ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ASTRA.IDE

Руководство пользователя

DPA-302

Версия документа 2.17

Версия ПО 1.7.2.0

Август 2024

История изменений руководства пользователя

Версия руководства пользователя	Описание изменения
2.7	Добавлена история изменений руководства пользователя. Добавлены знаки с предупреждающей и поясняющей информацией. *Раздел «Конфигурация задач»:* дополнено описание настройки сторожевого таймера в зависимости от значения параметра восприимчивость. Дополнено описание приоритетов задач. *Раздел «Задание параметров модулей аналогового ввода. Базовые параметры»:* добавлено описание дополнительных параметров для настройки модулей аналогового ввода R X00 AI 0X X31. Раздел «Задание параметров модулей дискретного ввода»: добавлен порядок работы с метками времени. Раздел «Задание параметров модулей дискретного вывода. Задание параметров алгоритма противоаварийной защиты»: актуализировано описание, в частности заменен рисунок с алгоритмом управления каналом. Добавлено описание работы с модулями управляемого коммутатора R000 CP 06 1X1. Раздел «Задание параметров модулей коммуникационного процессора»: добавлено описание параметров: – **Альтернативный MAC адрес** и **Альтернативный IP адрес (для ЦП R500)**; – **Сбросить блокировку**. Добавлен параметр HwError (на вкладку **Соотнесение входов/выходов),** информирующий о статусе состояния шины RegulBus, крейта и модуля соответственно. Добавлены новые разделы: – «Настройка системных параметров»; – «Запуск службы SNMP-сервера»; – «Запись данных на внешний накопитель»; – «Остановка службы NTP при старте»; – «Настройка синхронизации времени по протоколу PTP»; – «Изменение режима работы подсветки дисплея R400»; – «Политика пользователя БД MySQL». Раздел «Обновление ПО контроллера. Обновление системного программного обеспечения»: дополнено описание процесса обновления с предупреждающей информацией. *Раздел «Полный журнал»:* добавлена возможность изменять размер лог-файла и кол-во лог-файлов, прописав параметры в конфигурационном файле */etc/runtime.cfg*. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.8	*Раздел «Основные понятия среды разработки. Конфигурация задач»:* добавлена информация о запрете редактирования параметров планирования системных задач.

Версия руководства пользователя	Описание изменения
	Раздел «Конфигурирование крейтов. Редактор шины»: актуализирована формулы Автонастройки параметров. *Раздел «Задание параметров модулей коммуникационного процессора»:* обновлена информация о редакторе параметров коммуникационного модуля R500 CP 02 021. *Раздел «Сканер сети. Настройка IP-адресов»:* обновлена информация о сетевых параметрах контроллера. *Раздел «Отладка проекта. Запуск и мониторинг приложений»:* добавлена информация о настройке длительности таймаута сессии со средой разработки. *Раздел «Отладка проекта. Сброс приложений»:* добавлена возможность задавать тип и объем хранилища для **RETAIN** переменных. Раздел «Диагностика контроллера. Получение диагностической информации о контроллере»: добавлена возможность настройки логирования загрузки в журнал контроллера. Приложение А: обновлена таблица параметров конфигурационного файла */etc/runtime.cfg*. Добавлены новые разделы: – «Установка MAC-адресов на сетевые интерфейсы»; – «Настройка FTP»: o «Конфигурирование учетных записей»; o «Просмотр информации об учетных записях»; o «Защищенное TLS соединение»; – «Версия SNMP v1»; – «Отключение сенсорного экрана на время загрузки контроллера»; – «Обновление программного обеспечения модулей ввода/вывода»; – «Отключение входа в сервисный режим на время загрузки контроллера». Добавлено приложение Б «Настройка конфигурационного файла */etc/plc.cfg*.» Добавлено приложение В «Настройка конфигурационного файла */etc/network.cfg*» Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.9	*Раздел «Описание интерфейса. Общие сведения»:* добавлена информация про формат отображения вкладок и окон. *Раздел «Основные понятия среды разработки. Устройство, дерево устройств»:* дополнена информация о состояниях, в которых может находиться устройство. *Раздел «Основные понятия среды разработки. Конфигурация задач»:* обновлена информация о редактировании параметров планирования системных задач. Раздел «Привязка каналов к переменным программы. Соотнесение переменных и входов/выходов»: дополнена информация об опции **Всегда обновлять переменные**. Раздел «Запись данных на внешний накопитель. Сценарии копирования пользовательских данных»: обновлено описание параметра **Destination**

Версия руководства пользователя	Описание изменения
	конфигурационного файла copyjobs.cfg. *Раздел «Настройка времени»:* обновлено описание параметра **Root dispersion**. *Приложение Б «Настройка конфигурационного файла /etc/plc.cfg.»:* обновлено описание параметра **AllowPanic** конфигурационного файла /etc/plc.cfg. Добавлены новые разделы: – «Пакетный фильтр»; – «Калибровка сенсорного экрана». Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.10	*Раздел «Основные понятия среды разработки. Конфигурация задач*»: сокращен диапазон допустимых приоритетов (диапазон: от 0 до 15). *Раздел «Конфигурирование крейта. Редактор шины*»: дополнена информация о возможности задания временных параметров внутренней шины крейта R100. Раздел «Особенности размещения модулей и субмодулей в крейте контроллера Regul R100»: новый раздел о добавлении крейтов расширения серии Regul R100. *Раздел «Конфигурирование крейтов. Редактор шины»:* дополнена информация о минимально допустимом значении цикла шины Regul Bus в модулях ЦП. *Раздел «Привязка каналов к переменным программы. Общие сведения»:* дополнена информация о проверке привязок переменных на наличие ручного присвоение адреса. *Раздел «Подключение контроллера к сети. Сканер сети. Настройка IP-адресов»*: дополнена информация о сопровождении звукового сигнала одновременным миганием индикаторов при поиске ПЛК. *Раздел «Пакетный фильтр»*: добавлена информация о журналировании ошибок и изменении правил фильтрации сетевых пакетов в лог-файл system.log. *Раздел «Диагностика контроллера. Получение диагностической информации о контроллере»:* дополнена информация о возможности настроить журналирование загрузки процессора, сведений по сетевым интерфейсам и хранению данных в соответствующие лог-файлы. *Раздел «Журнал событий»:* добавлен новый раздел – «Журнал событий по МЭК задачам» со статистикой по всем задачам. *Раздел «Настройка времени»:* дополнена информация о возможности установки текущего времени на ПЛК. *Раздел «Настройка дисплея. Выбор разрешения дисплея»*: дополнена информация о необходимости наличия физического подключение монитора к модулю ЦП при смене разрешения. *Раздел «Резервное копирование и восстановление»*: обновлена информация в связи со сменой конфигурации при создании файл-образов. *Раздел «Информационная безопасность. Политика пользователей БД MySQL*»: дополнена информация о возможности задания ограничения размера БД и журналировании работы сервера. Приложение Е «Настройка конфигурационного файла /etc/plc.cfg»: добавлено описание секций - [Database], [GNSS], [Logging].

Версия руководства пользователя	Описание изменения
	Описание конфигурационных файлов ptp.conf и copyjobs.cfg перенесено в приложения Ж и Г соответственно. Изменена последовательность следований приложений. Добавлены новые приложения: – «Приложение Б. Типы модулей центрального процессора»; – «Приложение Д. Журналирование системных параметров ПЛК». Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.11	*Раздел «Задание параметров модулей счета импульсов. Настроечные параметры модуля (энкодера*)»: добавлено описание дополнительных параметров для настройки модуля счета импульсов R200 DA 01 111. *Раздел «Установка соединения с контроллером*»: добавлен новый подраздел – «Защита от широковещательного шторма». Добавлено описание реализованной защиты от широковещательного шторма на различных типах ПЛК. *Раздел «Настройка FTP»:* добавлен новый подраздел – «Изменение диапазона портов, используемых FTP сервером в пассивном режиме». Добавлено описание о возможности настраивать диапазон портов в пассивном режиме. *Раздел «Настройка времени»*: добавлен новый подраздел – «Источник времени от МЭК-приложения». В NTPD добавлена возможность получения времени из прикладного ПО. *Раздел «Настройка дисплея*»: добавлен новый подраздел – «Изменение фонового изображения». Добавлена информация о возможности изменения фонового изображения, появляющегося на экране при загрузке. *Приложение А «Настройка конфигурационного файла runtime.cfg*»: добавлены параметры в секцию [PSDebug], определяющие механизм обработки исключений в прикладном ПО. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.12	*Раздел «Настройка параметров модулей»*: добавлен новый раздел – «Задание параметров модулей источника питания». Добавлено описание параметров для настройки модуля источника питания R500 PP 00 051 и модуля источника внешнего питания R500 PO 08 041. *Раздел «Редактор шины»:* добавлено описание ограничений, накладываемых при настройке параметров шины RegulBus при работе с модулями аналогового ввода R500 AI 08 242. *Подраздел «Задание параметров модулей аналогового ввода»:* добавлено описание дополнительных параметров для настройки модуля аналогового ввода R500 AI 08 242. Подраздел «Задание параметров модулей коммуникационного процессора»: добавлен модуль R200 CP 01 021. *Подраздел «Установка соединения с контроллером»:* добавлено описание возможности включения/ выключения процедуры авторизации при подключении к ПЛК. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.13	Выпуск среды разработки Astra.IDE. Добавлена поддержка модулей в среде разработки:

Версия руководства пользователя	Описание изменения
	– R200 DA 01 012; – R500 AI 08 022; – R500 AI 08 042; – R500 AI 08 142; – R500 AI 08 342; – R500 DI 16 032; – R500 DS 32 012; – R500 PO 08 041. *Подраздел «Минимальные системные требования»*: изменены системные требования для установки среды разработки. *Подраздел «Построение конфигурации контроллера с помощью мастера»*: добавлено описание о возможности добавления нативной реализации драйвера шины контроллера (RegulBus OS). *Подраздел «Редактор шины»:* добавлено описание о журналировании работы шины RegulBus OS в отдельный лог-файл. Добавлено описание дополнительных параметров при работе с шиной RegulBus OS и ограничение в применении модулей ввода/вывода определенной версии СПО. *Раздел «Сканер сети. Настройка IP-адресов»*: добавлена возможность получения IP-адресов сетевыми интерфейсами модуля ЦП по протоколу DHCP. Добавлен новый подраздел – «Установка дополнительных IP-адресов». Добавлена возможность установки дополнительных IP-адресов сетевым интерфейсам модуля ЦП. *Раздел «Подключение контроллера к сети»*: добавлен новый подраздел – «Журналирование сетевого трафика». Добавлено описание о возможности перехватывать и анализировать сетевой трафик. *Подраздел «Журнал событий»*: добавлены новые подразделы: – «Журнал действий пользователя»; – «Журнал сообщений операционный (системный)». Раздел «Обновление ПО контроллера»: обновлено описание раздела. *Приложение Е «Настройка конфигурационного файла plc.cfg»*: добавлено описание секции [Syslog] с возможностью отправки сообщений журналов контроллера на syslog сервер. *Удалено «Приложение Ж. Журнал регистрации событий ПЛК»*, в результате изменился порядок: «Приложение З»⇨«Приложение Ж». Добавлены новые приложения: – «Приложение З. Настройка конфигурационного файла netdump.conf»; – «Приложение И. Обработка расширенных данных на шине RegulBus_OS»; – «Приложение К. Библиотеки». Дополнительно по тексту внесены небольшие изменения с уточняющей информацией

Версия руководства пользователя	Описание изменения
2.14	*Раздел «Обновление ПО контроллера»*: начиная с версии 1.7.0.0 доступна заводская установка на ПЛК операционной системы КПДА (ЗОСРВ «Нейтрино»), в связи с этим расширено описание по вариантам обновления. *Раздел «Установка соединения с контроллером. Авторизация при подключении к ПЛК»*: обновлено описание об изменении авторизации пользователя начиная с версии 1.7.0.0. *Раздел «Политика пользователей БД MySQL»*: обновлена служба сервера SQL (поддержка MariaDB 10.2.44) и добавлены новые параметры по настройке в конфигурационный файл /etc/plc.cfg секция [Database] (Приложение Е). Приложение Б «Типы модулей центрального процессора»: добавлены новые модули. *Раздел «Редактор шины»:* добавлено описание нового параметра **Скорость инициализации модулей.** Добавлен новый подпункт «Редактор параметров шины для модулей ЦП III-го типа» с описанием параметров. *Приложение К «Библиотеки»*: добавлено описание компонента резервирования PS_Redundancy_OS. *Приложение И «Обработка расширенных данных и событий модулей»*: скорректировано название раздела и дополнено описание с примерами программного кода. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.15	Добавлены новые разделы: – «Сохранение и восстановление значений переменных RETAIN»; – «Управление яркостью подсветки сенсорного экрана контроллера R400». *Приложение К «Библиотеки»*: в библиотеку PsLed добавлено описание функции setBacklight для управления яркостью экрана. Дополнительно по тексту внесены небольшие изменения с уточняющей информацией
2.16	Добавлены новые подразделы: – «Резервированные сборки модулей ввода/вывода контроллера Regul R500»; – «Добавление устройств к модулям»; – «Особенности обновления СПО до 1.7.0.0 и до 1.7.1.0»; – «Проверка файловой системы на целостность»; – «Задание параметров блоков расширения EU для модулей ЦП III-го типа». *Подраздел «Построение конфигурации контроллера с помощью мастера»:* при создании нового проекта по умолчанию задана внутренняя шина RegulBus OS. *Подраздел* «Особенности размещения модулей в крейте контроллера Regul R500»: появилась защита от подключения двух R500 ЦП на одну шину. *Подраздел «Редактор модуля»*: меняется фон параметра при изменении его значения в онлайн-режиме. *Подраздел «Соотнесение переменных и входов/выходов»:* добавлена возможность получения статуса состояния сетевых портов с интерфейсом Ethernet (Link Status). *Подраздел «Сканер сети. Настройка IP-адресов»:* удален параметр «Линия синхронизации» у ПЛК, не поддерживающих резервирование. Принят запрет на

Версия руководства пользователя	Описание изменения
	установку зарезервированных IP-адресов. *Подраздел «Отладка проекта. Компиляция и загрузка приложения в контроллер»:* на панели инструментов появилась кнопка глобальной компиляции всех приложений в проекте. Приложение А «Настройка конфигурационного файла runtime.cfg»: появились параметры: – **Bootproject.HardwareSelfDiagFail.Deny**, задающий ограничение на загрузку приложения в случае, если ПЛК при включении получил ошибку самодигностики; – появилась возможность в секции [CmpLog] задавать параметры для вывода информации в оперативный журнал из соответствующего лог-файла. *Приложение Е «Настройка конфигурационного файла plc.cfg»:* появилась возможность задавать глубину логирования различных журналов. Приложение К «Библиотеки»: – обновлены названия библиотек; – в **PsLog** добавлен новый ФБ (AddLogTsEntry) для записи сообщений в лог с пользовательской меткой времени; – в **PsPlcInfo** добавлены: 1) ФБ **FirmwareSHA256** и **FirmwareHash** для реализации подсчета контрольной суммы СПО по алгоритму SHA-256 и MD5 соответственно, 2) ФБ **Get FileHashMD5** и **GetFileHashSHA256** для вычисления хэш-функции файла по алгоритму MD5 и SHA256 соответственно, 3) Функции **GetNetIntVal** и **GetNetStringVal** для получения информации о скорости соединения Ethernet; – в **PsRedundancy** новая функция Synchronize4, взамен устаревшей Synchronize3
2.17	*Раздел «Конфигурирование крейтов. Редактор шины*»: дополнена информация по шине RegulBusOS о реализации возможностей: – журналирования типа модуля при добавлении устройства в конфигурацию; – журналирования причины последнего перезапуска (ошибки) модулей/субмодулей/крейта, начиная с версии СПО 1.0.33.0. Удален раздел: «Особенности размещения модулей и субмодулей в крейте контроллера Regul R100». *Раздел «Конфигурирование крейта. Редактор шины*»: дополнена информация о возможности задания временных параметров внутренней шины крейта R050. *Раздел «Размещение модулей в крейте»:* создан подраздел *«Особенности размещения модулей и субмодулей в крейте контроллера Regul R050»*. *Раздел «Редактор модуля»:* на вкладку редактора модуля добавлен параметр «Серийный номер модуля».

Версия руководства пользователя	Описание изменения
	Добавлена поддержка модулей в среде разработки: – R500 CP 01 031; – R500 CP 04 041; – R500 AI 16 012; – R500 DI 32 012; – R500 DO 32 013. *Раздел «Задание параметров модулей коммуникационного процессора»:* добавлено описание параметров модулей коммуникационного процессора R500 CP 04 041 и R500 CP 01 031. *Раздел «Изменение политики соединения с ПЛК»*: дополнено описание по работе с сертификатами. Добавлены новые разделы: – раздел «Режимы работы контроллера»; – раздел «Создание резервной копии загруженного приложения без перезагрузки контроллера». Приложение Б *«Типы модулей центрального процессора»*: добавлены новые модули ЦП: – серия R050: CU 00 031(-W), CU 00 051(-W), CU 00 061(-W); – серия R500: CU 00 052(-W), CU 00 062(-W), CU 00 072(-W), CU 00 082(-W). Добавлено новое приложение: «Приложение К. Перечень модулей, поддерживаемых шиной RegulBus и/или RegulBus OS». В результате изменился порядок: «Приложение К»⇨«Приложение Л». Приложение Л «Библиотеки»: – всем EXTERNAL-функциям добавлен префикс компонента (например, TimeStampNow →**PsLog**TimeStampNow); – в **PsRedundancy_OS (PS_Redundancy_OS)** добавлена новая функция **IsApplicationEqual** для запроса признака одинаковых приложений на CPU_A и CPU_B; – для работы с шиной **RegulBus_OS** добавлено описание свойств **RingIsBroken** и **RingBrokenPlace** компонента **PsIoDrvRegulBus_OS.** Дополнительно по тексту внесены небольшие изменения с уточняющей информацией

АННОТАЦИЯ

Настоящий документ содержит сведения об установке и работе в среде разработки Astra.IDE. Данное программное обеспечение предназначено для конфигурирования и программирования промышленных логических контроллеров серии Regul RX00.

