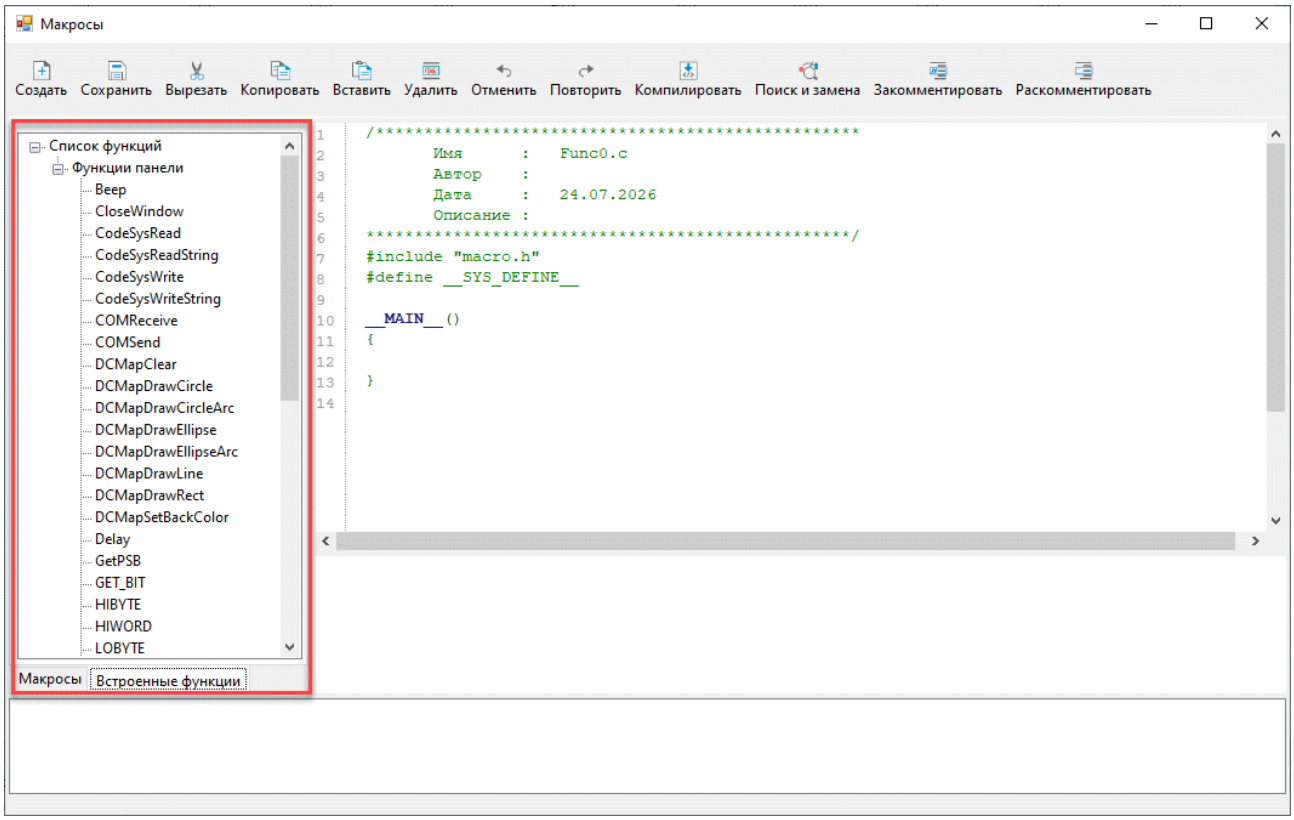


Рисунок 12.5 – Встроенные функции

Добавление пользовательских макросов на эту вкладку рассмотрено в пункте 12.6.9.

12.6.1 Работа с внутренними битами и регистрами панели

Регистры области памяти **PSW** и **SPSW** представлены как массивы данных:

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Запись значения в регистр PSW0
  PSW[0] = 123;

  // Чтение значения из регистра PSW1 в переменную
  WORD wVar;
  wVar = PSW[1];

  // Для работы с различными типами данных нужно использовать операторы конверсии:
  float rValue1;
  rValue1 = *(float*)(PSW + 10);

  float rValue2 = 11.22;
  *(float*)(PSW + 12) = rValue2;

  DWORD dwValue1;
  dwValue1 = *(DWORD*)(PSW + 14);

  DWORD dwValue2 = 123456;
  *(DWORD*)(PSW + 16) = dwValue2;

}
```

Для работы с другими областями памяти используются специальные функции. Некоторые из этих функций могут использоваться и для работы с **PSW** и **SPSW**:

Таблица 12.8 – Функции для работы с областями памяти панели

Функция	Области памяти, к которым она применима
GetPSB SetPSB	PSB
GetSPSB SetSPSB	SPSB
Read Reads Write Writes	PSB, SBSB, PSW, SPSW, PFW, SPFW
WriteF	PSW, SPSW, PFW, SPFW
ReadPFW WritePFW	PFW
ReadSPFW WriteSPFW	SPFW

12.6.2 Функции для работы с памятью панели и slave-устройств

В функциях **Read/Reads/Write/WriteF/Writes** в случае использования протокола Modbus для параметра **DataArea** доступны следующие значения:

Таблица 12.9 – Значения параметра DataArea для slave-устройств с протоколом Modbus

Modbus RTU	Modbus ASCII	Modbus TCP	Описание
TP7_0x	TP8_0x	TP9_0x	Область **coils** (биты, доступные для чтения и записи)
TP7_1x	TP8_1x	TP9_1x	Область **discrete inputs** (биты, доступные только для чтения)
TP7_3x	TP8_3x	TP9_3x	Область **input registers** (регистры, доступные только для чтения)
TP7_4x	TP8_4x	TP9_4x	Область **holding registers** (регистры, доступные для чтения и записи)

Эти наборы эквивалентны, т. е. можно, например, использовать **TP7_4x** для протоколов Modbus ASCII и Modbus TCP.

В случае использования других протоколов введите в редакторе кода имя функции (например, **Reads**) и поставьте открывающую скобку для вызова окна настройки аргументов функции. Выберите в нём нужное устройство и его область памяти, чтобы автоматически сформировать значение аргумента.

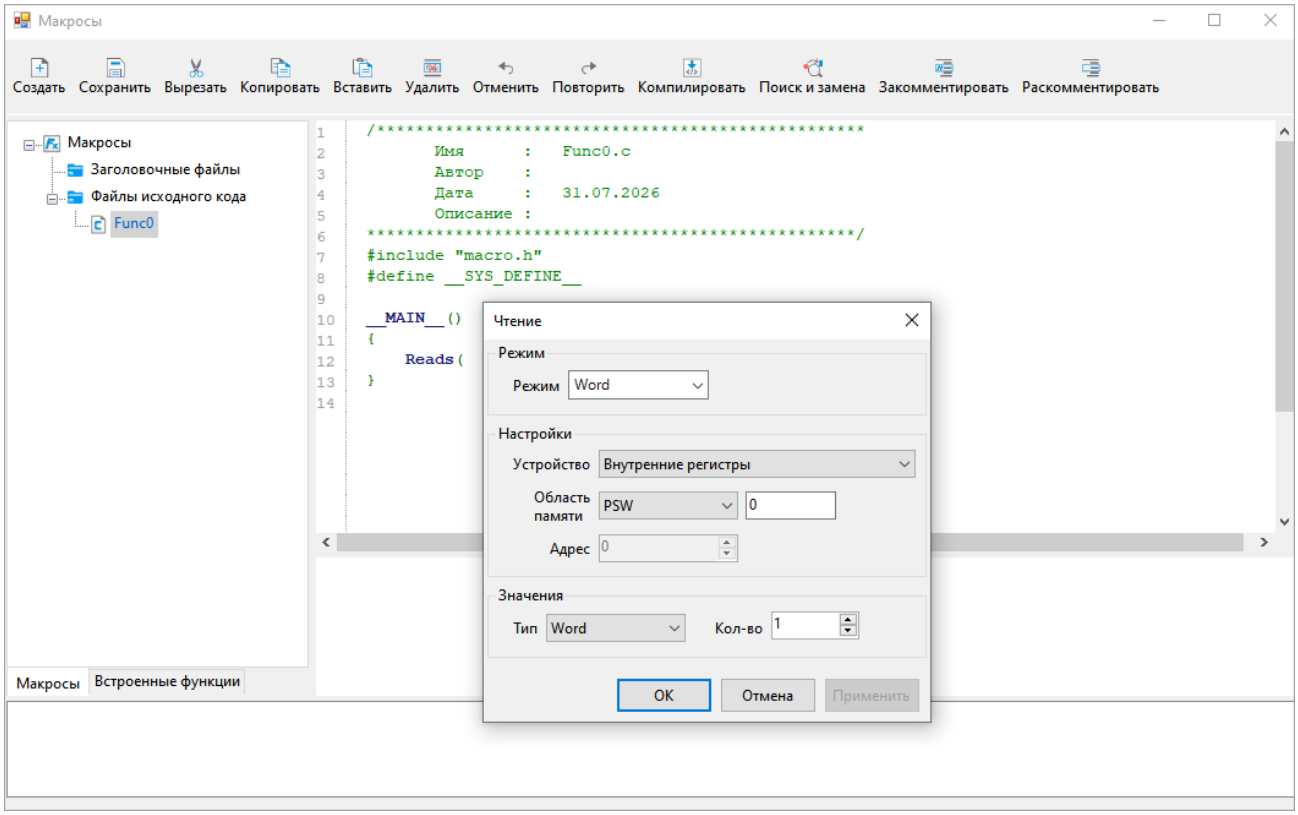


Рисунок 12.6 – Окно настройки аргументов функции

В функциях **Read/Reads/Write/WriteF/Writes** в случае доступа к внутренней памяти панели для параметра **DataArea** доступны следующие значения:

- TP_PSB;
- TP_SPSB;
- TP_PSW;
- TP_PFW;
- TP_SPSW;
- TP_SPFW.

В функциях **Read/Reads/Write/WriteF/Writes** для параметра **DataType** доступны следующие значения:

- DT_Bit;
- DT_Byte (доступен только для некоторых протоколов обмена);
- DT_WORD;
- DT_DWORD;
- DT_DDWORD;
- DT_Word_String;
- DT_DWord_String;
- DT_DDWord_String.

1. Функции GetPSB и GetSPSB

Функции **GetPSB** и **GetSPSB** используются для получения значения бита области **PSB** (GetPSB) или **SPSB** (GetSPSB).

Таблица 12.10 – Описание функции GetPSB (GetSPSB)

Определение	BOOL GetPSB(DWORD Addr);
Addr	Адрес бита
Возвращаемое значение	Значение бита – **1** (true) или **0** (false)
Примечание	Для использования макросов **true** и **false** следует подключить заголовочный файл **stdbool.h**

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Приведенный ниже фрагмент кода выполнится только в том случае, если PSB10 имеет...
  // ...значение true
  if (GetPSB(10) )
  {
    // ваш код
  }

}
```

2. Функции SetPSB и SetSPSB

Функции **SetPSB** и **SetSPSB** используются для установки значения бита области **PSB** (SetPSB) или **SPSB** (SetSPSB).