Astra.IDE позволяет осуществлять аппаратное конфигурирование контроллеров Regul RX00, создание и редактирование прикладного программного обеспечения, настройку резервирования, загрузку и выгрузку проектов, пошаговую отладку и онлайн-контроль прикладной программы, диагностику работы контроллера.

Данное руководство предназначено для эксплуатационного персонала и инженеров-проектировщиков АСУ ТП, которые должны:

– иметь, как минимум, среднее техническое образование;

– приступить к работе только после изучения данного руководства.

Обновление информации в Руководстве

Производитель ООО «РегЛаб» оставляет за собой право изменять информацию в настоящем Руководстве и обязуется публиковать более новые версии с внесенными изменениями. Обновленная версия Руководства доступна для скачивания на официальном сайте Производителя: https://reglab.ru/.

Для своевременного отслеживания выхода новой версии Руководства рекомендуется оформить подписку на обновление документа. Для этого необходимо на сайте Производителя: https://reglab.ru/ кликнуть на кнопку «Подписаться на обновления» и оставить свои контактные данные.

В руководстве присутствуют знаки с предупреждающей и поясняющей информацией. Каждый знак обозначает следующее:

ПРЕДУПРЕЖДАЮЩИЕ ЗНАКИ

	**ВНИМАНИЕ!** Здесь следует обратить внимание на способы и приемы, которые необходимо в точности выполнять во избежание ошибок при эксплуатации или настройке.

ИНФОРМАЦИОННЫЕ ЗНАКИ

	**ИНФОРМАЦИЯ** Здесь следует обратить внимание на важную информацию

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Основная цель настоящего документа – дать пользователю базовые знания о том, как настроить контроллер серии Regul RX00 с помощью программного обеспечения Astra.IDE. Подробное описание программирования, настройки каналов передачи данных, построения резервированной системы приведено в отдельных документах.

Работа в среде Astra.IDE – это построение аппаратной конфигурации контроллера, настройка подключения к компьютеру и к «полевым» устройствам, программирование контроллера, установка значений различных параметров. Все эти операции описаны в соответствующих разделах и/или отдельных документах. Каждый из разделов можно читать независимо от других и для решения текущей задачи. Вместе с тем, для правильного понимания среды разработки Astra.IDE рекомендуется ознакомиться с основными понятиями, с описанием интерфейса и с обзором среды разработки.

Термины и определения

Программируемый логический контроллер (ПЛК) — микропроцессорное устройство в промышленном исполнении, используемое для сбора, преобразования, обработки, хранения информации и выработки команд управления, имеющее конечное количество входов и выходов, подключенных к ним датчиков, ключей, исполнительных механизмов к объекту управления, и предназначенное для работы в режимах реального времени.

Интегрированная среда разработки (IDE) — комплекс программных средств, используемый техническими специалистами для разработки прикладного программного обеспечения (ПО).

Astra.IDE - среда разработки с набором компонент для настройки/программирования контроллеров серии Regul RX00, выпускается компанией «РегЛаб».

Astra.IDE позволяет работать в редакторах стандарта МЭК 61131-3:

- IL (Instruction List) — список инструкций,
- ST (Structured Text) — структурированный текст,
- LD (Ladder Diagram) — релейно-контактная логика,
- FBD (Function Block Diagram) — функциональные блоковые диаграммы,
- SFC (Sequential Function Chart) — последовательные функциональные диаграммы.

Крейт (Crate) – обособленная сборка из нескольких модулей, объединенных в общем корпусе или на DIN-рейке, с независимым электропитанием.

CPU (Central processing unit) – центральное процессорное устройство (ЦПУ), также модуль центрального процессора (ЦП) программируемого логического контроллера REGUL RX00. В

резервированном контроллере используются два модуля центрального процессора, называемые ЦП-партнерами (**CPU A и CPU B**).

Проект (Project) – совокупность программного кода и настроек устройств, необходимая для выполнения контроллером своих функций.

POU (Program Organization Unit) – компонент организации программы. В большинстве случаев под этим термином понимается пользовательская программа и функциональный блок.

Устройство (Device) - специализированное аппаратное средство, на котором должна запускаться ПЛК-программа (приложение).

Приложение (Application) – это набор программных объектов, необходимых для запуска конкретного экземпляра пользовательской программы на конкретном устройстве.

Задача (Task) – часть приложения, выполняющая блок подпрограмм в отдельном цикле контроллера. Для каждой задачи можно определить контроль времени выполнения.

Система исполнения (Runtime System) – это часть системного программного обеспечения контроллера, предназначенное для выполнения на контроллере пользовательской программы. Устанавливается в контроллер в процессе его изготовления.

Плагины (Plug-ins) – модули, подключаемые к среде разработки Astra.IDE, обеспечивающие настройку и конфигурирование контроллеров серии Regul RX00. Каждый плагин имеет имя, версию и ограничения версии.

Профиль (Profile) – набор плагинов конкретных версий, которые будут подгружены к среде разработки в момент запуска.

Перечень рекомендуемых документов

Для получения дополнительной подробной информации по настройке контроллеров серии Regul RX00 в среде разработки Astra.IDE рекомендуется ознакомиться со следующими документами (доступны на сайте https://reglab.ru/):

- Regul R600. Системное руководство, Regul R500. Системное руководство, Regul R400. Системное руководство, Regul R200. Системное руководство;
- Конфигурирование резервированной системы на контроллерах серии REGUL RX00. Руководство пользователя;
- Настройка обмена данными по протоколу Modbus на контроллерах серии REGUL RX00. Руководство пользователя;
- Настройка обмена данными по протоколу IEC-104 на контроллерах серии REGUL RX00. Руководство пользователя;

- Настройка обмена данными по протоколу HART на контроллерах серии REGUL RX00. Руководство пользователя;
- Настройка и работа REGUL OPC DA Server. Руководство пользователя.

УСТАНОВКА ПРОГРАММЫ

Минимальные системные требования

Для установки интегрированной среды разработки Astra.IDE требуется:

- минимально допустимая версия операционной системы:
  - Windows 10 PRO;
  - Windows Server 2016;
- 8 Гб оперативной памяти;
- 25 Гб свободного места на диске;
- процессор Intel core 2 Quad 2.4 GHz.

Процесс установки

Программное обеспечение Astra.IDE в общем случае состоит из среды разработки и пакетов обновления, которые распространяются по мере внесений улучшений и добавление нового функционала. При этом у пакетов обновления есть требования к минимальной версии среды разработки, на которую они могут быть установлены.

В этом подразделе описан процесс установки среды разработки. Описание установки пакета обновлений, приведено в подразделе «Обновление ПО контроллера».

Файл для установки среды разработки и файл пакета обновлений доступны на сайте предприятия-изготовителя по адресу: https://reglab.ru/. Для установки среды разработки, запустите файл **Astra.IDE 64 xxxx.exe**, где xxxx – номер версии.

Для работы Astra.IDE могут потребоваться дополнительные элементы. Если они отсутствуют на компьютере, программа-установщик предлагает их установить (Рисунок 1). Нажмите кнопку ***Установить***.

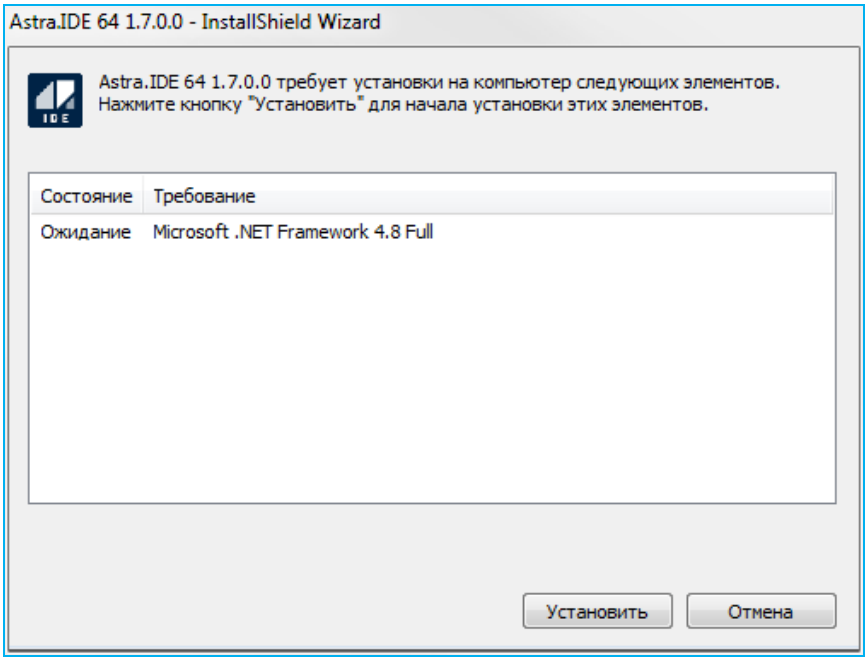


Рисунок 1 – Запрос установки дополнительных элементов, необходимых для работы Astra.IDE

По окончании установки дополнительных элементов откроется основное диалоговое окно программы-установщика (Рисунок 2). Нажмите кнопку ***Далее***.

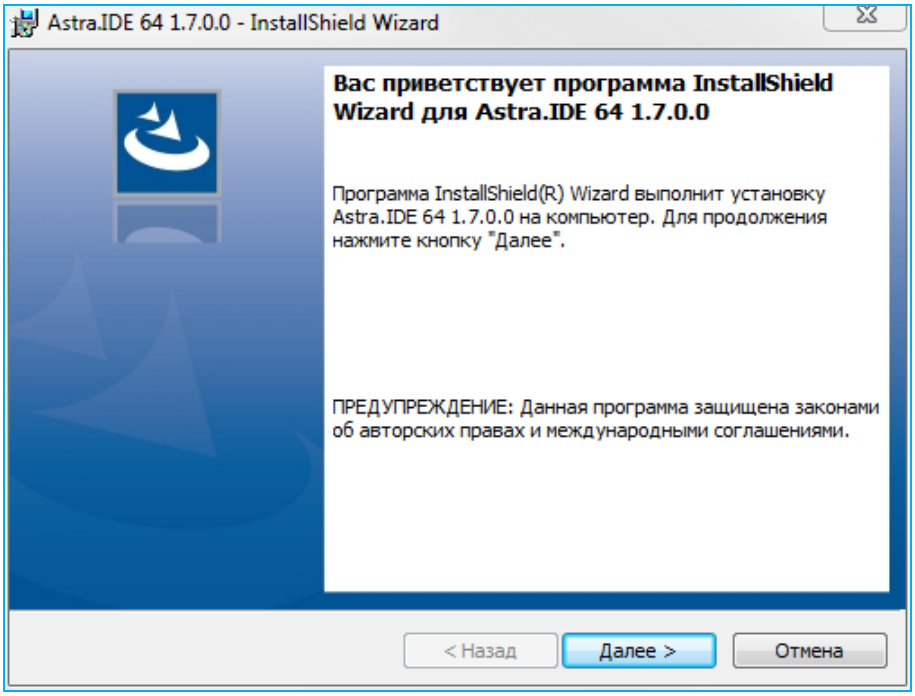


Рисунок 2 – Диалоговое окно программы-установщика Astra.IDE

Откроется окно лицензионного соглашения (Рисунок 3).

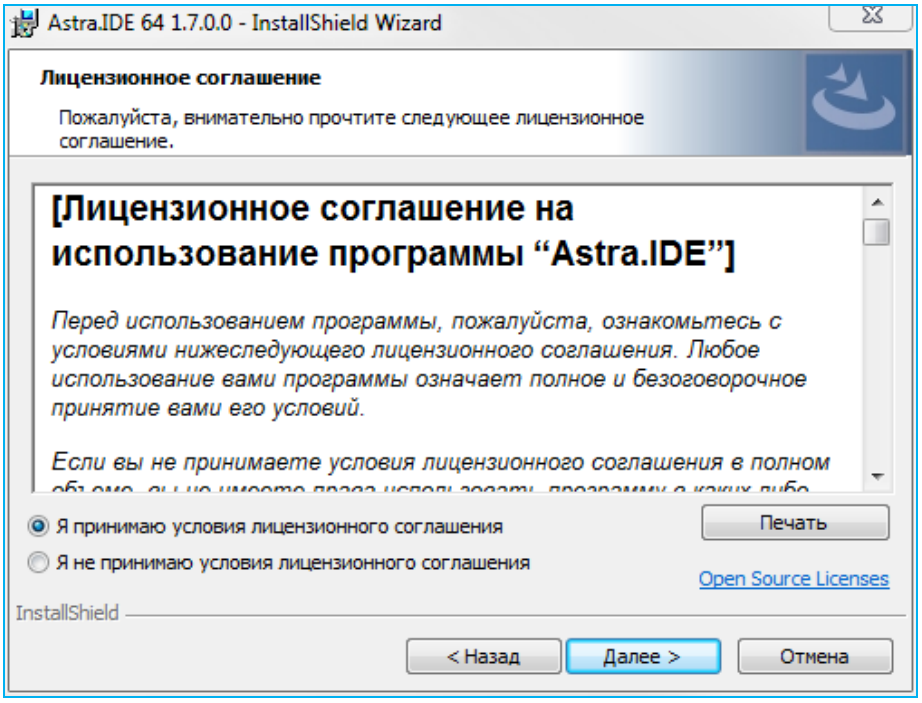


Рисунок 3 – Диалоговое окно с условиями лицензионного соглашения

Прочитайте лицензионное соглашение на использование программы. Для получения печатной копии соглашения предусмотрена кнопка ***Печать***, нажав которую вы перейдете к настройкам принтера и параметров печати, и сможете распечатать документ. Для продолжения установки поставьте переключатель в поле **Я принимаю условия лицензионного соглашения**. Нажмите кнопку ***Далее***.

Откроется окно для выбора местоположения программы (Рисунок 4). По умолчанию программа установки создает папку **AstraRegul** в каталоге **Program Files** (если операционная система Windows 32-bit) или в каталоге **Program Files (x86)** (если операционная система Windows 64-bit). Если требуется установить программу в другое место, нажмите кнопку ***Изменить…*** и в открывшемся окне укажите, куда следует установить программу.

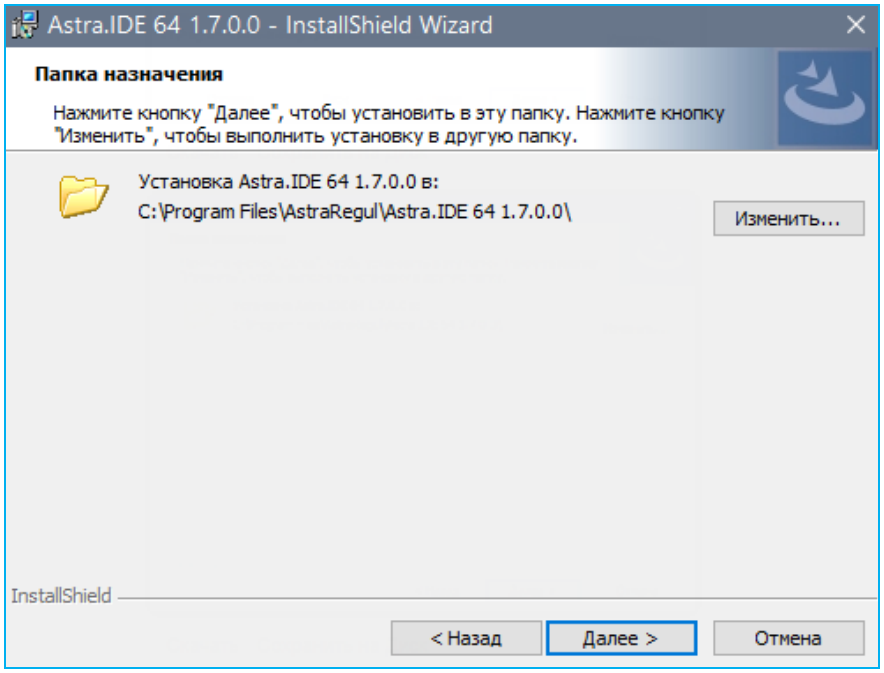


Рисунок 4 – Диалоговое окно выбора папки, куда будет установлена программа

В окне **Папки назначения** нажмите кнопку ***Далее***. Откроется диалоговое окно **Вид установки** (Рисунок 5). Выберите вид установки: **Полная** или **Выборочная.** Опытный пользователь может воспользоваться выборочной установкой компонентов программы.

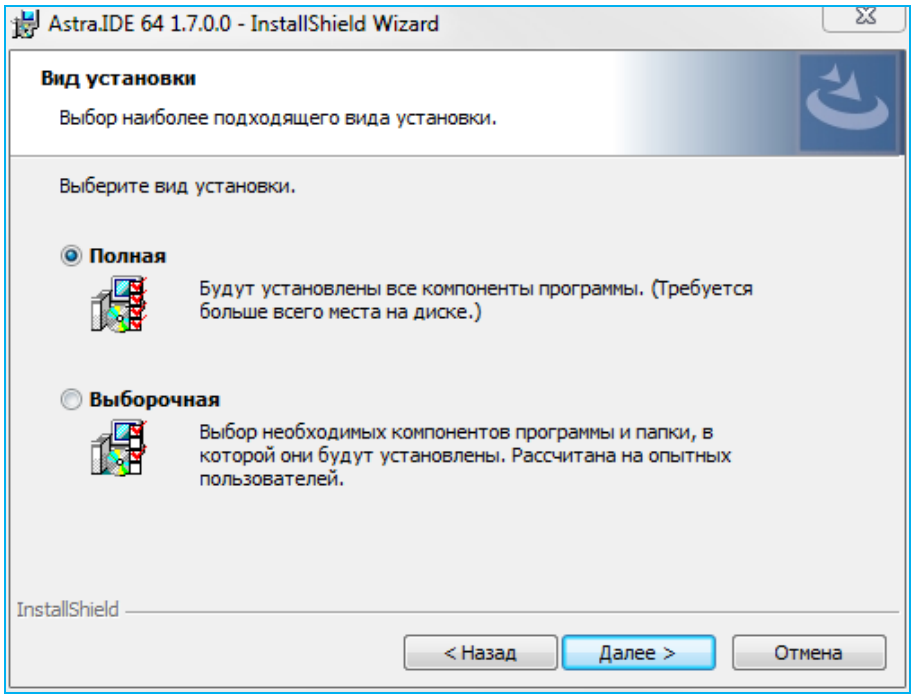


Рисунок 5 – Диалоговое окно выбора вида установки компонентов

При установке переключателя в поле **Выборочная** всплывет окно **Выборочная установка** (Рисунок 6).

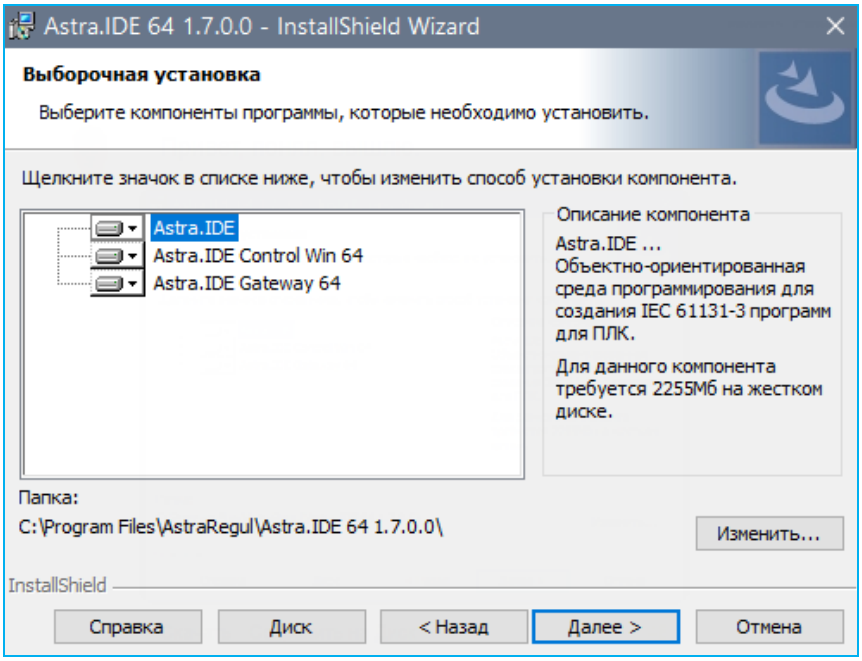


Рисунок 6– Диалоговое окно выборочной установки компонентов

Нажмите кнопку **Справка** и откроется окно **Советы по выборочной установке**, ознакомьтесь со способами изменения установки компонентов (Рисунок 7)

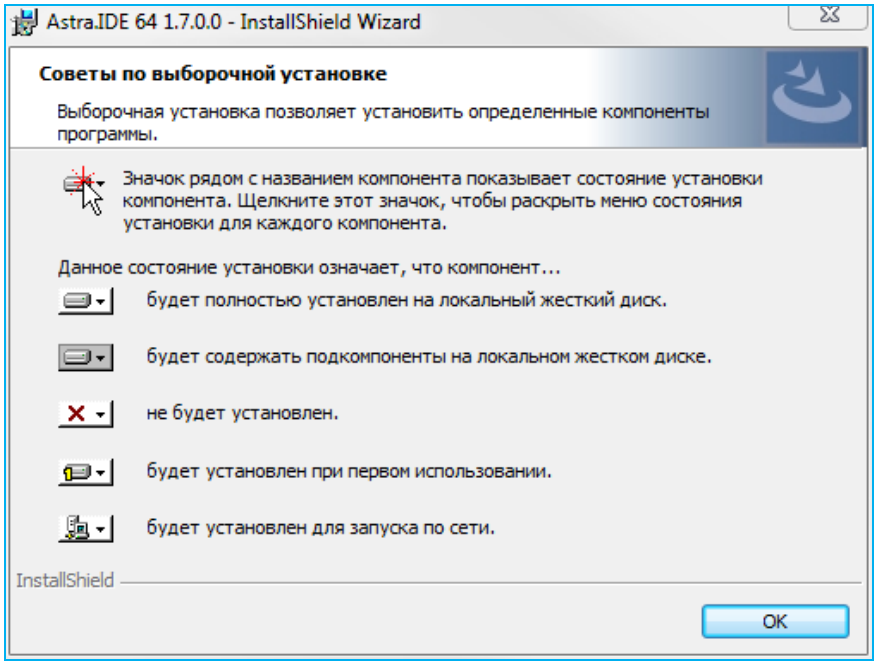


Рисунок 7– Диалоговое окно справки

Компоненты, не выбранные в процессы установки Astra.IDE, в дальнейшем могут быть установлены с помощью стандартных механизмов установки/удаления программ Windows.

После выбора компонентов для установки нажмите кнопку ***Далее***. Откроется диалоговое окно готовности установки программы (Рисунок 8).

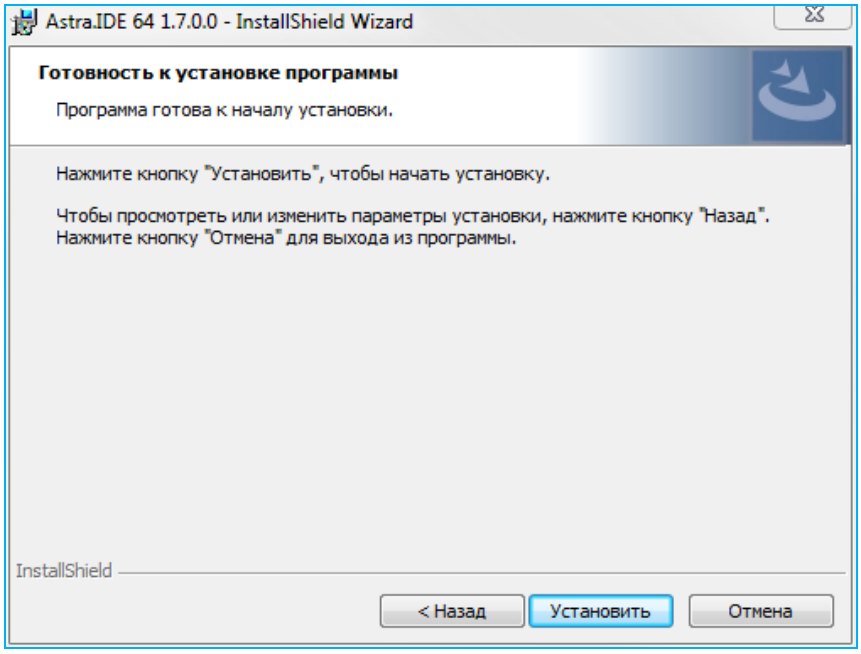


Рисунок 8 – Диалоговое окно готовности установки программы

Проверьте все настройки, сделанные на предыдущих этапах. Если требуется их изменить, воспользуйтесь кнопкой ***Назад*** и перейдите на нужный шаг. Если все настройки указаны верно, нажмите кнопку ***Установить.***

Начнется установка программы Astra.IDE с полным или выбранным набором компонентов, что может занять некоторое время (Рисунок 9).

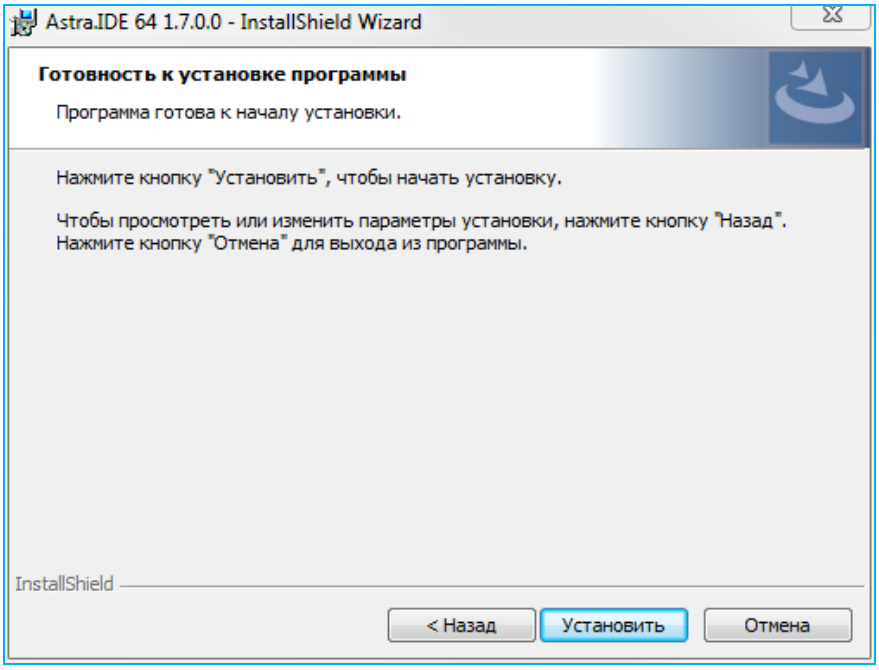


Рисунок 9 – Диалоговое окно установки программы

Дождитесь окончания процесса и появления оповещения об успешном окончании установки. Нажмите кнопку ***Готово***. По окончании установки программы автоматически будет создан ярлык на рабочем столе и в меню **Пуск**.

ОБЗОР СРЕДЫ РАЗРАБОТКИ

Описание интерфейса

Общие сведения

В среде разработки Astra.IDE интерфейс пользователя включает в себя: основное меню, панель инструментов, рабочую область с различными окнами, строку состояния (Рисунок 10).

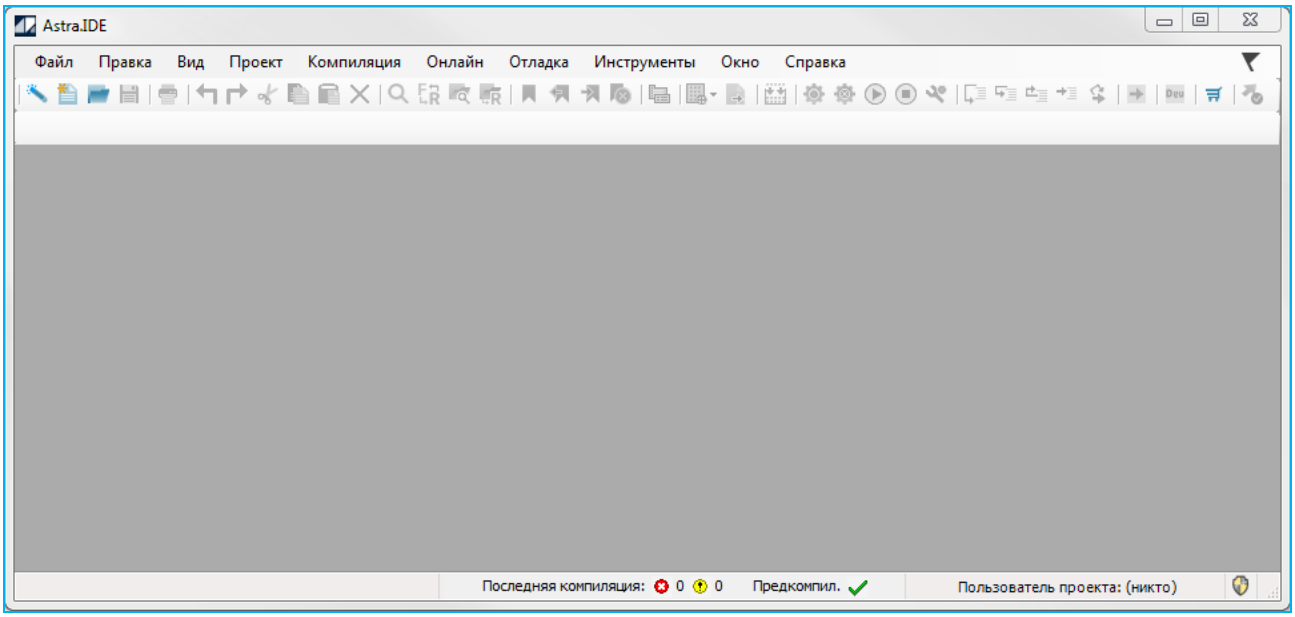


Рисунок 10 – Основное окно программы

Основное меню и панель инструментов можно настраивать «под себя» в настройках программы. Здесь можно добавлять и удалять пункты меню и команды, редактировать выпадающее меню, добавлять панели инструментов и отдельные команды, менять порядок элементов, назначать комбинации клавиш для различных операций. Все эти настройки сохраняются в специальном файле.

Так, например, если на основной панели отсутствует пиктограмма **Мастера конфигурации Regul** (), ее можно добавить следующим образом:

– выберите в основном меню **Инструменты** ⇨ **Настройка…** Откроется окно **Настройка**;

– перейдите на вкладку **Панель инструментов** и рядом с пунктом **Standard** нажмите кнопку , в открывшемся списке поставьте курсор на пункт **Новый проект…** Станет активной кнопка ***Добавить команду…*** (Рисунок 11);

– нажмите кнопку ***Добавить команду...*** Откроется окно **Добавление команды**;

– в левой части окна в блоке **Категории** выберите *Устройства*. В правой части окна в блоке **Команды** отобразятся все команды этой категории. Выберите команду **Мастер**

конфигурации Regul (Рисунок 12) и нажмите кнопку ***ОК***. В окне **Настройка** нажмите кнопку ***ОК*** и закройте окно. На панели инструментов перед пиктограммой  (Новый проект) появится пиктограмма  (Мастер конфигурации Regul).

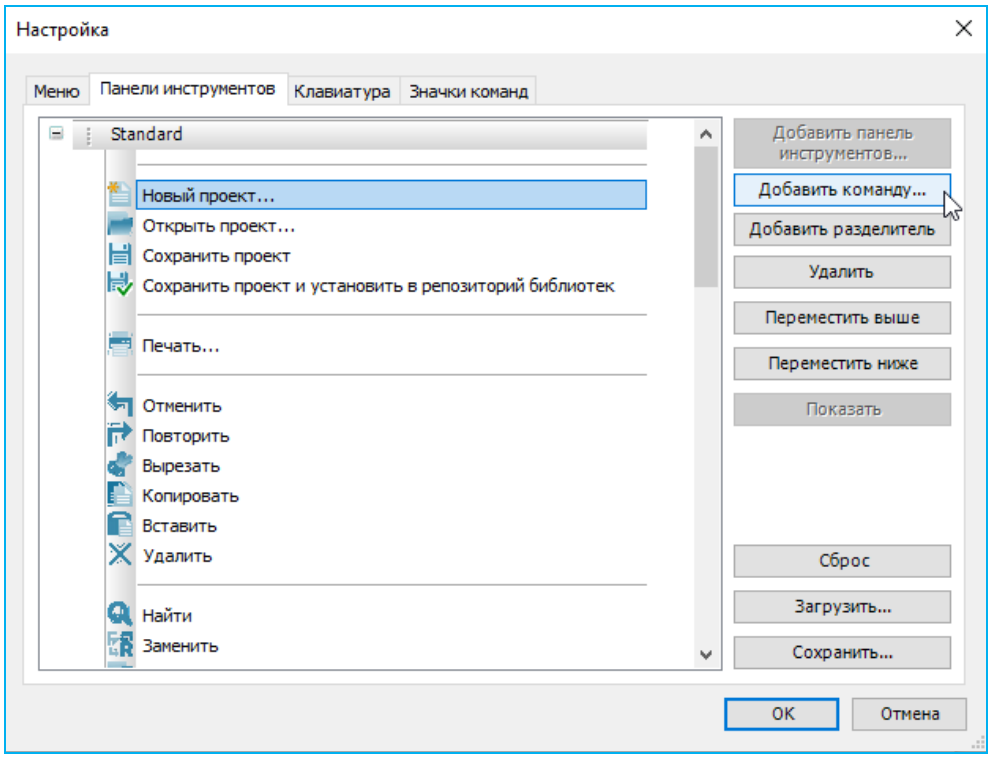


Рисунок 11 – Окно настройки основного меню и панели инструментов

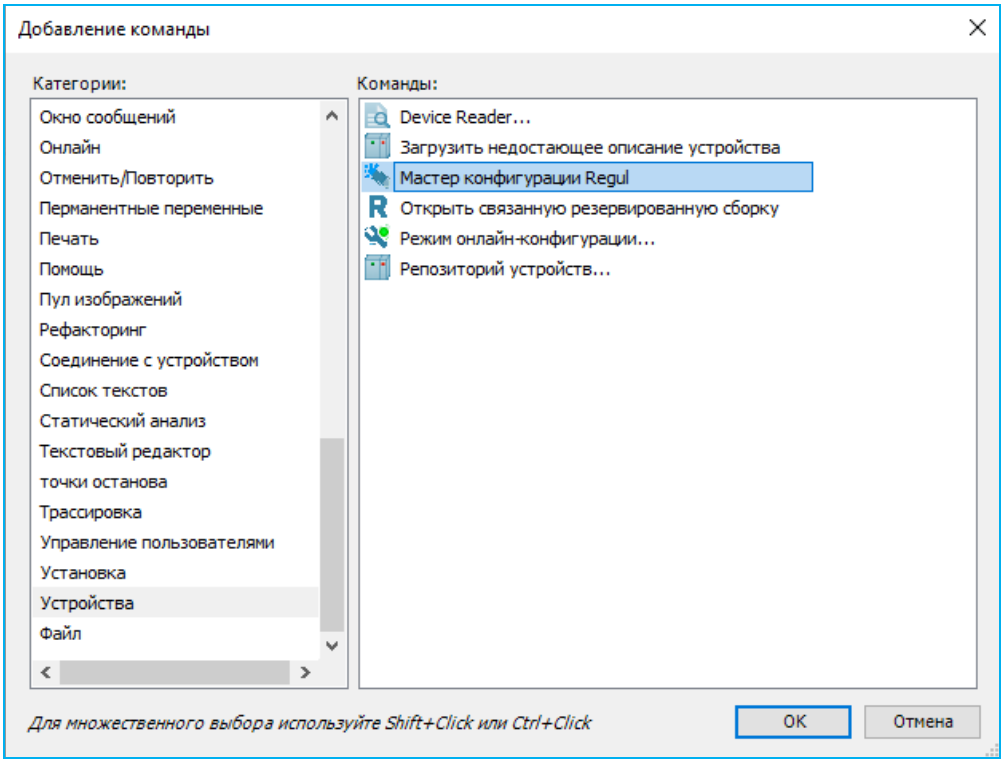


Рисунок 12 – Добавление команды на панель инструментов

Для некоторых команд по умолчанию заданы «горячие» клавиши, например, для компиляции – клавиша ***F11*** (Рисунок 13), открыть новый проект – ***Ctrl+N***.