Таблица 12.11 – Описание функции SetPSB (SetSPSB)

Определение	void SetPSB(DWORD Addr, BOOL Value);
Addr	Адрес бита
Value	Устанавливаемое значение бита – **1** (true) или **0** (false)
Возвращаемое значение	отсутствует
Примечание	Для использования макросов **true** и **false** следует подключить заголовочный файл **stdbool.h**

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Установка в PSB10 значения true
  SetPSB(10, 1);
}
```

3. Функция Read

Функция **Read** используется для чтения одного параметра из памяти панели или slave-устройства в переменную макроса. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **Read** и поставьте открывающую скобку.

Таблица 12.12 – Описание функции Read

Определение	BOOL Read (int Device, int SlaveId, int DataArea, int DataType, int RegAddr, int BitAddr, void* pValue);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
SlaveId	Адрес устройства (в случае чтения из памяти панели следует использовать адрес **0**)
DataArea	Область памяти
DataType	Тип данных
RegAddr	Адрес регистра
BitAddr	Номер бита (для чтения бита регистра, в остальных случаях равен 0)
pValue	Указатель на буфер, в котором будет размещено считанное значение
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Чтение holding-регистра 20 из Modbus RTU Slave-устройства с адресом 5
  WORD wValue1;
  Read(_T("Modbus RTU Master"), 5, TP7_4x, DT_WORD, 20, 0, &wValue1);

  // Чтение внутреннего регистра PFW300
  WORD wValue2;
  Read(_T("Внутренние регистры"), 0, TP_PFW, DT_Word, 300, 0, &wValue2);

  // Чтение бита PSB100
  bool xValue1;
  Read(_T("Внутренние регистры"), 0, TP_PSB, DT_Bit, 100, 0, &xValue1);

  // Чтение бита PSW200.8
  bool xValue2;
  Read(_T("Внутренние регистры"), 0, TP_PSW, DT_Bit, 200, 8, &xValue2);

}
```

4. Функция Write

Функция **Write** используется для записи одного значения макроса в память панели или slave-устройство. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **Write** и поставьте открывающую скобку.

Таблица 12.13 – Описание функции Write

Определение	BOOL Write(int Device, int SlaveId, int DataArea, int DataType, int RegAddr, int BitAddr, void* pValue);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
SlaveId	Адрес устройства (в случае записи в память панели следует использовать адрес **0**)
DataArea	Область памяти
DataType	Тип данных
RegAddr	Адрес регистра
BitAddr	Номер бита (для записи бита регистра, в остальных случаях равен 0)
pValue	Указатель на буфер, в котором размещается записываемое значение
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Запись holding-регистра 20 в Modbus RTU Slave-устройстве с адресом 5
  WORD wValue1 = 123;
  Write(_T("Modbus RTU Master"), 5, TP7_4x, DT_WORD, 20, 0, wValue1);

  // Запись во внутренний регистр PFW300
  WORD wValue2 = 123;
  Write(_T("Внутренние регистры"), 0, TP_PFW, DT_Word, 300, 0, wValue2);

  // Запись бита PSB100
  bool xValue1 = 1;
  Write(_T("Внутренние регистры"), 0, TP_PSB, DT_Bit, 100, 0, xValue1);

  // Запись бита PSW200.8
  bool xValue2 = 1;
  Write(_T("Внутренние регистры"), 0, TP_PSW, DT_Bit, 200, 8, xValue2);

}
```

5. Функция Reads

Функция **Reads** используется для чтения одного или нескольких параметров из памяти панели или slave-устройства в переменные макроса. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **Reads** и поставьте открывающую скобку.

Таблица 12.14 – Описание функции Reads

Определение	BOOL Reads(int Device, int SlaveId, int DataArea, int DataType, int RegAddr, int BitAddr, int Count, void* pValue);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
SlaveId	Адрес устройства (в случае чтения из памяти панели следует использовать адрес **0**)
DataArea	Область памяти
DataType	Тип данных
RegAddr	Адрес начального регистра
BitAddr	Номер бита (для чтения битов регистров, в остальных случаях равен 0)
Count	Количество объектов
pValue	Указатель на буфер, в котором будут размещены считанные значения
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Чтение holding-регистров 20...29 из Modbus RTU Slave-устройства с адресом 5
  WORD awData1[10];
  Reads(_T("Modbus RTU Master"), 5, TP7_4x, DT_WORD, 20, 0, 10, awData1);

  // Чтение внутренних регистров PFW300...329
  WORD awData2[30];
  Reads(_T("Внутренние регистры"), 0, TP_PFW, DT_Word, 300, 0, 30, awData2);

  // Чтение битов PSB100...PSB115
  WORD wBitMask;
  Reads(_T("Внутренние регистры"), 0, TP_PSB, DT_Bit, 100, 0, 16, &wBitMask);

  // Чтение битов PSW200.8...PSW200.15
  BYTE byBitMask;
  Reads(_T("Внутренние регистры"), 0, TP_PSW, DT_Bit, 200, 8, 8, &byBitMask);

}
```

6. Функция Writes

Функция **Writes** используется для записи одного или нескольких значений макроса в память панели или slave-устройство. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **Writes** и поставьте открывающую скобку.

Таблица 12.15 – Описание функции Writes

Определение	BOOL Writes(int Device, int SlaveId, int DataArea, int DataType, int RegAddr, int BitAddr, int Count, void* pValue);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
SlaveId	Адрес устройства (в случае чтения из памяти панели следует использовать адрес 0)
DataArea	Область памяти
DataType	Тип данных
RegAddr	Адрес начального регистра
BitAddr	Номер бита (для записи битов регистров, в остальных случаях равен 0)
Count	Количество объектов
pValue	Указатель на буфер, в котором размещаются записываемые значения
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Запись holding-регистров 20...29 в Modbus RTU Slave-устройстве с адресом 5
  WORD awData1[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
  Writes(_T("Modbus RTU Master"), 5, TP7_4x, DT_WORD, 20, 0, 10, awData1);

  // Запись во внутренние регистры PFW300...329
  // PWF300 получит значение 1, PFW310 получит значение 2, PFW320 получит значение 3,
  // PFW329 получит значение 4
  // Остальные регистры получат значение 0
  WORD awData2[30] = { [0] = 1, [10] = 2, [20] = 3, [29] = 4 };
  Writes(_T("Внутренние регистры"), 0, TP_PFW, DT_Word, 300, 0, 30, awData2);

  // Запись битов PSB100...PSB115
  WORD wBitMask = 39321;
  Writes(_T("Внутренние регистры"), 0, TP_PSB, DT_Bit, 100, 0, 16, &wBitMask);

}
```

7. Функция WriteF

Функция **WriteF** используется для записи одного значения с плавающей точкой в память панели или slave-устройство. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **WriteF** и поставьте открывающую скобку.

Таблица 12.16 – Описание функции WriteF

Определение	BOOL WriteF(int Device, int SlaveId, int DataArea, int DataType, int RegAddr, int BitAddr, void Value);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
SlaveId	Адрес устройства (в случае чтения из памяти панели следует использовать адрес **0**)
DataArea	Область памяти
DataType	Тип данных
RegAddr	Адрес начального регистра
BitAddr	Номер бита (не используется, равно 0)
Count	Количество объектов
Value	Записываемое значение типа **Float** или **Double** (в зависимости от **DataType**)
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  // Запись float-значения в holding-регистры 20-21 в Modbus RTU Slave-устройстве с
  // адресом 5
  float rValue = 11.22;
  WriteF(_T("Modbus RTU Master"), 5, TP7_4x, DT_DWord, 300, 0, rValue);

  // Запись float-значения во внутренние регистры PFW300-PFW301
  float rValue = 11.22;
  WriteF(_T("Внутренние регистры"), 0, TP_PFW, DT_DWord, 300, 0, rValue);
}
```

8. Функции ReadPFW и ReadSPFW

Функции **ReadPFW** и **ReadSPFW** используются для чтения одного или нескольких регистров области **PFW** (ReadPFW) или **SPFW** (ReadSPFW) в переменные макроса.

Таблица 12.17 – Описание функции ReadPFW (ReadSPFW)

Определение	void ReadPFW(int RegAddr, int ByteCount, void* pData);
RegAddr	Адрес регистра
ByteCount	Количество байт (1 регистр = 2 байта)
pData	Указатель на буфер, в котором будут размещены считанные значения
Возвращаемое значение	Отсутствует

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{

  // Чтение внутренних регистров PFW300...305
  WORD awValue[6];
  ReadPFW(300, 6 * sizeof(WORD), awValue);

}
```

9. Функции WritePFW и WriteSPFW

Функции **WritePFW** и **WriteSPFW** используются для записи значений макроса в один или несколько регистров области **PFW** (WritePFW) или **SPFW** (WriteSPFW).

Таблица 12.18 – Описание функции WritePFW (WriteSPFW)

Определение	void WritePFW(int RegAddr, int ByteCount, void* pData);
RegAddr	Адрес регистра
ByteCount	Количество байт (1 регистр = 2 байта)
pData	Указатель на буфер, в котором размещаются записываемые значения
Возвращаемое значение	Отсутствует

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  // Запись внутренних регистров PFW300...305
   WORD awValue[6] = {1, 2, 3, 4, 5, 6};
  WritePFW(300, 6 * sizeof(WORD), awValue);
}
```

12.6.3 Функции для обмена данными с ПЛК CODESYS V3.5

Данные функции используются для работы с переменными вкладки **С устройства,** импортированными из ПЛК с системой исполнения CODESYS V3.5 и опрашиваемыми по протоколу CODESYS V3 (PLC Handler) или OPC UA Client.;

1. Функция CodeSysRead

Функция **CodeSysRead** используется для чтения значения переменной из вкладки **Библиотека переменных – С устройства** в переменную макроса. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **CodeSysRead** и поставьте открывающую скобку.

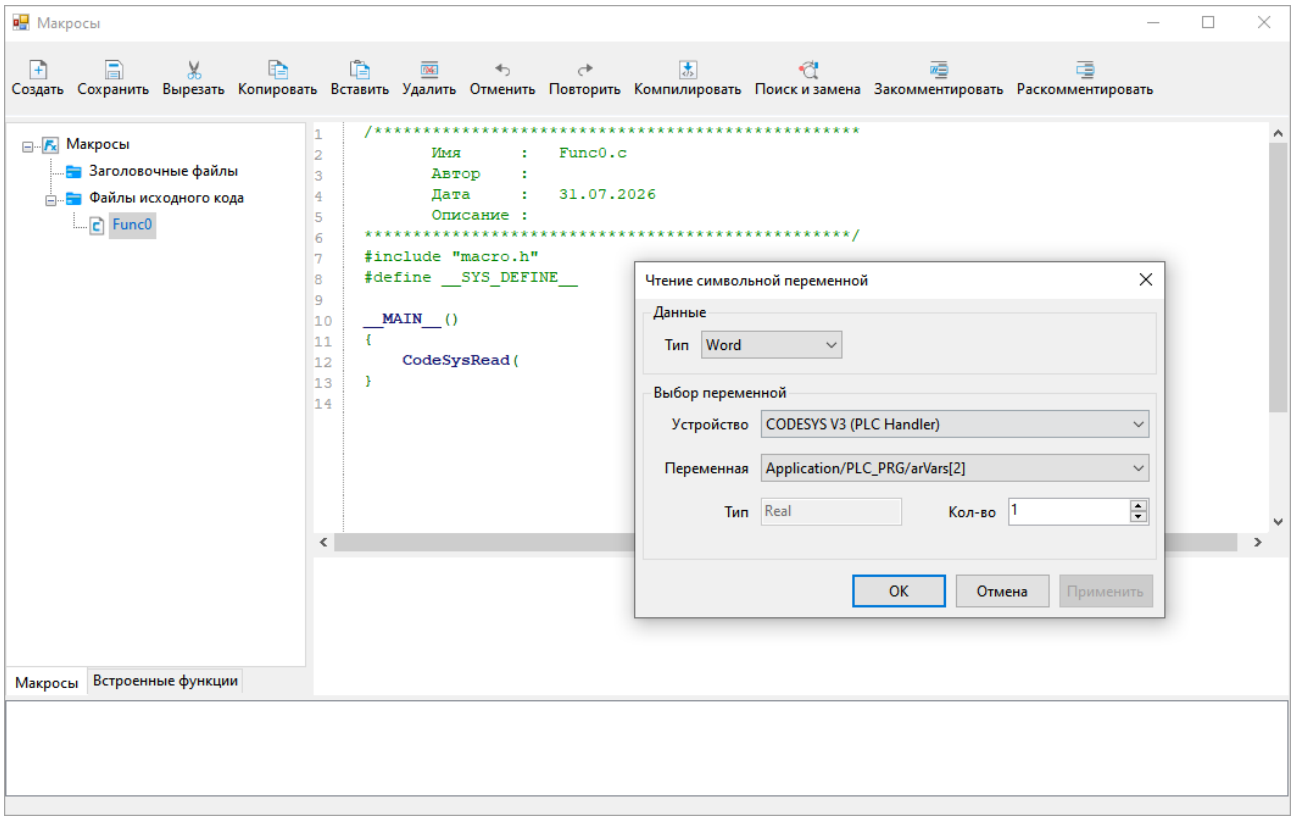


Рисунок 12.7 – Чтение символьной переменной

Таблица 12.19 – Описание функции CodeSysRead

Определение	BOOL CodeSysRead(int Device, char* LabelName, int Count, int LabelType, void* pData);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
LabelName	Путь к переменной, разделитель – «/»
Count	Количество элементов массива (для переменных элементарного типа устанавливается значение **1**)
LabelType	Тип переменной (BOOL = 0, BYTE = 1, WORD = 2, DWORD = 3, LWORD = 4, SINT = 5, USINT = 6, INT = 7, UINT = 8, DINT = 9, UDINT = 10, LINT = 11, REAL = 12, LREAL = 13, STRING = 14, TIME = 15, TIME_OF_DAY = 16, DATE = 17, DATE_AND_TIME = 18, WSTRING = 21)
pValue	Указатель на буфер, в котором будет размещено считанное значение (размер буфера должен соответствовать типу данных)
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  WORD wVar;
  CodeSysRead(_T("OPC UA Client"), "Application/PLC_PRG/wTestVar", 1, 2, &wVar);
  PSW[10] = wVar;
}
```

2. Функция CodeSysWrite

Функция **CodeSysWrite** используется для записи значения переменной макроса в переменную вкладки **Библиотека переменных – С устройства**. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **CodeSysWrite** и поставьте открывающую скобку.

Таблица 12.20 – Описание функции CodeSysWrite

Определение	BOOL CodeSysWrite (int Device, char* LabelName, int Count, int LabelType, void* pData);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
LabelName	Путь к переменной, разделитель – «/»
Count	Количество элементов массива (для переменных элементарного типа устанавливается значение **1**)
LabelType	Тип переменной (BOOL = 0, BYTE = 1, WORD = 2, DWORD = 3, LWORD = 4, SINT = 5, USINT = 6, INT = 7, UINT = 8, DINT = 9, UDINT = 10, LINT = 11, REAL = 12, LREAL = 13, STRING = 14, TIME = 15, TIME_OF_DAY = 16, DATE = 17, DATE_AND_TIME = 18, WSTRING = 21)
pValue	Указатель на буфер, в котором размещается записываемое значение (размер буфера должен соответствовать типу данных)
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  WORD wVar = 123;
  CodeSysWrite(_T("OPC UA Client"), "Application/PLC_PRG/wTestVar", 1, 2, &wVar);
}
```

3. Функция CodeSysReadString

Функция **CodeSysReadString** используется для чтения значения строковой переменной из вкладки **Библиотека переменных – С устройства** в переменную макроса. Для строк типа **WSTRING** выполняется автоматическая конвертация в **ASCII**. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **CodeSysReadString** и поставьте открывающую скобку.

Таблица 12.21 – Описание функции CodeSysReadString

Определение	CodeSysReadString(int Device, char* LabelName, int Count, int Len, void* pData);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
LabelName	Путь к переменной, разделитель – «/»
Len	Количество считываемых символов строки
pData	Указатель на буфер, в котором будет размещено считанное значение (размер буфера должен соответствовать **Len**)
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  char sVar[81] = {0};
  // sTestVar имеет тип STRING(80)
  CodeSysReadString(_T("OPC UA Client"), "Application/PLC_PRG/sTestVar", 1, 80, sVar);
  // Значение переменной будет записано в регистры панели PSW100...PSW139
  // Каждый символ строки типа STRING занимает 1 байт, размер регистра - 2 байта
  // Writes(_T("Внутренние регистры"), 0, TP_PSW, DT_Word, 100, 0, 40, sVar);
}
```

4. Функция CodeSysWriteString

Функция **CodeSysWriteString** используется для записи строковой переменной макроса в переменную вкладки **Библиотека переменных – С устройства**. Для строк типа **WSTRING** должна использоваться кодировка **ASCII** (конвертация будет выполнена автоматически). Строка не должна содержать символы кириллицы. Для настройки параметров функции с помощью вспомогательного меню введите в редакторе кода **CodeSysWriteString** и поставьте открывающую скобку.

Таблица 12.22 – Описание функции CodeSysWriteString

Определение	CodeSysWriteString(int Device, char* LabelName, int Count, int Len,void* pData);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
LabelName	Путь к переменной, разделитель – «/»
Len	Количество символов записываемой строки
pData	Указатель на буфер, в котором размещается записываемое значение (размер буфера должен соответствовать **Len**)
Возвращаемое значение	**TRUE** – успешное выполнение операции, **FALSE** – ошибка (например, не был получен ответ от slave-устройства)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  char sVar[12] = "hello, world";
  CodeSysWriteString(_T("OPC UA Client"), "Application/PLC_PRG/sTestVar", 1, 12, sVar);
}
```

12.6.4 Функции для реализации пользовательского протокола

Данные функции используются совместно с протоколом **Пользовательский** и позволяют реализовать отправку и получение через заданный COM-порт произвольного набора байт.

1. Функция COMSend

Функция **COMSend** отправляет в COM-порт заданный набор байт.

Таблица 12.23 – Описание функции COMSend

Определение	int COMSend(int Device, char* pData, int BufSize);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
pData	Указатель на буфер отправляемых данных
BufSize	Размер буфера в байтах
Возвращаемое значение	Количество отправленных байт (**-1** – ошибка выполнения функции)
Примечание	1. В меню **Настройки проекта - Устройства** должен быть выбран **Пользовательский протокол** 2. Функция блокирует поток выполнения до завершения своей работы

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "string.h"

__MAIN__()
{
  int result = -1;
  char abyData[20] = "Hello, world";
  // Для использования функции strlen требуется подключить заголовочный файл string.h
  result = COMSend(_T("Пользовательский протокол"), abyData, strlen(abyData) );
}
```

2. Функция COMReceive

Функция **COMReceive** используется для чтения данных из буфера COM-порта.

Таблица 12.24 – Описание функции COMReceive

Определение	int COMReceive(int Device, char* pData, int BufSize, unsigned short TimeOut, unsigned short EndFrameBytes);
Device	ID устройства. Используйте оператор **_T** для конвертации символьного имени устройства в его ID
pData	Указатель на буфер, в котором будут размещены считанные данные
BufSize	Размер буфера в байтах
TimeOut	Время ожидания (в мс) получения **BufSize** байт. По истечению времени происходит завершение работы функции **0** – время ожидания не ограничено
EndFrameBytes	Если в течение времени, требуемого для передачи указанного количества байт, не поступило новых данных, то происходит завершение работы функции. **0** – функция не завершает свою работу до получения **BufSize** байт
Возвращаемое значение	Количество полученных байт (**0** – в буфере отсутствуют данные, **-1** – ошибка выполнения функции)
Примечание	1. В меню **Настройки проекта - Устройства** должен быть выбран **Пользовательский протокол** 2. Функция блокирует поток выполнения до завершения своей работы

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "string.h"

__MAIN__()
{
  int result = -1;
  char abyData[10] = {0};

  // Функция завершает свою работу после получения 10 байт данных
  result = COMReceive(_T("Пользовательский протокол"), abyData, sizeof(abyData), 0, 0);
}
```

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "string.h"

__MAIN__()
{
  int result = -1;
  char abyData[10] = {0};
  // Функция завершает свою работу после получения 10 байт данных
  // Если в течение 1000 мс не получено 10 байт - то функция завершает свою работу
  result = COMReceive(_T("Пользовательский протокол"), abyData, sizeof(abyData), 1000, 0);
}
```

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

#include "string.h"

__MAIN__()
{
  int result = -1;
  char abyData[10] = {0};

  // Функция завершает свою работу после получения 10 байт данных
  // Если в течение 1000 мс не получено 10 байт - то функция завершает свою работу
  // Если в процессе получения данных возникает пауза, длительность которой равна...
  // ...времени передачи 10 байт на установленной для COM-порта скорости...
  // ...то функция завершает свою работу
  result = COMReceive(_T("Пользовательский протокол"), abyData, sizeof(abyData), 1000, 10);
}
```

12.6.5 Функции для работы с элементом Холст

Данные функции используются для работы с элементом **Холст**.

1. Функция DCMapClear

Функция **DCMapClear** используется для очистки холста.