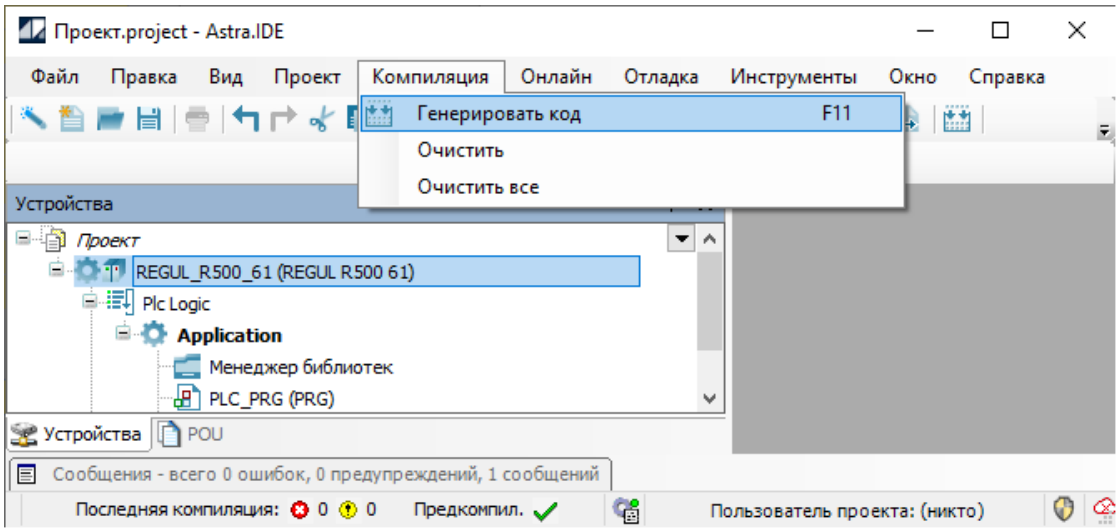


Рисунок 13 – Пример команды с назначеной «горячей» клавишей

Кроме основного меню существует контекстное меню, вызывается правой клавишей мыши (Рисунок 14). Состав контекстного меню меняется в зависимости от выбранного объекта.

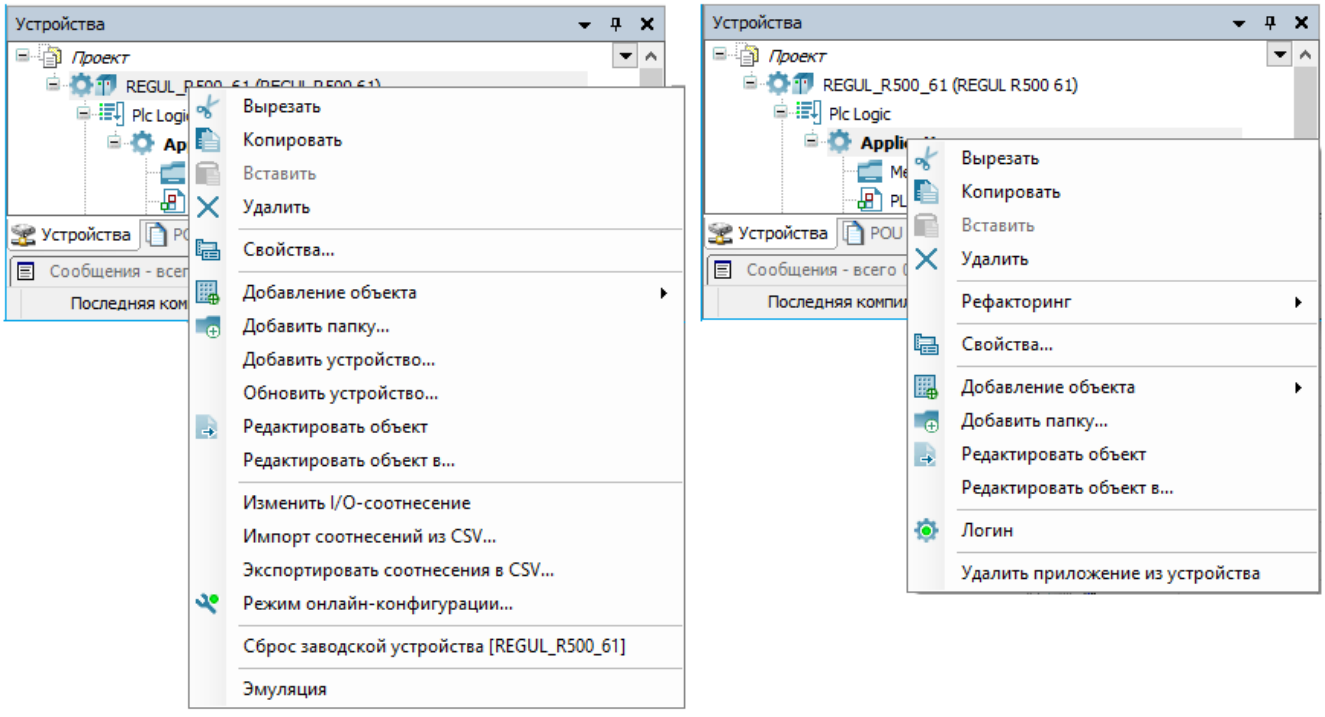


Рисунок 14 – Примеры контекстного меню

В рабочей области работа ведется в окнах и вкладках. (Рисунок 15).

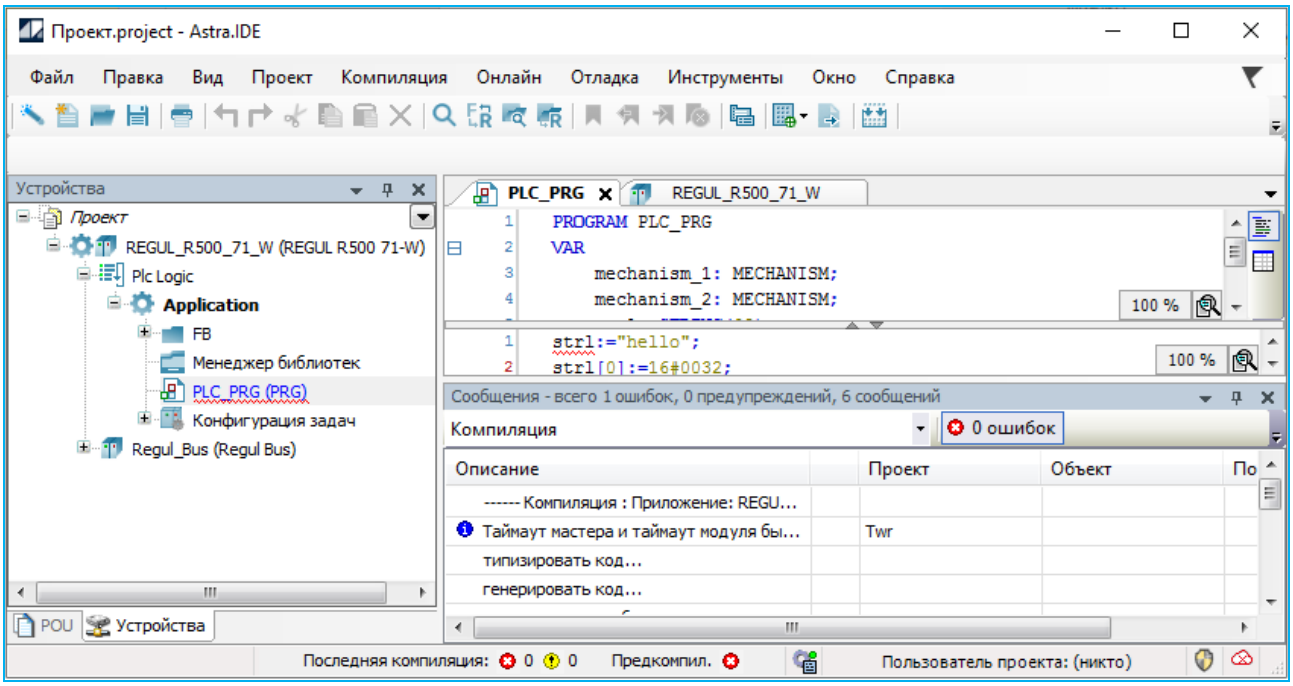


Рисунок 15 – Окна и вкладки в рабочей области

В документе могут содержаться изображения вкладок и окон в двух взаимозаменяемых видах (Рисунок 16).

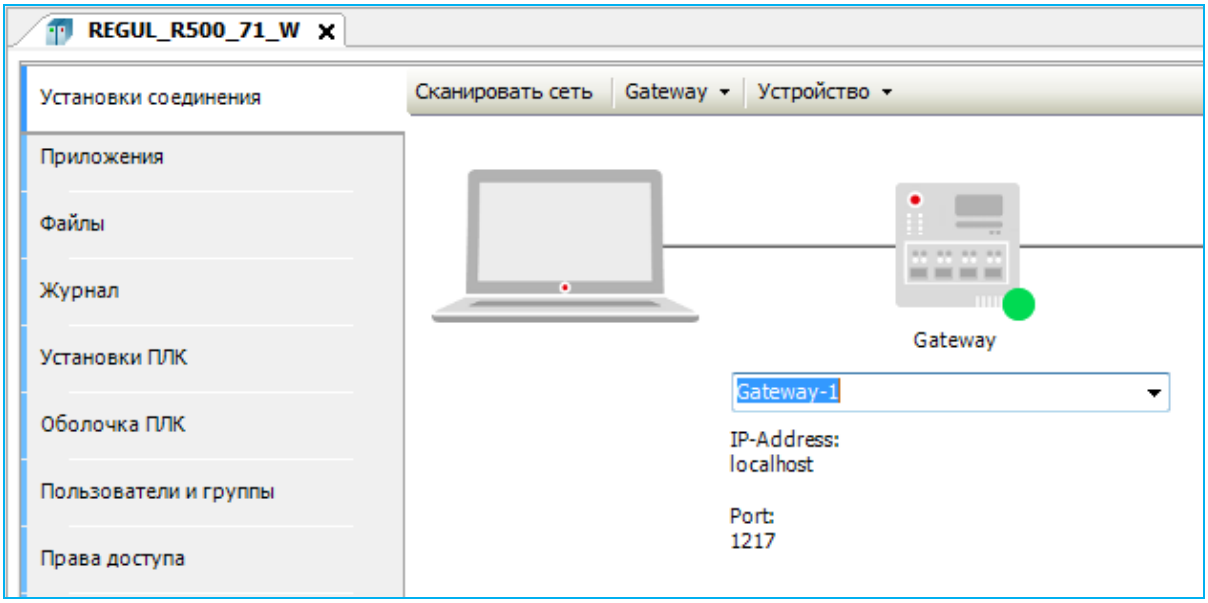


Рисунок 16 – Пример вертикального отображения вкладок

Чтобы изменить отображение, в главном меню программы Astra.IDE выберите пункт **Инструменты** ⇨ **Опции** и в открывшемся окне пункт **Редактор устройств**. Установите флажок в поле **Использовать горизонтальные вкладки** (Рисунок 17).

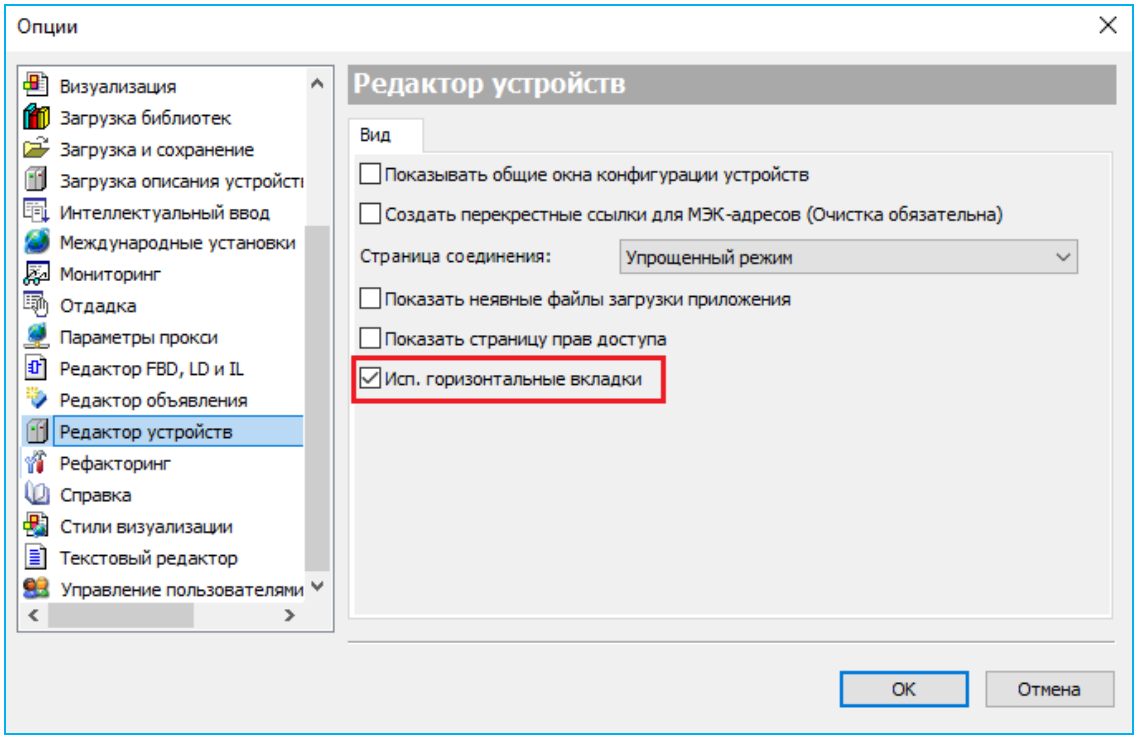


Рисунок 17 – Настройки редактора устройств

В примере (Рисунок 15) окно **Устройства** и вкладки прикреплены к границам основного окна, «вписаны» в рабочую область. Двойным щелчком левой кнопки мыши по заголовку окна/вкладки оно/она превращается в отдельное окно, которое можно перемещать по рабочей области и вне ее, менять размеры, свернуть, закрыть (Рисунок 18).