Таблица 12.25 – Описание функции DCMapClear

Определение	BOOL DCMapClear(DWORD dwDCMapID);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  DCMapClear(0);
}
```

2. Функция DCMapDrawCircle

Функция **DCMapDrawCircle** используется для отрисовки на холсте окружности.

Таблица 12.26 – Описание функции DCMapDrawCircle

Определение	BOOL DCMapDrawCircle(DWORD dwDCMapID, int x, int y, int Radius, int LineWidth, DWORD LineColor, BOOL DoFill, DWORD FillColor);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
x	Координата центра окружности по оси X (относительно верхнего левого края холста)
y	Координата центра окружности по оси Y (относительно верхнего левого края холста)
Radius	Радиус окружности
LineWidth	Толщина контура
LineColor	Цвет контура окружности в BGR
DoFill	Флаг заливки окружности цветом (**1** – заливка, **0** – отсутствие заливки)
FillColor	Цвет заливки окружности в BGR
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  DCMapDrawCircle(0, 200, 200, 100, 2, 0x0000000, 1, 0xFF0000);
}
```

3. Функция DCMapDrawCircleArc

Функция **DCMapDrawCircleArc** используется для отрисовки на холсте дуги.

Таблица 12.27 – Описание функции DCMapDrawCircleArc

Определение	BOOL DCMapDrawCircleArc(DWORD dwDCMapID, int x, int y, int Radius, int LineWidth, DWORD LineColor, DWORD StartAngle, DWORD EndAngle);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
x	Координата центра дуги по оси X (относительно верхнего левого края холста)
y	Координата центра дуги по оси Y (относительно верхнего левого края холста)
Radius	Радиус дуги
LineWidth	Толщина контура
LineColor	Цвет контура окружности в BGR
StartAngle	Угол начала дуги
EndAngle	Угол конца дуги
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  DCMapDrawCircleArc(0, 200, 200, 100, 2, 0x0000000, 70, 190);
}
```

4. Функция DCMapDrawEllipse

Функция **DCMapDrawEllipse** используется для отрисовки на холсте эллипса.

Таблица 12.28 – Описание функции DCMapDrawEllipse

Определение	BOOL DCMapDrawCircleArc(DWORD dwDCMapID, int x, int y, int xRadius, int yRadius, int LineWidth, DWORD LineColor, DWORD StartAngle, DWORD EndAngle);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
x	Координата центра эллипса по оси X (относительно верхнего левого края холста)
y	Координата центра эллипса по оси Y (относительно верхнего левого края холста)
xRadius	Радиус по X
yRadius	Радиус по Y
LineWidth	Толщина контура
LineColor	Цвет контура в BGR
DoFill	Флаг заливки эллипса цветом (**1** – заливка, **0** – отсутствие заливки)
FillColor	Цвет заливки эллипса в BGR
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"

#define __SYS_DEFINE__


__MAIN__()

{

  DCMapDrawEllipse(0, 200, 200, 100, 50, 2, 0x0000000, 1, 0xFF0000);

}
```

5. Функция DCMapDrawEllipseArc

Функция **DCMapDrawEllipseArc** используется для отрисовки на холсте эллиптической дуги.

Таблица 12.29 – Описание функции DCMapDrawEllipseArc

Определение	BOOL DCMapDrawCircleArc(DWORD dwDCMapID, int x, int y, int Radius, int LineWidth, DWORD LineColor, DWORD StartAngle, DWORD EndAngle);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
x	Координата центра эллиптической дуги по оси X (относительно верхнего левого края холста)
y	Координата центра эллиптической дуги по оси Y (относительно верхнего левого края холста)
Radius	Радиус эллиптической дуги
LineWidth	Толщина контура
LineColor	Цвет контура в BGR
StartAngle	Угол начала эллиптической дуги
EndAngle	Угол конца эллиптической дуги
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"

#define __SYS_DEFINE__


__MAIN__()

{

  DCMapDrawEllipseArc(0, 200, 200, 100, 50, 2, 0x0000000, 70, 190);

}
```

6. Функция DCMapDrawLine

Функция **DCMapDrawLine** используется для отрисовки на холсте линии.

Таблица 12.30 – Описание функции DCMapDrawLine

Определение	BOOL DCMapDrawLine(DWORD dwDCMapID, int xStart, int yStart, int xEnd, int yEnd, int LineWidth, DWORD LineColor);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
xStart	Координата начальной точки по оси X (относительно верхнего левого края холста)
yStart	Координата начальной точки по оси Y (относительно верхнего левого края холста)
xEnd	Координата конечной точки по оси X (относительно верхнего левого края холста)
yEnd	Координата конечной точки по оси Y (относительно верхнего левого края холста)
LineWidth	Толщина линии
LineColor	Цвет линии в BGR
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"

#define __SYS_DEFINE__


__MAIN__()

{

  DCMapDrawLine(0, 200, 200, 400, 500, 2, 0x0000000);

}
```

7. Функция DCMapDrawRect

Функция **DCMapDrawRect** используется для отрисовки на холсте прямоугольника.

Таблица 12.31 – Описание функции DCMapDrawRect

Определение	BOOL DCMapDrawRect(DWORD dwDCMapID, int x, int y, int Width, int Height, int LineWidth, DWORD LineColor, BOOL DoFill, DWORD FillColor);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
x	Координата начальной точки прямоугольника по оси X (относительно верхнего левого края холста)
y	Координата начальной точки прямоугольника по оси Y (относительно верхнего левого края холста)
Width	Ширина
Height	Высота
LineWidth	Толщина контура
LineColor	Цвет контура в BGR
DoFill	Флаг заливки прямоугольника цветом (**1** – заливка, **0** – отсутствие заливки)
FillColor	Цвет заливки прямоугольника в BGR
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  DCMapDrawRect(0, 200, 200, 100, 150, 2, 0x0000000, 1, 0xFF0000);
}
```

8. Функция DCMapSetBackColor

Функция **DCMapSetBackColor** используется для установки фона холста.

Таблица 12.32 – Описание функции DCMapSetBackColor

Определение	BOOL DCMapSetBackColor(DWORD dwDCMapID, DWORD Color);
dwDCMapID	Индекс холста (см. настройки элемента Холст)
Color	Цвет фона холста в BGR
Возвращаемое значение	**TRUE** – операция успешно выполнена, **FALSE** – операция не выполнена (холст с таким индексом не существует)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  DCMapSetBackColor(0, 0xFF0000);
}
```

12.6.6 Функции для работы с экранами и окнами

1. Функция ScreenJump

Функция **ScreenJump** используется для переключения экрана.

Таблица 12.33 – Описание функции ScreenJump

Определение	WORD ScreenJump(WORD ScreenNo);
ScreenNo	Номер экрана
Возвращаемое значение	Номер экрана, на который был осуществлен переход. Переключение не произойдет, если у текущего пользователя недостаточно прав доступа для перехода на экран – тогда функция вернет номер текущего экрана, на котором остался пользователь

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Переход на экран 2
  // В случае успешного перехода переменная wCurrentNo получит значение 2
  WORD wCurrentNo;
  wCurrentNo = ScreenJump(2);
}
```

2. Функция OpenWindow

Функция **OpenWindow** используется для открытия окна.

Таблица 12.34 – Описании функции OpenWindow

Определение	void OpenWindow(int winNo, int x, int y);
winNo	Номер окна
x	Координата открытия по X (для верхней левой точки окна)
y	Координата открытия по Y (для верхней левой точки окна)
Возвращаемое значение	Отсутствует

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  // Открыть окно с номером 2
  // Верхней левой точке окна будут соответствовать координаты (10, 10)
  OpenWindow(2, 10, 10);
}
```

3. Функция CloseWindow

Функция **CloseWindow** используется для закрытия окна.

Таблица 12.35 – Описание функции CloseWindow

Определение	void CloseWindow(WORD winNo);
winNo	Номер окна
Возвращаемое значение	Отсутствует

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  // Закрыть окно с номером 2
  CloseWindow(2);
}
```

12.6.7 Функции для обработки данных

1. Макрос GET_BIT

Макрос **GET_BIT** используется для извлечения указанного бита целочисленной переменной.

Таблица 12.36 – Описание макроса GET_BIT

Определение	bool GET_BIT(Value, BitNumber);
Value	Целочисленная переменная
BitNumber	Номер бита
Возвращаемое значение	Значение бита – **1** (true) или **0** (false)
Примечание	Для использования макросов **true** и **false** следует подключить заголовочный файл **stdbool.h**

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Приведенный ниже фрагмент кода выполнится только в том случае…
  // ...если PSW300.15 имеет значение true
  if (GET_BIT(PSW[300], 15) )
  {
    // ваш код
  }


}
```

2. Макрос SET_BIT

Макрос **SET_BIT** используется для установки указанного бита целочисленной переменной в значение **true**.

Определение	SET_BIT(Value, BitNumber);
Value	Целочисленная переменная
BitNumber	Номер бита
Возвращаемое значение	Целочисленное значение с установленным битом
Примечание	Для использования макросов **true** и **false** следует подключить заголовочный файл **stdbool.h**

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Установка бита PSW300.15 в значение true
  PSW[300] = SET_BIT(PSW[300], 15);
}
```

3. Макрос RESET_BIT

Макрос **RESET_BIT** используется для установки указанного бита целочисленной переменной в значение **false**.

Таблица 12.37 – Описание макроса RESET_BIT

Определение	RESET_BIT(Value, BitNumber);
Value	Целочисленная переменная
BitNumber	Номер бита
Возвращаемое значение	Целочисленное значение со сброшенным битом
Примечание	Для использования макросов **true** и **false** следует подключить заголовочный файл **stdbool.h**

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Установка бита PSW300.15 в значение false
  PSW[300] = RESET_BIT(PSW[300], 15);
}
```

4. Макрос HIWORD

Макрос **HIWORD** используется для получения старшего слова значения типа **DWORD**.

Таблица 12.38 – Описание макроса HIWORD

Определение	WORD HIWORD(DWORD dw);
dw	Значение типа DWORD
Возвращаемое значение	Старшее слово dw

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  DWORD dwVar = 0xFFFFAAAA;
  // PSW0 получит значение 65535 (0xFFFF)
  PSW[0] = HIWORD(dwVar);
}
```

5. Макрос LOWORD

Макрос **LOWORD** используется для получения младшего слова значения типа **DWORD**.

Таблица 12.39 – Описание макроса LOWORD

Определение	WORD LOWORD(DWORD dw);
dw	Значение типа DWORD
Возвращаемое значение	Младшее слово dw

Пример

```
#include "macro.h"
#define __SYS_DEFINE__

__MAIN__()
{
  DWORD dwVar = 0xAAAAFFFF;
  // PSW0 получит значение 65535 (0xFFFF)
  PSW[0] = LOWORD(dwVar);
}
```

6. Макрос HIBYTE

Макрос **HIBYTE** используется для получения старшего байта значения типа **WORD**.

Определение	BYTE HIBYTE(WORD w);
w	Значение типа WORD
Возвращаемое значение	Старший байт w

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  WORD wVar = 0xFFAA;
  // PSW0 получит значение 255 (0xFF)
  PSW[0] = HIBYTE(wVar);
}
```

7. Макрос LOBYTE

Макрос **LOBYTE** используется для получения младшего байта значения типа **WORD**.

Таблица 12.40 – Описание макроса LOBYTE

Определение	BYTE LOBYTE(WORD w);
w	Значение типа WORD
Возвращаемое значение	Младший байт w

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  WORD wVar = 0xAAFF;
  // PSW0 получит значение 255 (0xFF)
  PSW[0] = LOBYTE(wVar);
}
```

8. Макрос MAKEDWORD

Макрос **MAKEDWORD** используется для «сборки» значения типа **DWORD** из двух значений типа **WORD**.

Определение	DWORD MAKEDWORD(WORD wl, WORD wh);
wl	Младшее слово значения
wh	Старшее слово значения
Возвращаемое значение	Значение типа DWORD

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  WORD wLow = 0xAAAA;
  WORD wHigh = 0xFFFF;
  DWORD dwVar;
  // dwVar получит значение 0xFFFFAAAA
  *(DWORD*)(PSW + 0) = MAKEDWORD(wLow, wHigh);
}
```

9. Макрос MAKEWORD

Макрос **MAKEWORD** используется для «сборки» значения типа **WORD** из двух значений типа **BYTE**.

Таблица 12.41 – Описание макроса MAKEWORD

Определение	DWORD MAKEWORD(BYTE byl, BYTE byh);
byl	Младший байт значения
byh	Старший байт значения
Возвращаемое значение	Значение типа WORD

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  BYTE byLow = 0xAA;
  BYTE byHigh = 0xFF;
  // PSW0 получит значение 65450 (0xFFAA)
 PSW[0] = MAKEWORD(byLow, byHigh);
}
```

10. Макрос MAX

Макрос **MAX** возвращает наибольшее из двух числовых значений.

Таблица 12.42 – Описание макроса MAX

Определение	MAX(a, b);
a	Числовое значение
b	Числовое значение
Возвращаемое значение	Наибольшее из a и b

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  float rVar1 = 11.22;
  float rVar2 = 22.33;
  float rMax;
  // rMax получит значение 22.33
  rMax = MAX(rVar1, rVar2);
 *(float*)(PSW + 0) = rMax;
}
```

11. Макрос MIN

Макрос **MIN** возвращает наименьшее из двух числовых значений.

Таблица 12.43 – Описание макроса MIN

Определение	MIN(a, b);
a	Числовое значение
b	Числовое значение
Возвращаемое значение	Наименьшее из a и b

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  float rVar1 = 11.22;
  float rVar2 = 22.33;
  float rMin;
  // rMin получит значение 11.22
  rMin = MIN(rVar1, rVar2)
 *(float*)(PSW + 0) = rMin;
}
```

12.6.8 Остальные функции

1. Функция Beep

Функция **Beep** используется для включения зуммера.

Таблица 12.44 – Описание функции Beep

Определение	void Beep(void);
Возвращаемое значение	Отсутствует
Примечание	Не имеет эффекта, если зуммер отключен (**Настройки проекта – Общие – Отключить зуммер**)

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Пока бит PSB300 имеет значение TRUE - зуммер будет активен
  if (GetPSB(300) )
  {
    Beep();
  }
}
```

2. Функция Delay

Функция **Delay** организует синхронную (блокирующую) задержку выполнения кода функции.

Таблица 12.45 – Описание функции Delay

Определение	void Delay(UINT ms);
ms	Интервал задержки в миллисекундах
Возвращаемое значение	Отсутствует
Примечание	Если вызывается в функции с типом вызова **Синхронная**, то блокирует работу всех остальных функций в пределах элемента, к которому привязана

Пример

```
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
  // Задержка выполнения следующих строк кода на 1 секунду
  Delay(1000);
}
```

3. Функции Lock и UnLock

Функция **Lock** организует блокировку доступа к секции кода для исключения несогласованного доступа к одним и тем же данным из разных функций. Функция **UnLock** снимает блокировку.

Таблица 12.46 – Описание функций Lock и UnLock

Определение	void Lock(int id); void UnLock(int id);
id	Идентификатор секции (0...9)
Возвращаемое значение	Отсутствует
Примечание	1. Каждому вызову функции **Lock** должен соответствовать вызов функции **UnLock** 2. Рекомендуется по возможности избегать доступа к одним и тем же данным из разных функций

Пример

```
// Функции func0 и func1 вызываются независимо друг от друга...
// ...и осуществляют запись в один и тот же регистр
// (по возможности такого поведения стоит избегать)
// Для осуществления согласованного доступа (чтобы функции не начали перезаписывать...
// ...регистр одновременно, «испортив» его значение) используется блокировка

void func0()
{
  Lock(0);
  PSW[300] = 55;
  UnLock(0);
}

void func1()
{
  Lock(0);
  PSW[300] = 66;
  UnLock(0);
}
```

12.6.9 Добавление пользовательских макросов на вкладку Встроенные функции

У пользователя отсутствует возможность подключения дополнительных библиотек, написанных на языке Си. Вместо этого можно добавить на вкладку **Встроенные функции** пользовательские макросы. Для этого следует в директории установки конфигуратора:

- открыть файл **\Tool\Macro\sys\macro_def.h**;
- в конец файла с помощью директивы **#define** добавить новый макрос;
- в папке **\Tool\Macro\Definition** добавить **.html** файл, имя которого должно совпадать с именем макроса. В качестве примера структуры файла смотрите уже существующие файлы данной директории.

12.6.10 Синхронное и асинхронное выполнение макросов

При настройке в графическом элементе действия **Вызов макроса** следует указать режим исполнения: синхронный или асинхронный. Эта настройка определяет, блокирует ли выполнение макроса работу остальных макросов проекта: **синхронный** — блокирует, **асинхронный** — не блокирует.

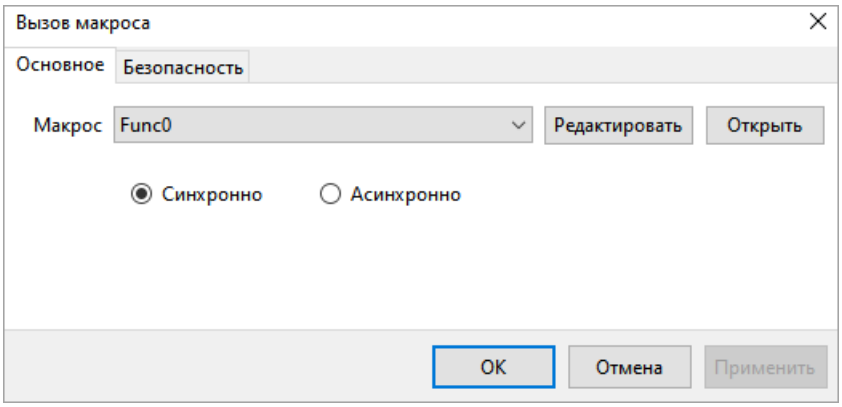


Рисунок 12.8 – Вызов макроса

Рассмотрим пример. Пусть в проекте создано два макроса:

Пример

```
/*************************************************
   Name    : Func0.c
*************************************************/
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
   PSW[0] = 1;
   Delay(5000); // 5000 мс = 5 секунд
}
```

Пример

```
/***************************************************
   Name    : Func1.c
***************************************************/
#include "macro.h"
#define __SYS_DEFINE__


__MAIN__()
{
   PSW[1] = 2;
}
```

Значения регистров **PSW0** и **PSW1** привязаны к цифровым дисплеям.

Создана функциональная область с режимом запуска бит управления **(PSB0)**, в которой настроен вызов макросов **Func0** и **Func1**. На экране также размещен переключатель, устанавливающий бит **PSB0** в значение **TRUE** и таким образом запускающий исполнение обоих макросов.

Если для макроса **Func0** настроен режим исполнения **Синхронный**, то сразу после нажатия на переключатель в регистр **PSW0** будет записано значение **1**, а спустя 5 секунд в регистр **PSW1** будет записано значение **2**. То есть фактически макрос **Func1** будет запущен на исполнение только после завершения работы макроса **Func0**, в котором с помощью функции **Delay** организована блокирующая задержка.

Если для макроса **Func0** настроен режим исполнения **Асинхронный**, то сразу после нажатия на переключатель в регистр **PSW0** будет записано значение **1**, а в регистр **PSW1** – значение **2**. То есть фактически макрос **Func1** будет запущен на исполнение одновременно с **Func0**.

При этом режим исполнения макроса **Func1** не повлияет на описанное выше поведение; режим исполнения определяет, будет ли настраиваемый макрос блокировать исполнение других макросов проекта.



ПРИМЕЧАНИЕ
Синхронные макросы не блокируют обработку визуализации панели и обмен.

12.6.11 Особенности работы с файлами

При работе с файлами следует учитывать следующие особенности:

- для работы с файлами используются функции заголовочного файла **stdio.h**;
- для работы с директориями используются функции заголовочного файла **unistd.h** (mkdir, rmdir);
- директория монтирования USB-накопителя: **/mnt/udisk**;
- требования к USB-накопителю: файловая система FAT32, объём – не более 32 Гб;
- перед работой с USB-накопителем следует проверить факт его монтирования с помощью системного бита **SPSB15**. Попытка доступа к непримонтированному накопителю может привести к исключению в приложении панели;
- для размонтирования USB-накопителя перед его извлечением следует установить системный бит **SPSB201**. Извлечение накопителя «на горячую» (без размонтирования) может привести к повреждению файлов или выходу из строя самого накопителя;
- в режиме эмуляции можно работать с файлами на ПК (при условии, что эти файлы доступны для пользователя, от имени которого запускается конфигуратор). В этом случае пути к файлам должны экранироваться: **C:\\test_folder\\test_file.txt**.

  В качестве рабочей директории используется директория установки конфигуратора;
- закрытие файла (**fclose**) не гарантирует, что сразу после этого буферизированные данные будут записаны в файл. Используйте функцию **fflush** для принудительной записи. При работе с директориями для аналогичной цели можно использовать функцию **system("sync")** из заголовочного файла **stdlib.h**.