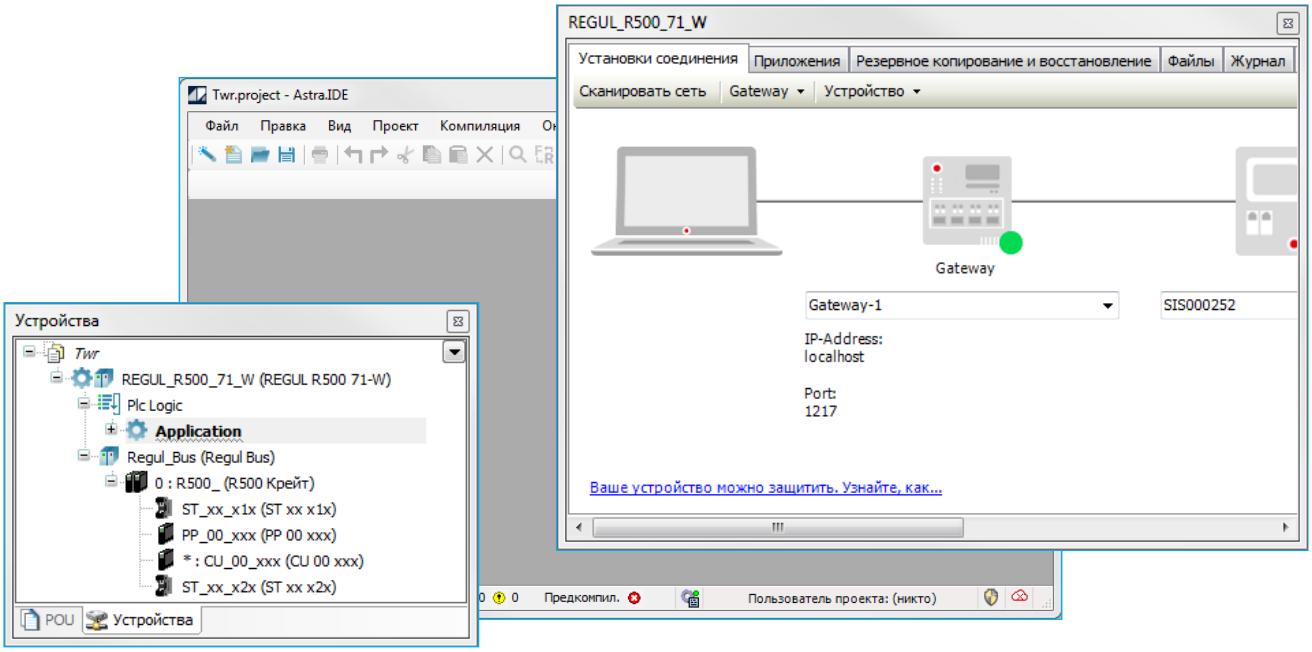


Рисунок 18 – Окна и вкладки в виде отдельных окон

Также двойным щелчком мыши по заголовку окна оно возвращается в исходное состояние. Случайно закрытое или просто отсутствующее окно можно найти, выбрав его в основном меню в пункте **Вид** (Рисунок 19).

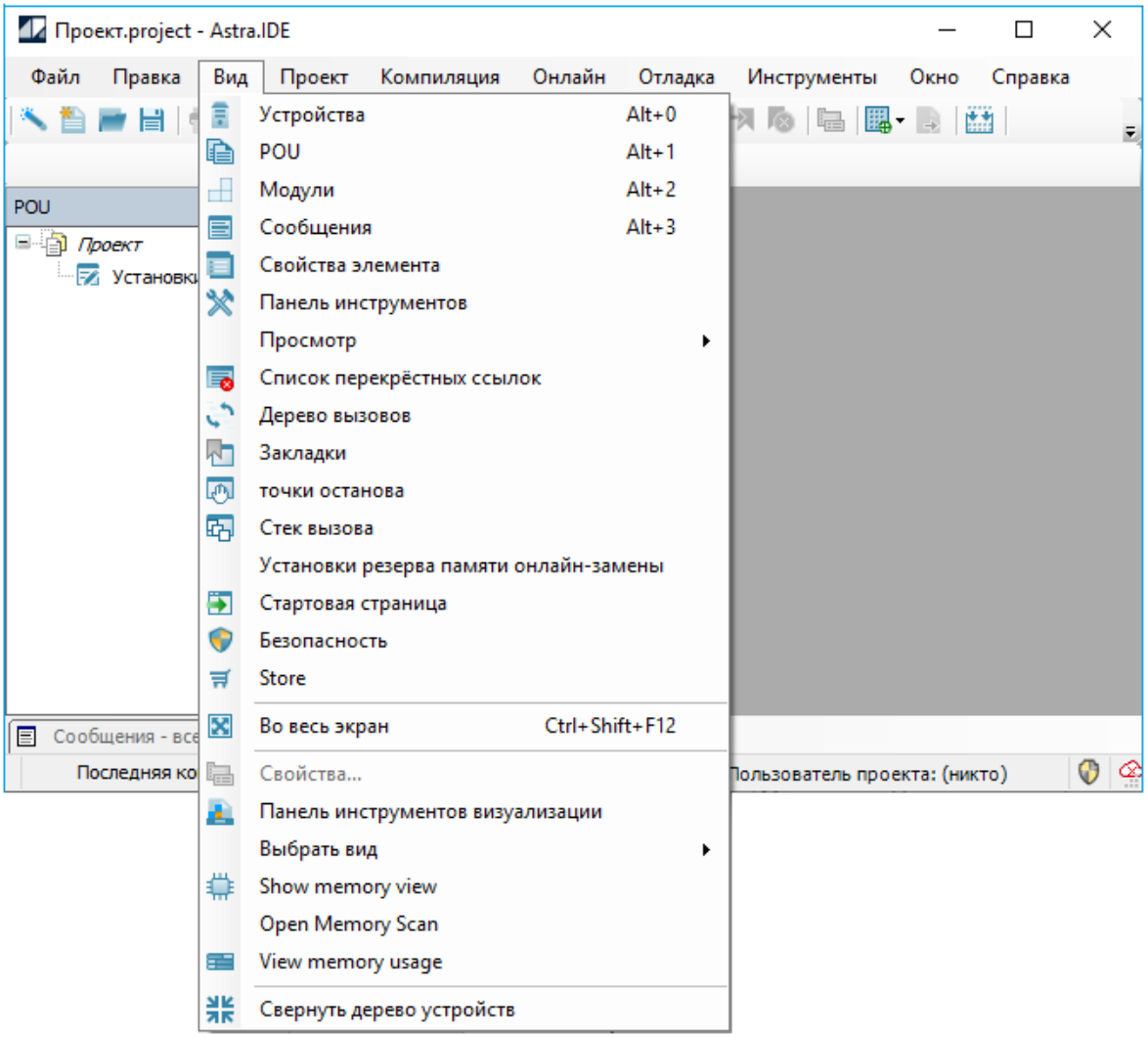


Рисунок 19 – Выбор нужного окна в пункте меню «Вид»

Существуют окна, которые открываются при просмотре или редактировании конкретных объектов проекта в соответствующем редакторе. Их нельзя «спрятать» или «отсоединить» от обрамляющего окна. Доступ к ним осуществляется через пункт основного меню **Окно**.

Окна, которые используются в среде разработки:

- окно **панели инструментов** содержит инструменты конкретных редакторов;
- окно **POU** служит для организации программных компонентов проекта в виде дерева;
- окно **Устройства** служит для организации ресурсных объектов проекта в виде дерева;
- окно **редактора** используется для создания конкретного объекта в соответствующем редакторе, например, ST-редактор, CFC-редактор, редактор задач, редактор устройств;
- окно **сообщений**: в этом окне отображаются сообщения компиляции, предкомпиляции, загрузки и так далее;
- окна **наблюдения** и **онлайн просмотр редакторов**: обеспечивают просмотр POU или списков наблюдаемых выражений, заданных пользователем.

В нижней части основного окна программы находится **строка состояния** (Рисунок 20).

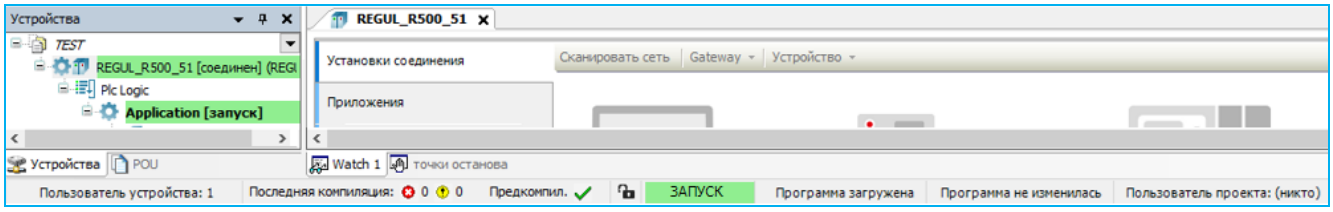


Рисунок 20 – Пример строки состояния

Строка состояния содержит:

– сообщения последней компиляции;

– информацию о текущем пользователе;

– текущую позицию курсора и режим редактирования (в момент, когда вы работаете в окне редактора);

– режим работы контроллера (символ):

  - (Отладка (Debug)): режим работы без ограничений, позволяет выполнять все онлайн-команды;
  - (Заблокировано (Debug)): текущее состояние отладки приложения заблокировано. Дополнительно нельзя устанавливать точки останова и принудительно задавать переменные. Запись переменных по-прежнему возможна;
  - (Оперативный (Operational)): ничего нельзя изменить. Точки останова не могут быть установлены. Запись переменных по-прежнему возможна.

– текущее состояние программы (в режиме онлайн):

  - РАБОТА (RUN) – программа запущена,
  - СТОП (STOP) – программа остановлена,
  - ТОЧКА ОСТАНОВА – программа остановлена в точке останова,
  - Программа загружена (Program loaded) – программа загружена на контроллер,
  - Программа не изменилась (Program unchanged) – программа в контроллере идентична программе в системе программирования,
  - Программа изменена (онлайн-изменение) (Program modified (Online Change)) – программа в контроллере отличается от программы в системе программирования, необходимо онлайн-изменение,
  - Программа изменена (полная загрузка) (Program modified (Full download)) – программа в контроллере отличается от программы в системе программирования, требуется полная загрузка;
  - Программа загружена - ИСКЛЮЧЕНИЕ (Program loaded - EXCEPTION) – программа загружена, но в процессе работы возникла исключительная ситуация,

- Программа загружена – ЭМУЛЯЦИЯ (Program loaded - SIMULATION) – программа загружена в режиме эмуляции.

Добавление объектов

Для добавления объектов чаще всего используется контекстное меню, вызываемое правой клавишей мыши, а в нем соответствующие пункты, например, **Добавить объект** (Рисунок 21) или **Добавить устройство…** (Рисунок 22).

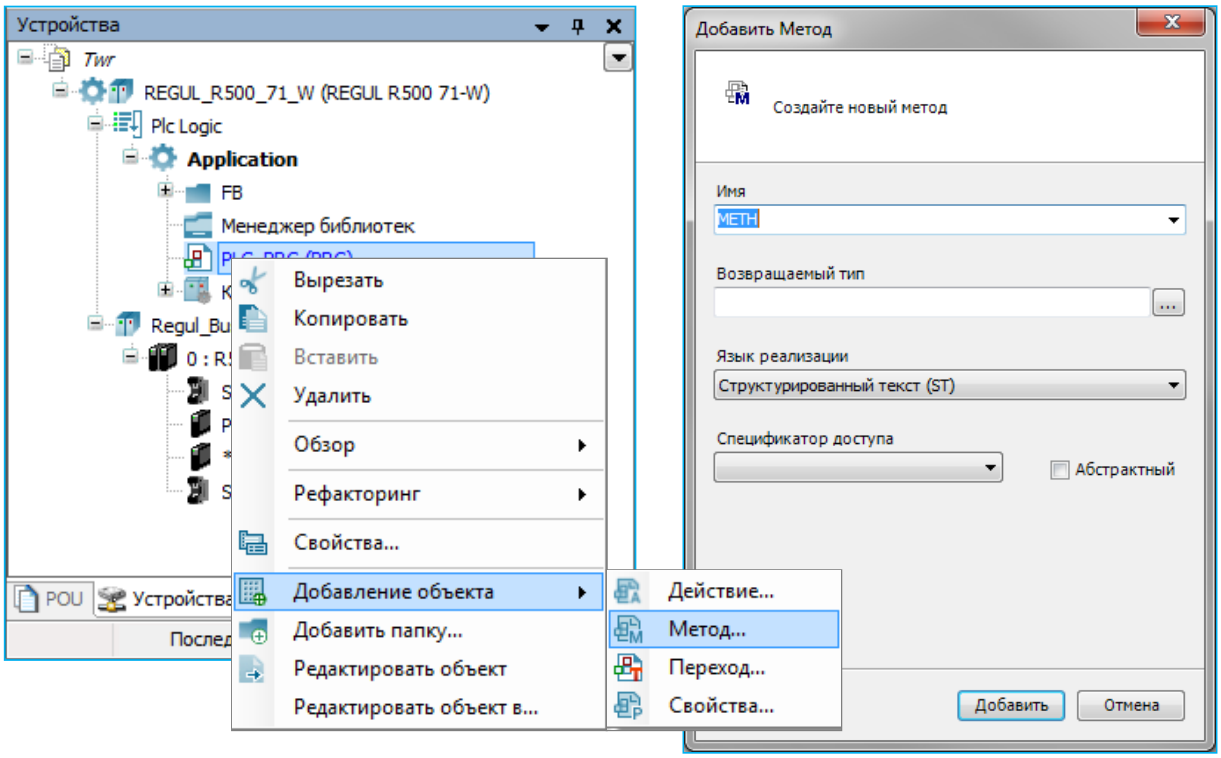


Рисунок 21 – Добавление объекта

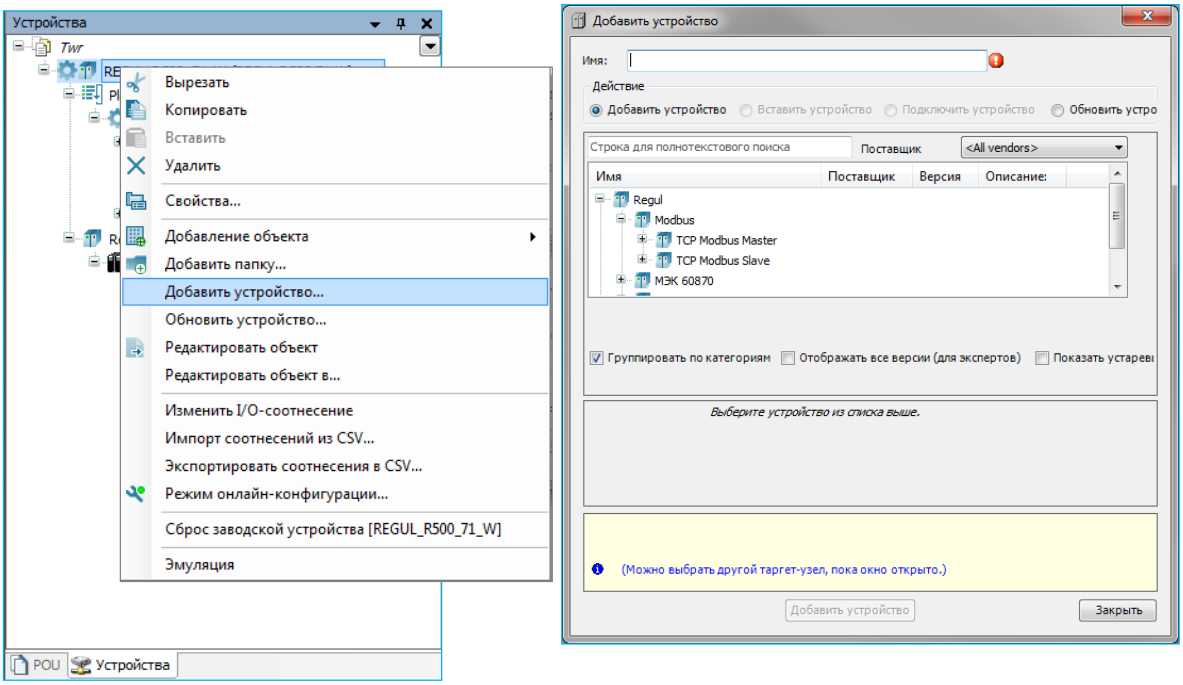


Рисунок 22 – Добавление устройства

Переименование объектов

Для переименования объекта выберите его и нажмите на его название один раз. Строка с названием перейдет в режим редактирования, где текст можно изменить.

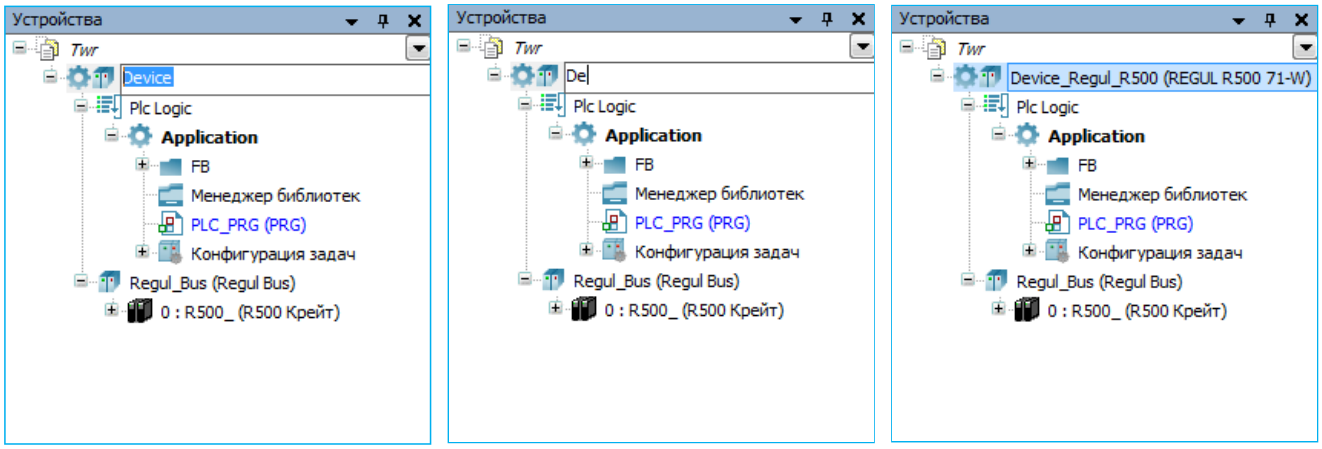


Рисунок 23 – Переименование устройства

Таким образом переименовать можно не только контроллер, но и другие устройства (крейты, модули, так далее), приложение, а также объекты POU (программные компоненты).