13 Приложение

13.1 Системные биты и регистры

Таблица 13.1 – Системные биты (SPSB)

Название	Адрес	Тип доступа	Описание
Флаг авторизации	SPSB0	Только чтение	Принимает значение **TRUE** после успешной авторизации любого пользователя. Принимает значение **FALSE** после выхода пользователя из системы
Флаг выхода из системы	SPSB1	Только чтение	Принимает значение **TRUE** после успешного выхода из системы любого пользователя. Принимает значение **FALSE** после успешной авторизации любого пользователя
Флаг «Есть свободная flash-память»	SPSB2	Только чтение	Принимает значение **FALSE**, если размер свободной flash-памяти панели становится меньше, чем 100 Кб. Если памяти больше – то бит имеет значение **TRUE**
Флаг «Недостаточно flash-памяти»	SPSB3	Только чтение	Принимает значение **TRUE** в случае исчерпания flash-памяти панели. Одновременно с этим происходит открытие системного окна **25005**
Флаг заставки	SPSB4	Только чтение	Принимает значение **TRUE** в случае включения экрана заставки (**Настройки проекта – Общие – Заставка**). Принимает значение **FALSE** после перехода на любой другой экран
Флаг отключения подсветки	SPSB5	Только чтение	Принимает значение **TRUE** в случае отключения подсветки (**Настройки проекта – Общие – Заставка**). Принимает значение **FALSE** после перехода на любой экран
Флаг загрузки проекта	SPSB7	Только чтение	Принимает значение **TRUE** после загрузки проекта из конфигуратора или с USB-накопителя
Флаг включения панели	SPSB8	Только чтение	Принимает значение **TRUE** после включения панели
Генератор прямоугольных импульсов (100 мс)	SPSB9	Только чтение	100 мс – **TRUE**, 100 мс – **FALSE**
Генератор прямоугольных импульсов (1 с)	SPSB10	Только чтение	1 с – **TRUE**, 1 с – **FALSE**
Генератор прямоугольных импульсов (1 минута)	SPSB11	Только чтение	1 минута – **TRUE**, 1 минута – **FALSE**
Бит, который всегда имеет значение TRUE	SPSB13	Только чтение	
Бит, который всегда имеет значение FALSE	SPSB14	Только чтение	
Флаг подключения USB-накопителя	SPSB15	Только чтение	**TRUE** – накопитель подключен, **FALSE** – накопитель не подключен
Флаг подключения к MQTT-брокеру	SPSB19	Только чтение	**TRUE** – подключение установлено, **FALSE** – подключение отсутствует

Продолжение таблицы 13.1

Название	Адрес	Тип доступа	Описание
Флаг возможности подключения по VNC	SPSB22	Только чтение	**TRUE** – подключение по VNC разрешено, **FALSE** – подключение по VNC запрещено
Флаг подключения к панели из конфигуратора	SPSB23	Только чтение	**TRUE** – установлено подключение из конфигуратора (см. **Инструменты – VNC и MQTT**)
Общий флаг ошибки обмена	SPSB48	Только чтение	Кратковременно устанавливается в **TRUE** при ошибке обмена по любому из интерфейсов
Флаг ошибки обмена по COM1	SPSB49	Только чтение	Кратковременно устанавливается в **TRUE** при ошибке обмена
Флаг ошибки обмена по COM2	SPSB50	Только чтение	
Флаг ошибки обмена по Ethernet	SPSB52	Только чтение	
Очистка истории тревог	SPSB120	Чтение и запись	
Установка системного времени	SPSB123	Только запись	По переднему фронту происходит применение значений регистров системного времени
Установка сетевых настроек	SPSB124	Только запись	По переднему фронту происходит применение значений системных регистров сетевых настроек. См. таблицу 11.18
Ошибка протокола CODESYS	SPSB174	Чтение и запись	**TRUE** – ошибка связи по протоколу CODESYS. Часто связана с тем, что в импортированном файле присутствуют переменные, в данный момент уже отсутствующие в приложении ПЛК
Команда перезагрузки панели	SPSB200	Только запись	Должен быть привязан к элементу **Переключатель** или **Переключатель с индикацией**. Не может быть использован в элементах **Функциональная кнопка** и **Функциональная область** (в т. ч. в макросах)
Команда размонтирования USB-накопителя	SPSB201	Только запись	
Принудительное сохранение всех PFW-регистров	SPSB205	Только запись	Должен быть привязан к элементу **Переключатель** или **Переключатель с индикацией**. Не может быть использован в элементах **Функциональная кнопка** и **Функциональная область** (в т. ч. в макросах). По переднему фронту происходит запись значений всех PFW-регистров в энергонезависимую память панели

Таблица 13.2 – Системные оперативные регистры (SPSW)

Название	Адрес	Тип данных	Тип доступа	Описание
ID панели	SPSW0-15	String[32]	Только чтение	
Год (DEC)	SPSW16	Word	Только чтение	Регистры системного времени
Месяц (DEC)	SPSW17	Word	Только чтение	
День (DEC)	SPSW18	Word	Только чтение	

Продолжение таблицы 13.2

Название	Адрес	Тип данных	Тип доступа	Описание
Час (DEC)	SPSW19	Word	Только чтение	
Минута (DEC)	SPSW20	Word	Только чтение	
Секунда (DEC)	SPSW21	Word	Только чтение	
День недели (DEC)	SPSW22	Word	Только чтение	**1** – понедельник, **7** – воскресенье
Год (HEX)	SPSW23	Word	Только чтение	
Месяц (HEX)	SPSW24	Word	Только чтение	
День (HEX)	SPSW25	Word	Только чтение	
Час (HEX)	SPSW26	Word	Только чтение	
Минута (HEX)	SPSW27	Word	Только чтение	
Секунда (HEX)	SPSW28	Word	Только чтение	
День недели (HEX)	SPSW29	Word	Только чтение	
Номер текущего экрана	SPSW30	Word	Только чтение	Соответствует номеру экрана в дереве проекта
Системный таймер	SPSW31-32	Dword	Только чтение	Время с момента включения панели, в секундах
Аппаратная ревизия панели	SPSW33-42	String[20]	Только чтение	
Кол-во устройств	SPSW43	Word	Только чтение	Количество устройств, добавленных на вкладке **Настройки проекта – Устройства**
COM1: счетчик успешных запросов	SPSW44	Word	Только чтение	Кол-во запросов, на которые был получен корректный ответ
COM1: счетчик ошибок	SPSW45	Word	Только чтение	Кол-во запросов, на которые был получен ответ с некорректной структурой пакета или кодом ошибки протокола Modbus
COM1: счетчик таймаутов	SPSW46	Word	Только чтение	Кол-во запросов, на которые не был получен ответ
COM1: адрес устройства с ошибкой обмена	SPSW47	Word	Только чтение	Адрес последнего устройства, при обмене с которым возникла ошибка или сработал таймаут
COM2: счетчик успешных запросов	SPSW48	Word	Только чтение	Кол-во запросов, на которые был получен корректный ответ

Продолжение таблицы 13.2

Название	Адрес	Тип данных	Тип доступа	Описание
COM2: счетчик ошибок	SPSW49	Word	Только чтение	Кол-во запросов, на которые был получен ответ с некорректной структурой пакета или кодом ошибки протокола Modbus
COM2: счетчик таймаутов	SPSW50	Word	Только чтение	Кол-во запросов, на которые не был получен ответ
COM2: адрес устройства с ошибкой обмена	SPSW51	Word	Только чтение	Адрес последнего устройства, при обмене с которым возникла ошибка или сработал таймаут
IP-адрес панели	SPSW58-61	Word[4]	Только чтение	Каждый регистр – октет адреса
Маска подсети	SPSW62-65	Word[4]	Только чтение	Каждый регистр – октет маски
Шлюз	SPSW66-69	Word[4]	Только чтение	Каждый регистр – октет шлюза
Номер сервисного порта	SPSW70	Word	Только чтение	В данный момент не используется
DNS-сервер	SPSW71-74	Word[4]	Только чтение	Каждый регистр – октет адреса
MAC-адрес	SPSW75-80	Word[6]	Только чтение	Каждый регистр – байт MAC-адреса
Версия сервера данных	SPSW90-99	String[20]	Только чтение	
Кол-во блоков flash-памяти	SPSW82	Word	Только чтение	
Кол-во bad-блоков flash-памяти	SPSW83	Word	Только чтение	
Наработка панели (часы)	SPSW200	Word	Только чтение	Время работы панели с момента её изготовления
Наработка панели (минуты)	SPSW201	Word	Только чтение	
Наработка панели (секунды)	SPSW202	Word	Только чтение	
COM1: код ошибки	SPSW203	Dword	Только чтение	Если **SPFW453 = 1**, то в случае ошибки обмена со slave-устройством в данном регистре записывается код ошибки драйвера протокола (используется для отладки)
COM2: код ошибки	SPSW205	Dword	Только чтение	

Продолжение таблицы 13.2

Название	Адрес	Тип данных	Тип доступа	Описание
Модель панели	SPSW209	Dword	Только чтение	852227 – СП210-05-Б 852739 – СП210-07-Б 853762 – СП210-05-Р 852226 – СП210-07-Р 852482 – СП210-10-Р 853506 – СП210-15-Р
Версия HMI	SPSW211-220	String[20]	Только чтение	
Версия прошивки	SPSW221-230	String[20]	Только чтение	
Объем используемой RAM	SPSW231-232	Float	Только чтение	В мегабайтах
Общий объем RAM	SPSW233-234	Float	Только чтение	
Объем используемой flash-памяти	SPSW235-236	Float	Только чтение	
Общий объем flash-памяти	SPSW237-238	Float	Только чтение	
Буфер для установки настроек системного времени и сетевых настроек	SPSW258-273	-	Чтение и запись	См. Системное время и таблицу 11.18
Режим DHCP-клиента	SPSW480	Word	Чтение и запись	1 – отключить, 0 – включить. Применение происходит по переднему фронту бита **SPSB124**

Таблица 13.3 – Системные энергонезависимые регистры (SPFW)

Название	Адрес	Тип данных	Тип доступа	Описание
IP-адрес Ethernet устройства 1	SPFW1-4	Word[4]	Чтение и запись	
Порт Ethernet устройства 1	SPFW5	Word	Чтение и запись	
IP-адрес Ethernet устройства 2	SPFW6-9	Word[4]	Чтение и запись	
Порт Ethernet устройства 2	SPFW10	Word	Чтение и запись	
…				
IP-адрес Ethernet устройства 50	SPFW246-249	Word[4]	Чтение и запись	
Порт Ethernet устройства 50	SPFW250	Word	Чтение и запись	
Уровень яркости подсветки экрана	SPFW252	Word	Чтение и запись	В диапазоне 1…11