Удаление объектов

Для удаления объектов также используется контекстное меню, вызываемое правой клавишей мыши, а в нем соответственно пункт **Удалить**. Или клавиша ***Delete*** на клавиатуре.

⚠	**ВНИМАНИЕ!** Программа не запрашивает подтверждения на удаление, поэтому можно случайно удалить нужное устройство или большой фрагмент работы

Надо отметить, что пока изменения не сохранены в проекте, их можно отменить, в том числе вернуть удаленные объекты. То есть если вы еще не выбрали в основном меню **Файл** ⇨ **Сохранить проект** или не нажали кнопку , а в заголовке окна присутствует значок * (имеются несохраненные изменения), то с помощью кнопки на панели инструментов можно отменить некоторое количество действий. На сколько шагов назад можно вернуться зависит от последнего сохранения проекта. Например, с момента последнего сохранения проекта пользователь выполнил три действия, значит с помощью кнопки можно отменить три действия.

Кроме того, проверьте, включено ли автосохранение и с каким интервалом. Все операции в проекте, выполненные до автоматического сохранения проекта, отменить уже нельзя. Если выбрано автосохранение с интервалом в 5 минут, например, то можно отменить действия, произведенные в течение последних пяти минут, но невозможно отменить операции, выполненные 10 или 15 минут назад. Для включения/отключения функции автосохранения и изменения интервала сохранения выберите в основном меню **Инструменты** ⇨ **Опции** и в открывшемся окне пункт **Загрузка и сохранение** (Рисунок 24).

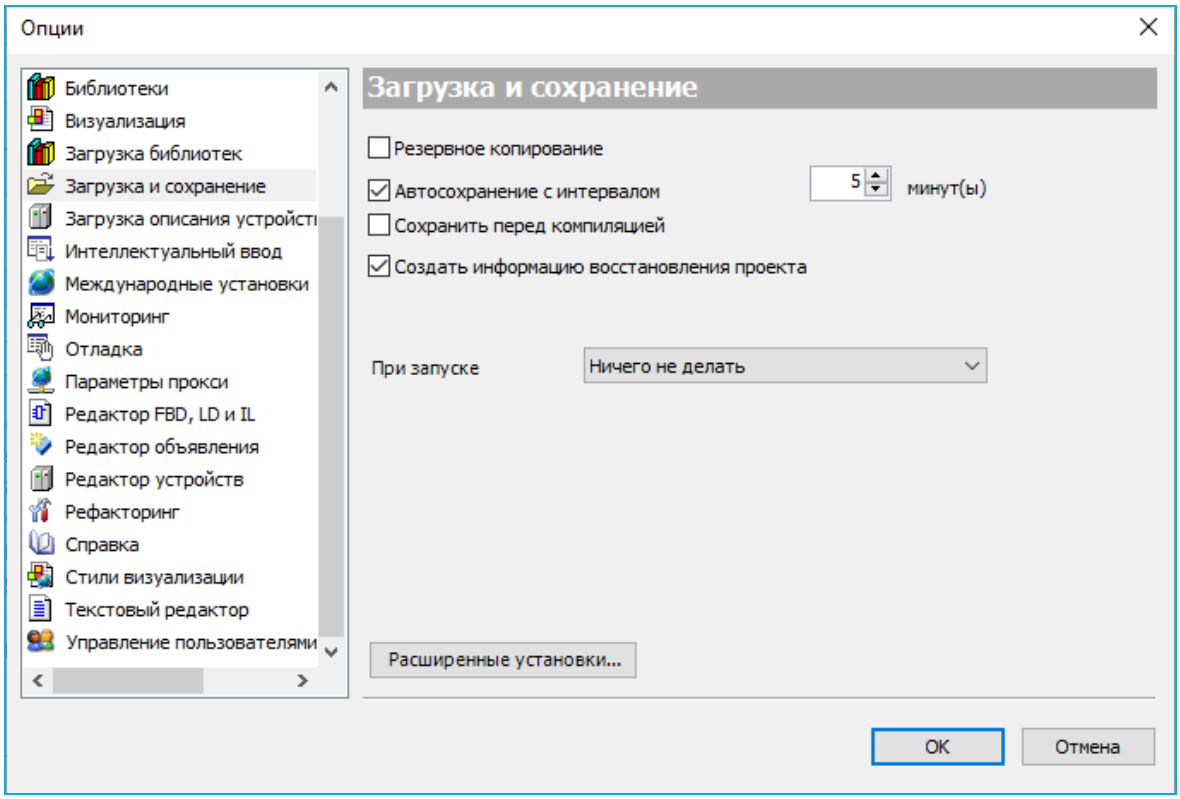


Рисунок 24 – Настройка автосохранения

Обновление устройств

Когда требуется заменить в проекте контроллер или крейт, нет необходимости удалять существующее устройство и добавлять устройство другого типа, следует использовать команду контекстного меню **Обновить устройство…** Откроется окно, аналогичное окну **Добавить устройство…**, где, с помощью кнопки + раскрывая список устройств, выберите актуальное устройство.

После обновления устройства его можно переименовать в дереве устройств (Рисунок 25).

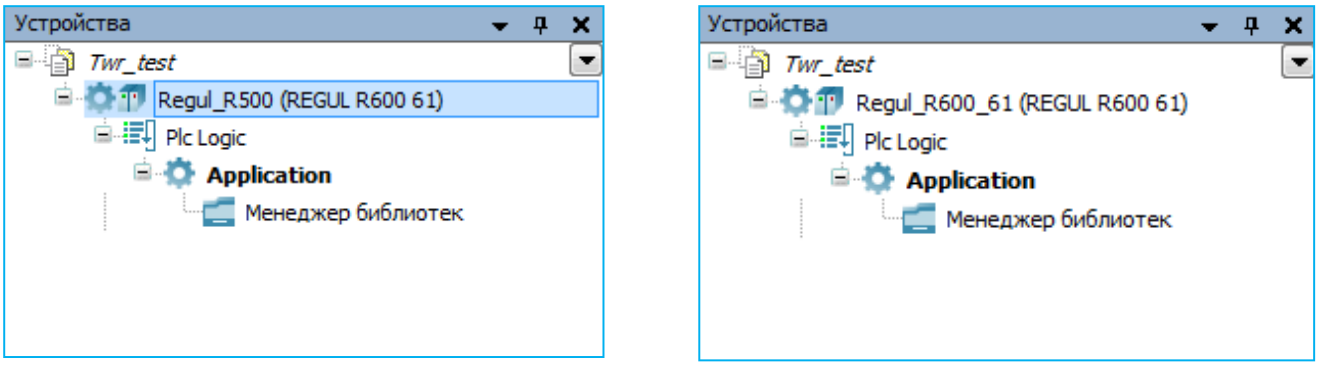


Рисунок 25 – Переименование контроллера после обновления устройства

Заполнение форм, установка параметров

Для заполнения форм используются привычные инструменты, такие как установка или снятие флажка, выбор значения в раскрывающемся списке, ручной ввод значения в поле (Рисунок 26).

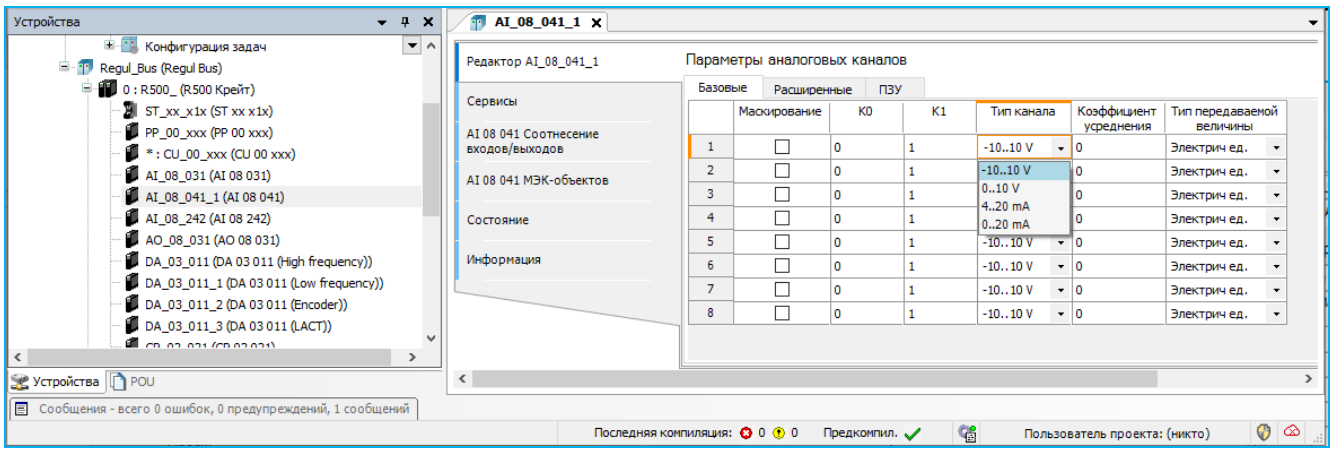


Рисунок 26 – Пример окна установки параметров

При ручном вводе значения в поле для подтверждения действия нажмите клавишу ***Enter*** или поместите курсор мыши в произвольном месте формы. Новое значение будет принято программой. Отменить ошибочно введенное значение можно, нажав клавишу ***Esc***. В некоторых формах, например, при настройке параметров шины, обязательно нужно нажать клавишу ***Enter*** после изменения значения в поле, иначе оно не будет установлено.

Основные понятия среды разработки

Краткое описание структуры проекта

В среде программирования аппаратная конфигурация контроллера представлена в виде дерева, где контроллер - головное, главное устройство (Device), а внутренние и внешние устройства связаны с ним по определенным иерархическим правилам (Рисунок 27).

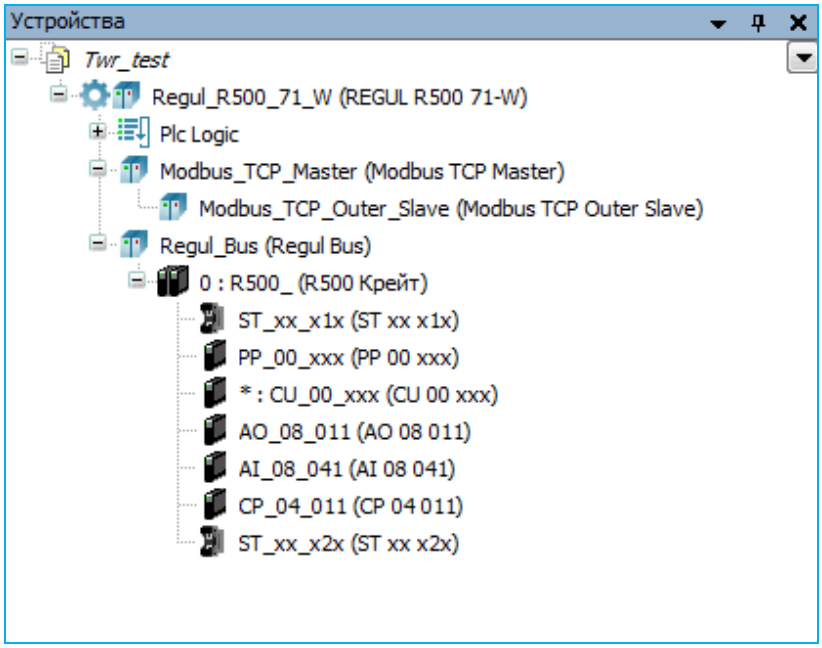


Рисунок 27 – Устройства в иерархической структуре контроллера

Программа (ПЛК-программа) – реализуемый пользователем алгоритм, целью которого является управление технологическим процессом, аварийная защита оборудования, телеметрия и так далее. Состоит из программных компонентов (POU) таких как: программный код, функция, глобальные переменные, функциональные блоки, а также метод, действие, интерфейс, объект типа данных (DUT) или какой-либо внешний файл произвольного формата. Программа привязана к контроллеру и занимает определенное место в структуре контроллера (Рисунок 28). Для контроллера может быть создана одна или несколько программ.

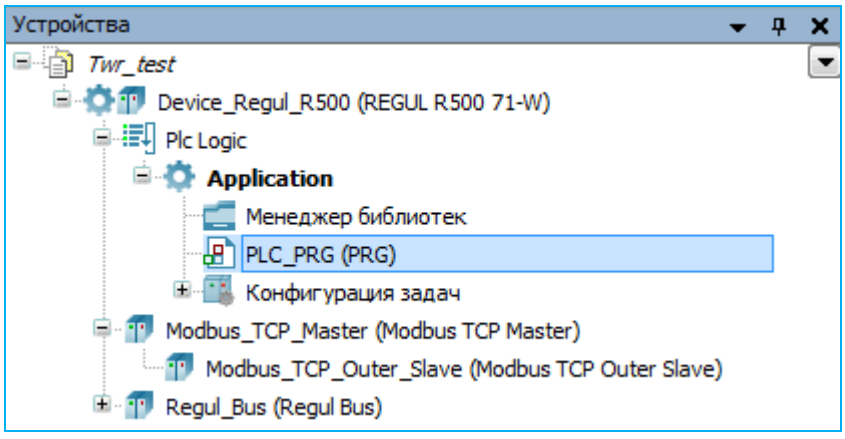


Рисунок 28 – ПЛК-программа в иерархической структуре контроллера

Приложение – это набор объектов, необходимых для запуска конкретного экземпляра ПЛК-программы на конкретном контроллере (далее по тексту – приложение или прикладная программа). В дереве устройств представлено объектом **Application** (Рисунок 29).

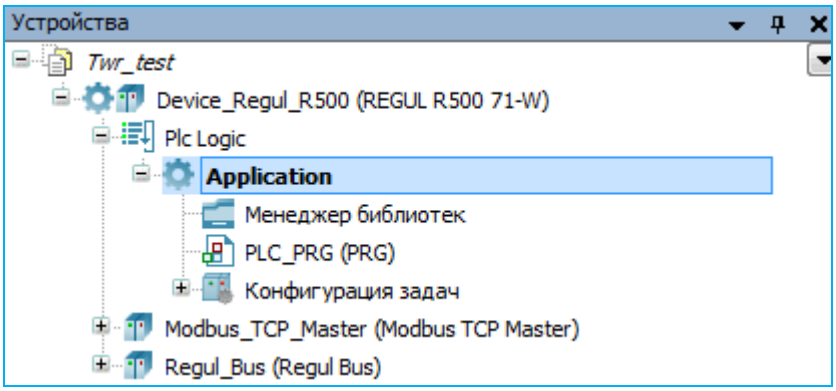


Рисунок 29 – Приложение в иерархической структуре контроллера

Задача – это единица обработки МЭК программы. Имеет название, приоритет и тип.

Тип определяет условие вызова задачи. Условием может служить время (циклическое или свободное выполнение) или событие (например, превышение заданного порога глобальной переменной).

Для каждой задачи назначается ряд программ, которые будут в ней выполняться.

Для каждой задачи можно задать сторожевой таймер (контроль времени выполнения). Помимо этого, существует возможность непосредственно связать системные события (Старт, Стоп, Сброс) с выполнением определенных POU проекта.

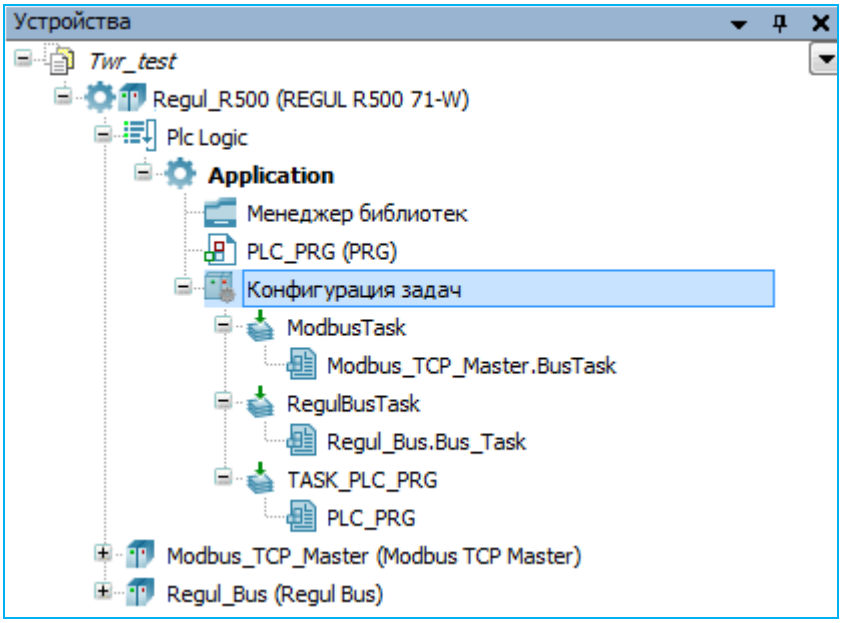


Рисунок 30 – Конфигурация задач в иерархической структуре контроллера

Кроме устройств, ПЛК-программы, задач еще существует конфигурация устройства, касающаяся соединения, параметров, соотнесения входов/выходов.