Продолжение таблицы 13.3

Название	Адрес	Тип данных	Тип доступа	Описание
Номер выбранного рецепта	SPFW256	Word	Чтение и запись	
Номер начального экрана	SPFW257	Word	Чтение и запись	
Период неактивности для заставки	SPFW258	Word	Чтение и запись	В минутах
Номер выбранного языка проекта	SPFW260	Word	Чтение и запись	В диапазоне 1…8
IP-адрес панели	SPFW318	Word[4]	Чтение и запись	См. таблицу 11.18
Маска подсети	SPFW322	Word[4]	Чтение и запись	
Шлюз	SPFW326	Word[4]	Чтение и запись	
Номер сервисного порта	SPFW330	Word	Чтение и запись	В данный момент не используется
DNS-сервер	SPFW331	Word[4]	Чтение и запись	См. таблицу 11.18
COM1: тип интерфейса	SPFW335	Word	Только чтение	0 – RS232, 1 – RS485
COM1: адрес устройства	SPFW336	Word	Чтение и запись	Только для режима Master
COM1: скорость обмена	SPFW337	Word	Чтение и запись	48 – 4800 бит/с 96 – 9600 бит/с 192 – 19200 бит/с 384 – 38400 бит/с 576 – 57600 бит/с 1152 – 115200 бит/с 1875 – 187500 бит/с
COM1: кол-во бит данных	SPFW338	Word	Чтение и запись	8 – 8 бит 7 – 7 бит
COM1: кол-во стоп-бит	SPFW339	Word	Чтение и запись	1 – 1 бит 2 – 2 бита
COM1: режим контроля четности	SPFW340	Word	Чтение и запись	0 – None 1 – Odd 2 – Even
COM1: задержка опроса	SPFW341	Word	Чтение и запись	0…32767 мс
COM1: отключение опроса	SPFW400	Word[16]	Чтение и запись	См. пункт 11.9
COM2: тип интерфейса	SPFW343	Word	Только чтение	0 – RS232, 1 – RS485
COM2: адрес устройства	SPFW344	Word	Чтение и запись	Только для режима Master
COM2: скорость обмена	SPFW345	Word	Чтение и запись	48 – 4800 бит/с 96 – 9600 бит/с 192 – 19200 бит/с 384 – 38400 бит/с 576 – 57600 бит/с 1152 – 115200 бит/с 1875 – 187500 бит/с

Продолжение таблицы 13.3

Название	Адрес	Тип данных	Тип доступа	Описание
COM2: кол-во бит данных	SPFW346	Word	Чтение и запись	8 – 8 бит 7 – 7 бит
COM2: кол-во стоп-бит	SPFW347	Word	Чтение и запись	1 – 1 бит 2 – 2 бита
COM2: режим контроля четности	SPFW348	Word	Чтение и запись	0 – None 1 – Odd 2 – Even
COM2: задержка опроса	SPFW349	Word	Чтение и запись	0…32767 мс
COM2: отключение опроса	SPFW416	Word[16]	Чтение и запись	См. пункт 11.9
Отключить зуммер	SPFW448	Word	Чтение и запись	1 – отключить, 0 – включить Отключает звук нажатий на элементы, а также элемент Зуммер и функцию Beep
Скрыть курсор	SPFW449	Word	Чтение и запись	1 – скрыть, 0 – отображать
Скрыть системное меню	SPFW450	Word	Чтение и запись	1 – скрыть кнопку системного меню, 0 – отображать
Режим доступа по VNC	SPFW451	Word	Чтение и запись	1 – запрещено, 0 – разрешено
Отключить подсветку	SPFW452	Word	Чтение и запись	1 – отключить подсветку, 0 – включить подсветку
Режим диагностики обмена	SPFW453	Word	Чтение и запись	1 – включить режим диагностики обмена. В этом режиме: • в случае ошибок обмена в регистр **SPSW205** (COM1) и **SPSW207** (COM2) записывается код ошибки драйвера обмена; • счетчики ошибок увеличиваются (и биты ошибок устанавливаются) не при каждой ошибке, а при каждой N-й ошибке, где N – значение параметра **Кол-во попыток**;
Пароль пользователя 1	SPFW359-362	Word[4]	Чтение и запись	Пароли, заданные в меню **Настройки проекта – Права доступа**. Пароль пользователя 1 соответствует паролю администратора
Пароль пользователя 2	SPFW363-366	Word[4]	Чтение и запись	
Пароль пользователя 3	SPFW367-370	Word[4]	Чтение и запись	
Пароль пользователя 4	SPFW371-374	Word[4]	Чтение и запись	
Пароль пользователя 5	SPFW375-378	Word[4]	Чтение и запись	
Пароль пользователя 6	SPFW379-382	Word[4]	Чтение и запись	
Пароль пользователя 7	SPFW383-386	Word[4]	Чтение и запись	
Пароль пользователя 8	SPFW387-390	Word[4]	Чтение и запись	

Продолжение таблицы 13.3

Название	Адрес	Тип данных	Тип доступа	Описание
Пароль пользователя 9	SPFW391-394	Word[4]	Чтение и запись	
Пароль пользователя 10	SPFW395-398	Word[4]	Чтение и запись	
Пароль пользователя 11	SPFW454-457	Word[4]	Чтение и запись	
Пароль пользователя 12	SPFW458-461	Word[4]	Чтение и запись	
Пароль пользователя 13	SPFW462-465	Word[4]	Чтение и запись	
Пароль пользователя 14	SPFW466-469	Word[4]	Чтение и запись	
Пароль пользователя 15	SPFW470-473	Word[4]	Чтение и запись	
Пароль пользователя 16	SPFW474-477	Word[4]	Чтение и запись	
Пароль пользователя 17	SPFW478-481	Word[4]	Чтение и запись	
Пароль пользователя 18	SPFW482-485	Word[4]	Чтение и запись	
Пароль пользователя 19	SPFW485-488	Word[4]	Чтение и запись	
Пароль пользователя 20	SPFW489-492	Word[4]	Чтение и запись	

13.2 VNC

VNC (Virtual Network Computing) – это технология доступа к экрану устройству, основанная на протоколе **RFB** (Remote FrameBuffer).

Панели оператора СП210 расширенной модификации (-P) могут функционировать в режиме **VNC-сервера**, предоставляя доступ к своему экрану с устройства-клиента (например, компьютера, смартфона и т. д). На клиенте должна быть установлена утилита, выполняющая функцию VNC-клиента (VNC viewer).

Таблица 13.4 – Примеры VNC Viewer

Утилита	Поддерживамые ОС	Тип
RealVNC Viewer	Windows, Linux, Android, macOS, iOS	с закрытым исходным кодом
UltraVNC Viewer	Windows	open-source
TightVNC Viewer	Windows Linux (в старых версиях – 1.3.x)	
TigerVNC Viewer	Windows, Linux, macOS	
AVNC	Android	
bVNC		

В системном меню панели настраивается пароль для подключения по VNC и доступный режим подключения.

В VNC-клиенте указывается IP-адрес панели, порт (**5900**; недоступен для изменения) и пароль. Кроме того, VNC-клиенты обычно имеют настройки отображения (разрешение экрана, количество цветов и т. д.)

Конфигуратор СП210 имеет встроенный **VNC Viewer**.

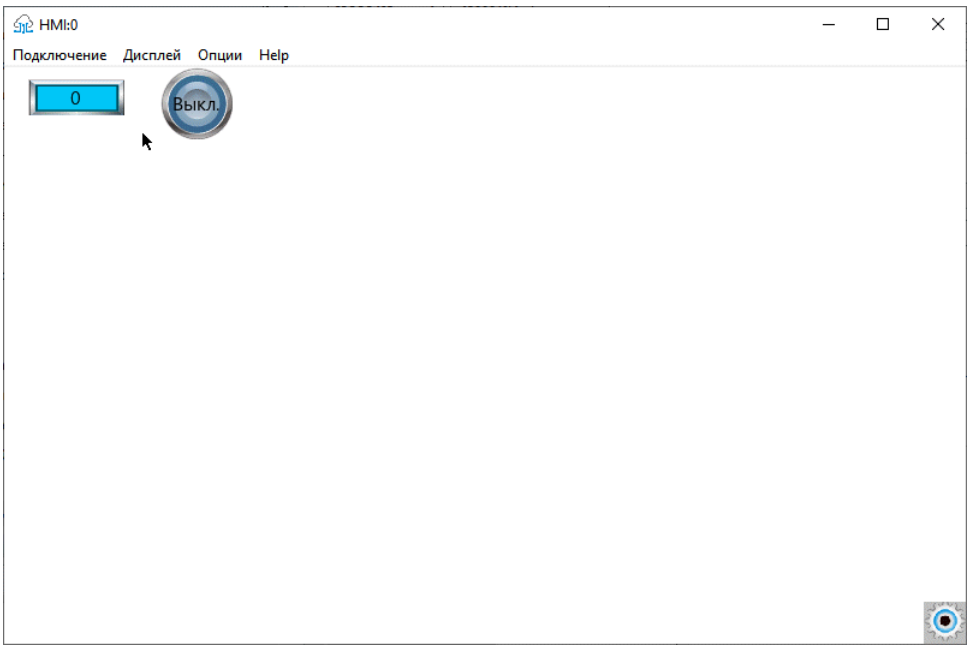


Рисунок 13.1 – Отображение экрана панели во встроенном VNC Viewer

Для использования встроенного **VNC Viewer** следует подключиться к панели с помощью команды **Инструменты – VNC и MQTT**. Для подключения нужно указать ID панели (можно выбрать его из сохранённых с помощью кнопки «↓») и пароль удалённого подключения (по умолчанию – **12345678**).

Нажатие на кнопку **Поиск устройств** производит поиск панелей в локальной сети. ПК отправляет UDP-пакет на широковещательный адрес сети (порт **3333**), в результате чего его получают все панели. Результаты сканирования отображаются в виде таблицы с указанием имени панели, IP-адреса, идентификатора и модификации.

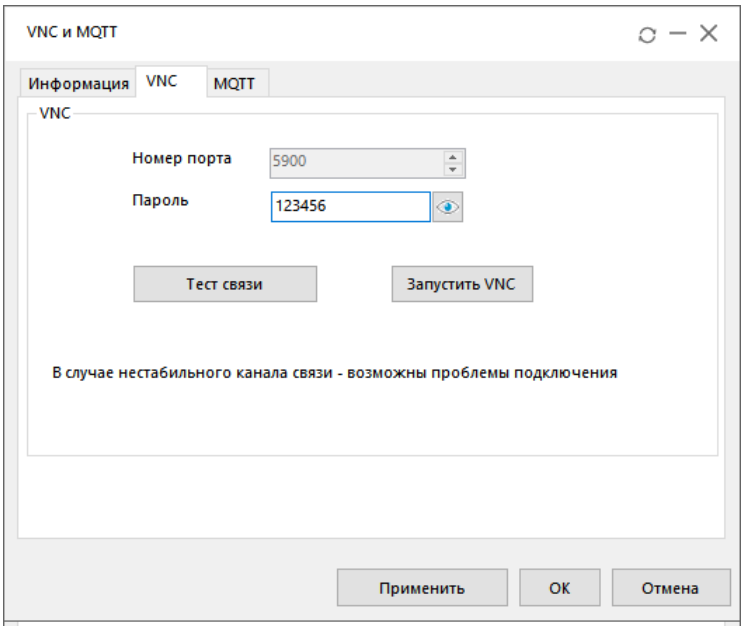


Рисунок 13.2 – Внешний вид вкладки VNC

Нажатие на кнопку **Тест связи** производит проверку связи с панелью (путём отправки ping-запроса), после чего отображается соответствующее информационное сообщение.