Всё это – устройства, программные компоненты, приложения, настройки и так далее, относящиеся к конкретному контроллеру, объединены в **Проект**, то есть собраны в одном месте, структурированы по определенным правилам, имеют установленные связи между разными объектами.

Проект

В одном проекте можно настраивать и программировать несколько контроллеров не зависимо от типа. В этом случае в проекте присутствует несколько контроллеров, каждый из которых имеет в своем составе выполняемую прикладную программу (Application). Приложение, выделенное полужирным шрифтом, находится в состоянии *Активное приложение* и именно оно будет загружаться при запуске контроллера. Выбор другого приложения в качестве активного осуществляется через контекстное меню (см. пункт «Приложение» подраздела «Основные понятия среды разработки»).

Есть возможность создавать отдельный проект для каждого контроллера. Если есть потребность работать одновременно с двумя или несколькими контроллерами (сравнивать параметры, например), можно открыть несколько экземпляров среды разработки Astra.IDE, в каждом – отдельный проект.

Проект создается с помощью Мастера конфигурации Regul во время выбора контроллера (подраздел «Построение конфигурации контроллера с помощью мастера»).

Конфигурация проекта и среды разработки могут быть определены в диалогах **Инструменты ⇨ Настройка** (настройка среды) и **Инструменты ⇨ Опции**. Базовая конфигурация нового проекта (структура меню, заданные объекты) определяется текущим шаблоном проекта.

Для задания установок проекта используется объект **Установки проекта**, который по умолчанию отображается в окне **POU**. Также существует объект **Информация о проекте**, который добавляется в окно **POU** сразу после того, как он был вызван соответствующей командой меню (обычно меню **Проект**). Диалог **Информация о проекте** можно использовать для редактирования или просмотра специфической информации о проекте, такой как: данные файла, статистика объектов, имя автора.

Файл проекта может быть защищен паролем или электронным ключом. Кроме того, предусмотрена возможность распечатать проект как текстовый документ.

Устройство, дерево устройств

Каждое устройство представляет собой целевой аппаратный объект. Примеры: контроллер, шинный соединитель, модуль ввода/вывода, монитор.

Окно **Устройства** (дерево устройств) содержит не только устройства, но и все объекты, необходимые для запуска приложения на контроллере.

Общие понятия:

– корневым узлом дерева всегда является проект ;

– конфигурация контроллера определяется топологической структурой устройств в дереве устройств. В отдельных диалогах выполняется лишь конфигурация конкретных устройств или параметров задачи. Таким образом, аппаратная структура представлена в дереве устройств соответствующим расположением объектов-устройств;

– каждый объект дерева устройств представлен символом, символьным именем (редактируется) и типом устройства (то есть имя устройства, заданное в описании устройства);

– устройство может быть программируемым или просто настраиваемым. Тип устройства определяет его возможную позицию в дереве ресурсов, а также какие ресурсы можно вставлять под данным устройством. Программируемые устройства обозначаются дополнительным узлом **Plc Logic**, который автоматически вставляется под устройством. Под этим узлом можно вставлять объекты, необходимые для программирования устройства (приложения, списки текстов и так далее), а также функциональные объекты, такие как «Менеджер параметров»;

– конфигурация устройства, касающаяся соединения, параметров, соотнесения входов/выходов, выполняется в диалоге устройства (редакторе устройства), который открывается двойным щелчком мыши по названию устройства в дереве устройств.

В среде разработки значок рядом с устройством указывает на его состояние:

– : устройство запущено;

– : устройство пока не запущено или не настроено, либо содержит ошибки;

– : устройство запущено, доступна диагностическая информация;

– : устройство настроено, но не запущено;

– : устройство остановлено; обмен данными с устройством не производится;

– : устройство находится в пассивном (не диагностируемом) состоянии.

Правила организации и конфигурирования объектов в дереве устройств:

– добавить устройство можно с помощью контекстного меню (подраздел «Описание интерфейса. Добавление объектов»);

– изменить позицию объекта можно перетаскиванием с зажатой клавишей мыши;

– скопировать объект в другое место можно с помощью стандартных команд категории «Буфер обмена» (Вырезать, Копировать, Вставить) или перетащив выбранный объект с

зажатой кнопкой мыши и клавишей ***Ctrl***. В случае, если вставляемый объект может быть помещен как ниже текущего элемента, так и над ним, откроется диалог **Выберите позицию вставки**, в котором необходимо указать позицию для вставки объекта. При добавлении устройств с помощью функции **Копировать и вставить** новое устройство получает то же имя, к которому присоединяется номер по возрастанию;

– только объекты **Устройство** могут помещаться непосредственно под корневым узлом с именем проекта. Если в пункте контекстного меню **Добавить объект** вы выбрали другой тип объекта, например, *Список текстов*, он будет добавлен в окно **POU**. Если не выбрано ни одного элемента, будет автоматически выбран корневой узел.

Приложение

Основные моменты работы с приложениями:

– приложение представлено объектом **Приложение** в дереве устройств. Он может быть вставлен только под узлом **Plc Logic** (программируемое устройство). Под приложениями могут быть вставлены другие программные объекты, такие как DUT (объект типа данных), GVL (список глобальных переменных), или объекты визуализации;

– конфигурация задач должна быть вставлена под каждым приложением;

– при попытке загрузить приложение на целевое устройство (контроллер или эмулирующее устройство) будет выполнена проверка текущего приложения на контроллере, а также проверка на соответствие параметров этого приложения и приложения в конфигурации проекта. При несоответствии будут выдаваться сообщения, после чего можно будет заменить приложение на контроллере;

– приложение, с которым вы собираетесь работать, должно иметь состояние *Активное приложение*. Это устанавливается через контекстное меню, активное приложение будет выделено полужирным шрифтом (Рисунок 31).

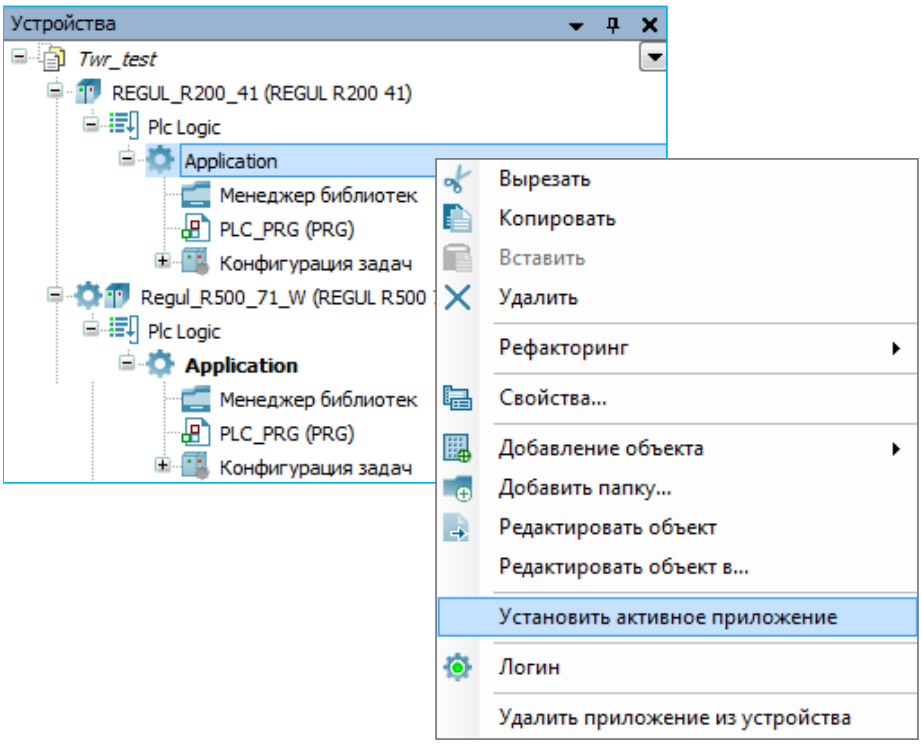


Рисунок 31 – Установка активного приложения

POU

POU (Program Organization Unit) – компонент организации программы. В большинстве случаев под этим термином понимается любой объект, являющийся составной частью ПЛК-программы.

Программные компоненты могут относиться к конкретному устройству. В этом случае другие устройства проекта не смогут использовать эти POU. Зона видимости таких объектов ограничена устройством, POU отображаются в окне **Устройства**.

В рамках одного проекта могут быть созданы программные компоненты, общие для нескольких устройств. Такие объекты имеют зону видимости глобальную по проекту, отображаются в окне **POU**. Для запуска приложения на конкретном устройстве вызывается экземпляр программного компонента.

Просмотр и изменение программных компонентов выполняется в окне редактора. Объектом POU может быть программа, функция, функциональный блок, а также метод, действие, интерфейс, DUT (объект типа данных) или какой-либо внешний файл произвольного формата.

Предусмотрена возможность задавать определенные свойства (такие как условия компиляции и так далее) отдельно для каждого объекта POU.

Конфигурация задач

В **Конфигурации задач** определяется одна или несколько задач, контролирующих выполнение прикладной программы. Она является неотъемлемым ресурсным объектом для

приложения и должна быть помещена под приложением в дереве устройств. Задача может вызывать как POU для конкретного приложения, так и программы, расположенные в окне **POU**.

Каждая задача обладает определенным приоритетом и реализуется с вытесняющей многоприоритетной многозадачностью. Многозадачность заключается в способности обрабатывать до 100 прикладных программ. Одновременно может выполняться только одна задача. Чтобы настроить приоритет, выберите в окне устройств элемент **Конфигурация задач**. Далее, в раскрывшемся дереве устройств, двойным щелчком левой кнопки мыши по вкладке задачи откройте отдельное окно **Конфигурация** (Рисунок 32).

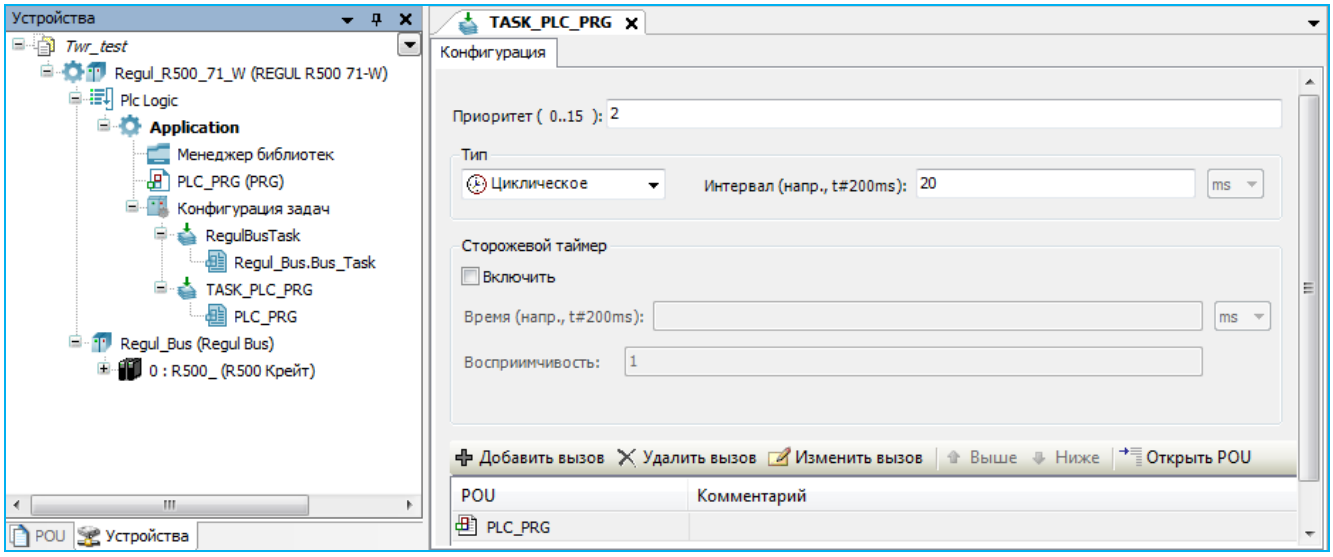


Рисунок 32 – Диалоговое окно конфигурации

ВНИМАНИЕ!

Рекомендуется не изменять установленные по умолчанию параметры системных задач:

- **Modbus:** ModbusTask
- **IEC:** Slave104Task, Master104Task
- **RegulBus:** RegulBusTask

В поле **Приоритет** будет приведено значение приоритета по умолчанию, определяемое числом в диапазоне от 0 до 15. Чем меньше число, тем выше приоритет, соответственно 0 – самый высокий, 15 – самый низкий приоритет. Если две или более задачи должны получить управление одновременно, то запустится задача с более высоким приоритетом. Задача с более высоким приоритетом может прерывать выполнение задачи с более низким приоритетом. Выполнение задачи с более низким приоритетом продолжится после окончания цикла задачи с более высоким приоритетом. На примере показано выполнения двух задач с разным приоритетом (Рисунок 33).

В том случае, если высокоприоритетная задача в процессе своего выполнения освободит ресурсы (например, в ожидании какого-либо события или ответа), то в это время будут выполняться низкоприоритетные задачи.

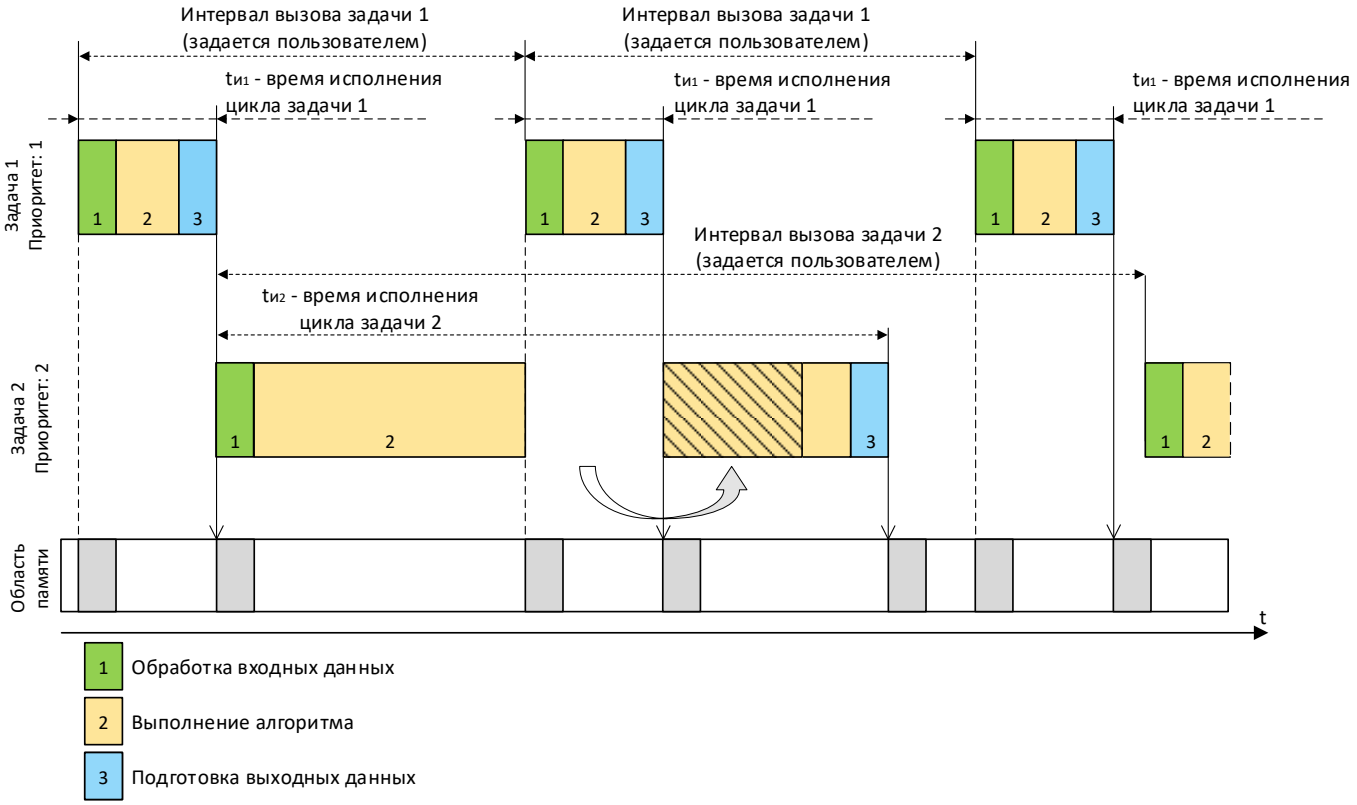


Рисунок 33– Обработка задач в условиях многозадачности

В области памяти производится обновление значений переменных. Выполнение задачи происходит со значениями переменных, зафиксированных в области памяти на момент вызова задачи. После окончания задачи обработанные данные выгружаются в область памяти.

Если одни и те же переменные используются в разных задача, то обновление значений этих переменных в области памяти происходит при вызове каждой задачи. Если задача прерывается более приоритетной задачей, в которой задействованы переменные, используемые в прерванной задаче, то в момент вызова высокоприоритетной задачи они будут обновлены. И в этом случае, окончание выполнения прерванной задачи будет осуществляться уже с обновленными значениями переменных, которые могут отличаться от значений, при которых была выполнена первая часть вытесненной задачи.