Нажатие на кнопку **Подключение** осуществляет подключение к панели.

На вкладке **VNC** следует указать пароль и нажать кнопку **Запустить VNC**.

13.3 Контроль оплаты

Функция **Контроль оплаты** позволяет снизить риск неполной оплаты работ со стороны заказчика. При использовании этой функции время работы приложения панели ограничивается указанным периодом. По истечении периода для продолжения работы требуется ввести специальный код.

ПРИМЕЧАНИЕ
При активации функции блокируется возможность изменения системного времени панели.

Для настройки функции следует использовать команду меню **Инструменты – Контроль оплаты** или одноименную команду панели ярлыков.

Пользователю следует выбрать один из режимов работы (статический или динамический) и установить соответствующую галочку.

Статический контроль

При статическом контроле пользователь создает в проекте один или несколько интервалов времени (с дискретностью до минут), вводя для каждого код активации (не более 10 знаков, не может содержать символы кириллицы и спецсимволы).

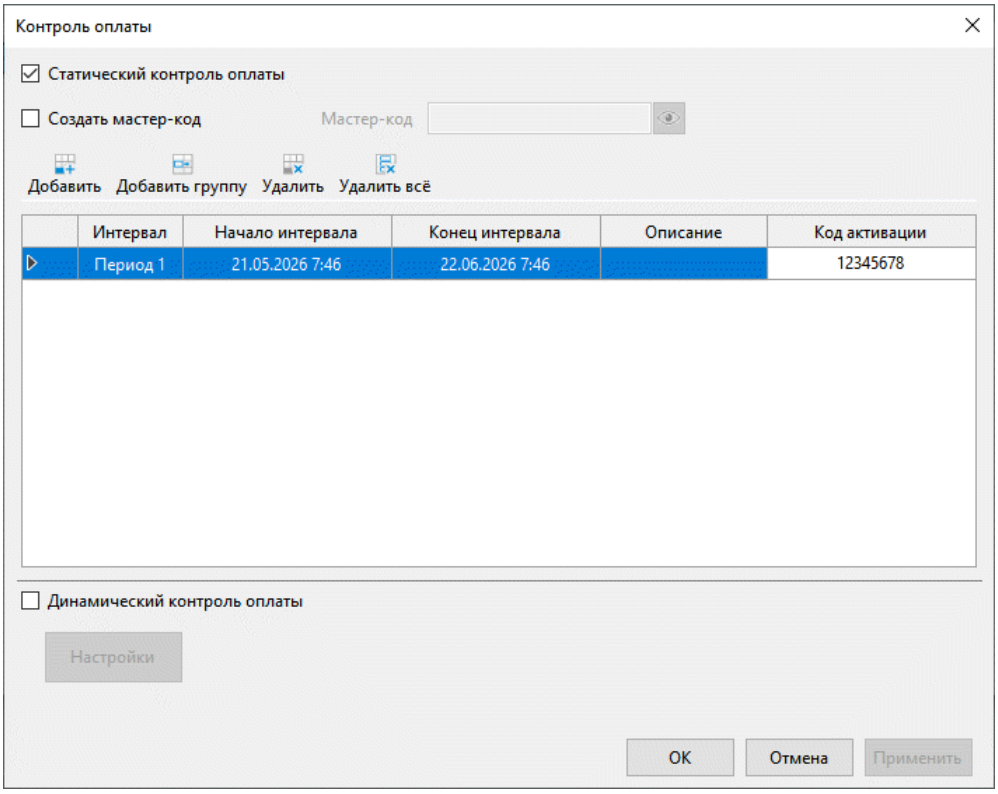


Рисунок 13.3 – Настройки статического контроля оплаты

При наступлении интервала на экране панели отобразится соответствующее окно, в котором нужно ввести код активации для продолжения работы. Работа будет продолжена до конца интервала, после чего окно появится повторно. После истечения последнего интервала приложение панели будет работать без ограничений.

В случае установки галочки **Создать мастер-код** пользователь задает мастер-код. При его вводе в окне панели все следующие интервалы отключаются (например, можно передать заказчику мастер-код после получения полной оплаты).

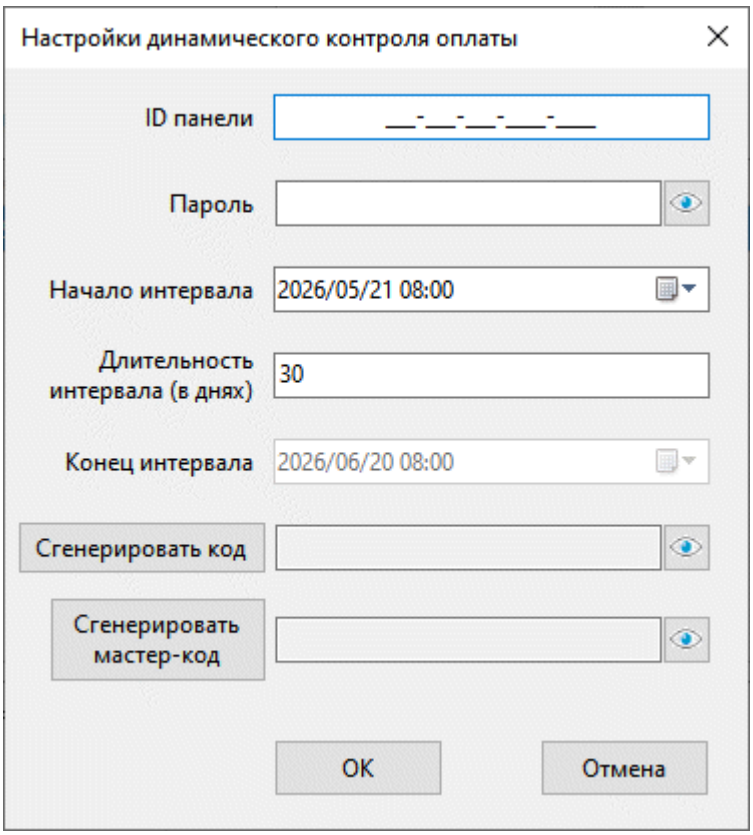


Рисунок 13.4 – Настройки динамического контроля оплаты

Динамический контроль

При динамическом контроле пользователь указывает идентификатор панели и заданный пароль (не более 10 знаков, не может содержать символы кириллицы и спецсимволы). После этого указывается начало интервала времени (с дискретностью до часа) и его продолжительность в днях; далее проект с этими настройками загружается в панель. Выполнить команды **Сгенерировать код** и **Сгенерировать мастер-код** пользователь может в любой момент времени, запустив конфигуратор и указав идентификатор панели, заданный ранее пароль и дату интервала, относительно которой можно будет применить коды. Доступ к панели и загрузка в неё проекта для формирования кодов не требуются. При наступлении интервала на экране панели отобразится соответствующее окно. В случае ввода кода время работы приложения панели будет продлено на указанное количество дней. В случае ввода мастер-кода все следующие интервалы отключаются.

Для использования функции при загрузке проекта из конфигуратора или создания загрузочного проекта для USB-накопителя следует устанавливать галочку **Включить контроль оплаты**.

Таблица 13.5 – Сравнение статического и динамического режима контроля оплаты

Параметр/Режим	Статический	Динамический
Количество интервалов	Заранее предопределенное	Произвольное
Дискретность интервала	Минуты	Часы
Ввод кода	Обеспечивает работу приложения в текущем интервале времени	Продлевает работу приложения на указанное количество дней
Длина кода и мастер-кода	От 1 до 10 знаков, вводится пользователем	32 знака, формируется автоматически как хеш-функция от введенного пользователем пароля
Условие полного снятия ограничений	Истечение последнего интервала или ввод мастер-кода	Ввод мастер-кода

13.4 Подключение устройств к порту USB A

К порту USB A панели могут быть подключены следующие устройства:

- USB-накопитель (для записи/чтения файлов, загрузки проекта, обновления прошивки панели). Требования к накопителю: файловая система FAT32, объем не более 32 Гб;
- клавиатура и мышь;
- сканер штрихкодов.

При подключении устройств к порту USB A следует учитывать, что их максимальный ток потребления не должен превышать **500 мА**. В случае необходимости можно использовать USB-хаб с внешним питанием.

Для модификаций СП210-10-P и СП210-15-P, имеющих два порта USB A, допускается подключение только одного USB-накопителя единовременно.

13.5 Системное время

В этом пункте описаны компоненты панели и проекта, связанные с системным временем.

Системное время панели можно установить в системном меню или экранном конфигураторе.

ПРИМЕЧАНИЕ
При активации функции **Контроль оплаты** блокируется возможность изменения системного времени панели.

Для отображения системного времени на экране панели могут использоваться элементы Дата и Время.

В меню **Файл – Настройки проекта – Время** можно указать источник системного времени панели:

- встроенные часы;
- регистры панели или одного из подключённых к ней slave-устройств. В первом случае запись системного времени в регистры панели должно организовать подключенное к ней master-устройство.

Кроме того, можно настроить циклическое или событийное копирование системного времени панели в одно или несколько подключенных к ней slave-устройств.

Для работы с системным временем (в т. ч., в макросах) можно использовать следующие системные регистры (порядок разрядов: год-месяц-день-час-минута-секунда-день недели):

- **SPSW16…22** – текущее системное время в формате **DEC**;
- **SPSW23…29** – текущее системное время в формате **HEX**;
- **SPSW258…264** – устанавливаемое системное время в формате DEC. Установка происходит по переднему фронту системного бита **SPSB123**. Все перечисленные регистры должны иметь корректные значения.

13.6 Редактирование шаблона проекта

При выполнении команды **Файл** – **Создать проект** открываемый в конфигураторе проект формируется на основе шаблона. Пользователь может отредактировать этот шаблон — например, добавив в него собственные окна экранных клавиатур или изменив существующие. Для редактирования шаблона следует:

- перейти в директорию установки конфигуратора по пути **\HMI\Template**;
- открыть папку, название которой соответствует разрешению дисплея используемой панели с нужной ориентацией экрана (горизонтальной или вертикальной);
- скопировать файл **template-information.xjtmp** в другую пользовательскую директорию;
- изменить расширение файла **template-information** с **.xjtmp** на **.xjp**;
- открыть этот файл в конфигураторе и отредактировать его нужным образом;
- сохранить изменения (**Файл** – **Сохранить**);
- закрыть конфигуратор;
- изменить расширение файла обратно на **.xjtmp**;
- скопировать файл в соответствующую исходную папку конфигуратора с перезаписью существующего файла.

При следующем создании нового проекта будет использован отредактированный шаблон.

Россия, 111024, Москва, 2-я ул. Энтузиастов, д. 5, корп. 5
тел.: +7 (495) 641-11-56, факс: (495) 728-41-45
тех. поддержка 24/7: 8-800-775-63-83, support@owen.ru
отдел продаж: sales@owen.ru
www.owen.ru
рег.:1-RU-162457-1.6