Это же ситуация характерна для случая, когда вытесняемая задача использует выходные данные высокоприоритетной задачи.

В диалоговом окне конфигурации, при смене значения в поле **Приоритет**, в правом углу поля высветится знак  . Наведите на знак и появится описание приоритета.

	**ВНИМАНИЕ!** Запрещается присваивать приоритеты «0» или «1» пользовательским задачам, данные приоритеты закреплены за задачами RegulBus и Redundancy соответственно (см. таблицу 1)

Ниже в таблицах представлено описание параметров: в таблице 1 - диапазон значений приоритетов, в таблице 2 - типы задач и в таблице 3 - свойства.

Таблица 1 - Диапазон значений приоритетов

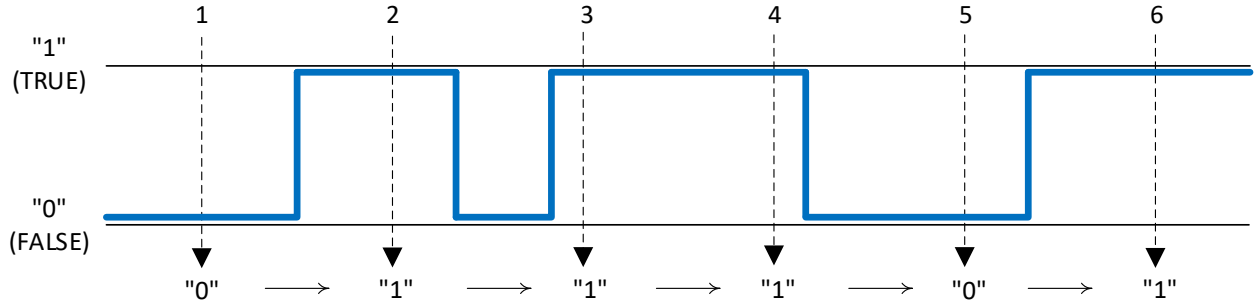
Значение	Приоритет
0	Высший приоритет - highest realtime priority (RegulBus only)
1	Средний приоритет - realtime priority (Redundancy only)
2	Средний приоритет (над сетевой системой) - realtime priority (above network system)
3	Средний приоритет (над сетевой системой) - realtime priority (above network system)
4	Средний приоритет - realtime priority
…	Средний приоритет - realtime priority
13	Средний приоритет - realtime priority
14	Низкий приоритет (равен системным службам) - low priority (equal to system services)
15	Низкий приоритет - lowest priority (ниже системных служб)

Таблица 2 – Тип выполнения вызова задачи

Тип	Описание
Циклическое 	Задача вызывается циклически через заданный интервал времени
Свободное выполнение 	Задача вызывается вновь автоматически, сразу же после окончания, в непрерывном цикле, без задания каких-либо интервалов, выполнятся в фоновом режиме. Например: Сбор и обработка параметров системы и вывод их на экран. **НЕ ИСПОЛЬЗУЙТЕ ДАННЫЙ ТИП ДЛЯ ЗАДАЧИ ЦИКЛА ШИНЫ!**
Событие 	Задача вызывается по нарастающему фронту значения логической переменной, которая определена в поле **Событие** (поле появляется при выборе типа вызова: **Событие**)
Статус 	Задача начинает выполняться, если переменная, указанная в поле **Событие**, возвращает значение **TRUE**. Задача вызывается по событию, которое определено в поле **Событие** (поле появляется при выборе типа вызова: **Статус**).

Таблица 3 – Свойства задачи

Свойство	Описание
Интервал (задачи)	Период времени, после которого задача должна быть вызвана в очередной раз. Необходим для типа **Циклическое**, когда событию требуется заданное время
Событие	Глобальная переменная, инициализирующая запуск задачи. Свойство необходимо для типов задач **Событие** и **Статус**. Необходимо обратить внимание на различие: – запуск задачи типа **Статус** выполняется, если заданное событие возвращает **TRUE**; – запуск задачи типа **Событие** требует переключения события с **FALSE** на **TRUE**. Необходимо учесть, что, если частота сканирования задач слишком низкая, то передний фронт события может остаться незамеченным (Рисунок 34)



Вызов задачи по **СТАТУСУ**	"0" – не запущена	"1" – запущена	"1" – запущена	"1" – запущена	"0" – не запущена	"1" – запущена
Вызов задачи по **СОБЫТИЮ**	"0" – не запущена	с "0" на "1" – запущена	"1" – не запущена, слишком быстрый переход с "1" на "0" и назад к "1"	с "1" на "1" – не запущена	с "1" на "0" – не запущена	с "0" на "1" – запущена

Рисунок 34– Вызов задачи по Статусу или по Событию

Для каждой задачи можно определить контроль времени выполнения (сторожевой таймер). Если сторожевой таймер включен (установлен флажок ☑ в поле **Включить**), то задача будет прервана с установкой статуса исключительная ситуация, если её выполнение заняло больше времени, чем задано в поле **Время**, произошла задержка. По истечении времени, указанного в поле **Время**, задача должна завершить работу.

Также необходимо принимать в расчет значение, указанное в поле **Восприимчивость**. Под восприимчивостью подразумевают допустимое количество раз превышения времени сторожевого таймера, происходящих без последующего формирования статуса исключительной ситуации.

Возможные варианты:

- **несколько задержек подряд:**
  - **восприимчивость** равна «**0**», «**1**» - исключение в **1** цикле вызова задачи;

- **восприимчивость** равна «**2**» - исключение во **2** цикле вызова задачи;
- **восприимчивость** равна «**N**» - исключение в **N** цикле вызова задачи;

– **одна задержка:** текущее время выполнения одной задачи будет больше чем заданное время, умноженное на восприимчивость.

Например: **Время**=t#15ms, **Восприимчивость**=5. При данных условиях статус возникнет:

- как только в течении 5 последовательных циклов вызова задачи фактическое время выполнения каждой задачи будет превышать 15мс;
- как только одна задача будет выполняться более 75мс.

Для определения и отображения базовых параметров конфигурации задач, в окне устройств наведите мышь на элемент **Конфигурация задач** и двойным щелчком левой кнопки мыши откройте отдельное окно **Конфигурация задач** (Рисунок 35).

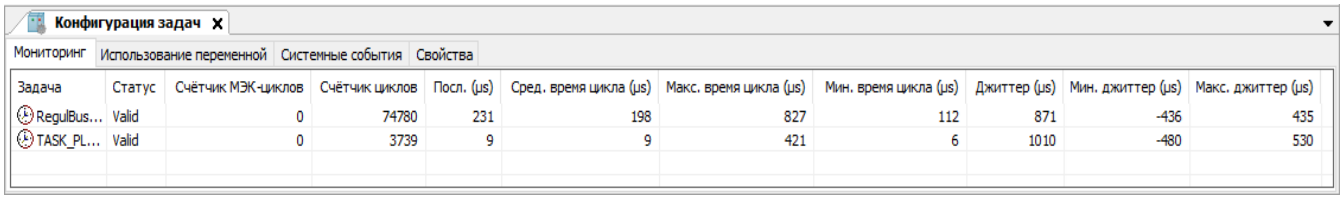


Рисунок 35– Диалоговое окно конфигурации задач. Вкладка мониторинг

На вкладке **Мониторинг** в режиме онлайн отображается состояние и текущая статистика, характеризующая время выполнения задачи. Описание представлено в таблице 4.

Таблица 4 – Мониторинг задач

Наименование	Описание
Задача	Имя задачи
Состояние	Задача может быть в одном из следующих состояний: – **Не сгенерирована** (**Not created**) – не запускалась с момента последнего обновления; – **Сгенерирована (Created)** – распознается в системе, но в данный момент не в работе; – **Допустима (Valid)** – работает нормально; – **Исключение (Exception)** – возникла исключительная ситуация
Счетчик МЭК-циклов	Количество циклов, выполненных с момента запуска приложения (0, если целевая система не поддерживает функцию счетчика)
Счетчик циклов	Количество выполненных циклов с момента подключения к ПЛК. В зависимости от целевой системы, может быть равным **Счетчику МЭК-циклов** или больше (тогда, когда приложение не работает, но продолжает считать циклы)
Посл. (µs)	Последнее измеренное время выполнения цикла (мкс)
Сред.время цикла (µs)	Среднее время выполнения всех циклов (мкс)

Наименование	Описание
Макс.время цикла (µs)	Максимальное измеренное время выполнения из всех циклов (мкс)
Мин.время цикла (µs)	Минимальное измеренное время выполнения из всех циклов (мкс)
Джиттер (µs)	Последний измеренный джиттер (мкс) Джиттер показывает разницу времени между фактическим вызовом задачи и плановым ее началом. Например, если задача должна запускаться каждые 900 мкс, а на деле она запускается через 1000 мкс, то джиттер составляет 100 мкс. Джиттер может быть как положительный, так и отрицательный.
Мин.джиттер (µs)	Минимальный измеренный джиттер (мкс)
Макс.джиттер (µs)	Максимальный измеренный джиттер (мкс)

Для фиксации корректного значения параметра «Макс. время цикла», рекомендуется после запуска контроллера сбросить значения в «ноль», так как при старте время выполнения задачи может значительно превышать реальное максимальное значение в последующие циклы исполнения задачи. Чтобы сбросить значения параметров любой из задач необходимо выполнить следующую последовательность действий:

– установите курсор на поле с именем задачи (в столбце **Задача**)
– по щелчку правой кнопки мыши откройте команду «Сброс»,
– выберите команду «Сброс», после чего произойдет обнуление значений (Рисунок 36).

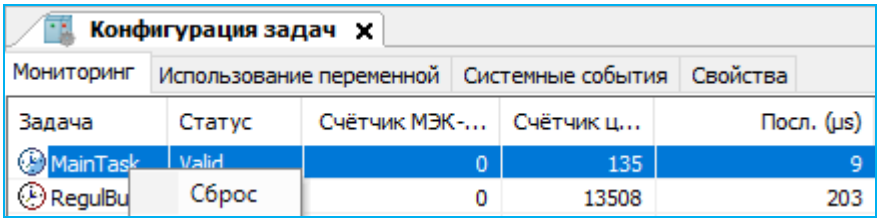


Рисунок 36 – Сброс значений параметров задач

КОНФИГУРИРОВАНИЕ АППАРАТНОЙ ЧАСТИ КОНТРОЛЛЕРА

Контроллер Regul RX00 имеет блочно-модульную конструкцию, состоящую из одного или нескольких крейтов, которые, в свою очередь, включают в себя модули различного типа. Подробное описание конструкции контроллера приведено в документе «Системное руководство» соответствующей модели контроллера.

Для настройки и программирования контроллера нужно построить в среде разработки Astra.IDE конфигурацию контроллера, полностью соответствующую реальной аппаратной конфигурации. Вся конструкция (состав модулей, их расположение в крейте, количество и положение крейтов) должна быть идентична существующей структуре контроллера.

Построение конфигурации контроллера с помощью мастера

Конфигурацию контроллера можно строить в новом проекте или в уже существующем. Если контроллер будет относиться к новому проекту, сразу переходите к шагу активации Мастера конфигурации Regul, проект будет автоматически создан мастером на определенном этапе. Если предполагается добавить контроллер в уже существующий проект, то откройте проект с помощью пункта основного меню **Файл ⇨ Открыть проект…** или **Файл ⇨ Недавние проекты ▸**.

Для активации Мастера конфигурации Regul выберите на панели инструментов кнопку  (крайняя слева). Откроется окно **Мастер конфигурации Regul** (Рисунок 37).

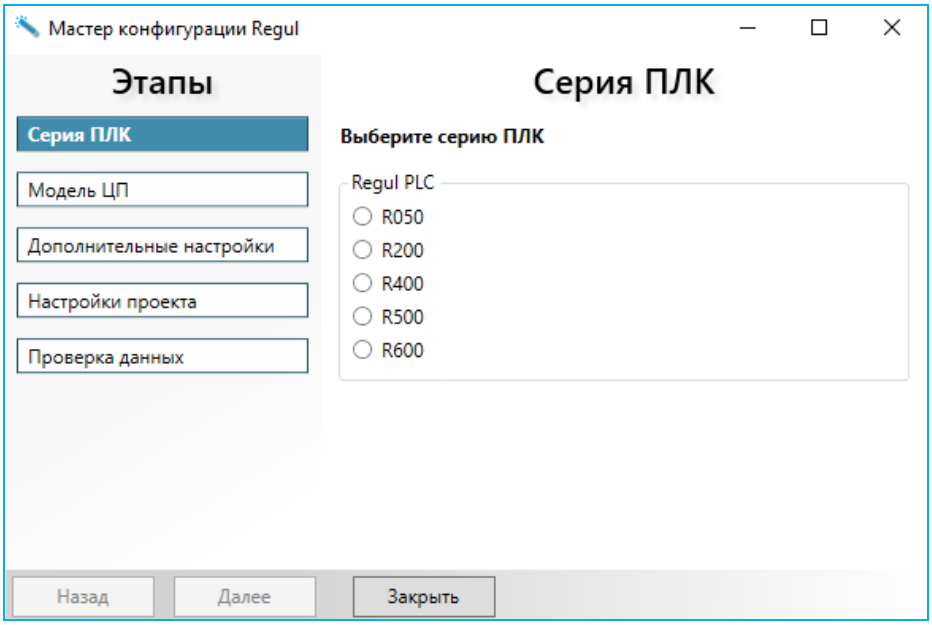


Рисунок 37 – Мастер конфигурации Regul. Выбор серии контроллеров

Выберите, к какой серии относится контроллер. Чтобы это определить, посмотрите, какой серии принадлежит модуль этого контроллера, содержащий центральный процессор. Остальные модули контроллера могут относиться как к этой же серии, так и к другим сериям, например, контроллер R200 с модулями R200 и R500.

Нажмите кнопку ***Далее***. Произойдет переход к выбору модели центрального процессора (Рисунок 38).

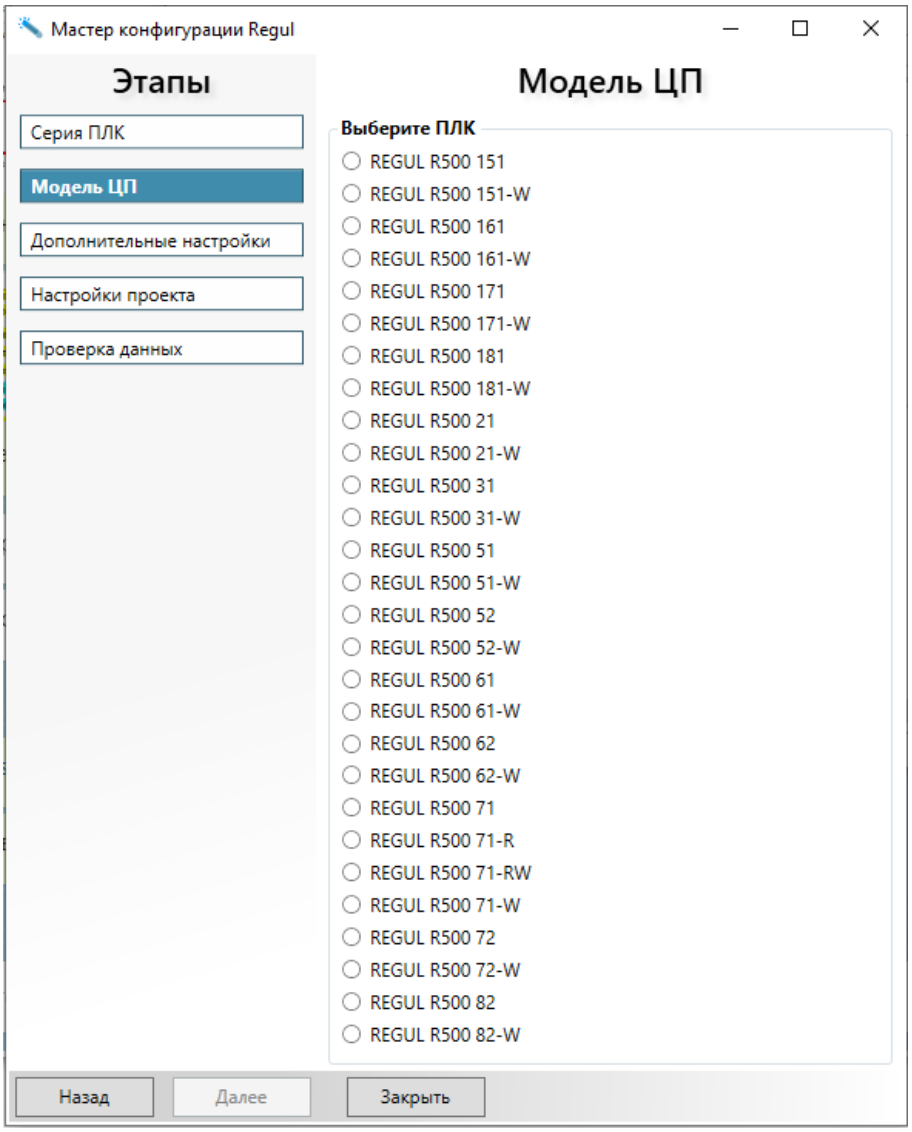


Рисунок 38 – Мастер конфигурации Regul. Выбор контроллера

Выберите модель центрального процессора. В дальнейшем при необходимости можно будет сменить модель контроллера с помощью команды **Обновить устройство**. Нажмите кнопку ***Далее***. Произойдет переход к дополнительным настройкам (Рисунок 39).

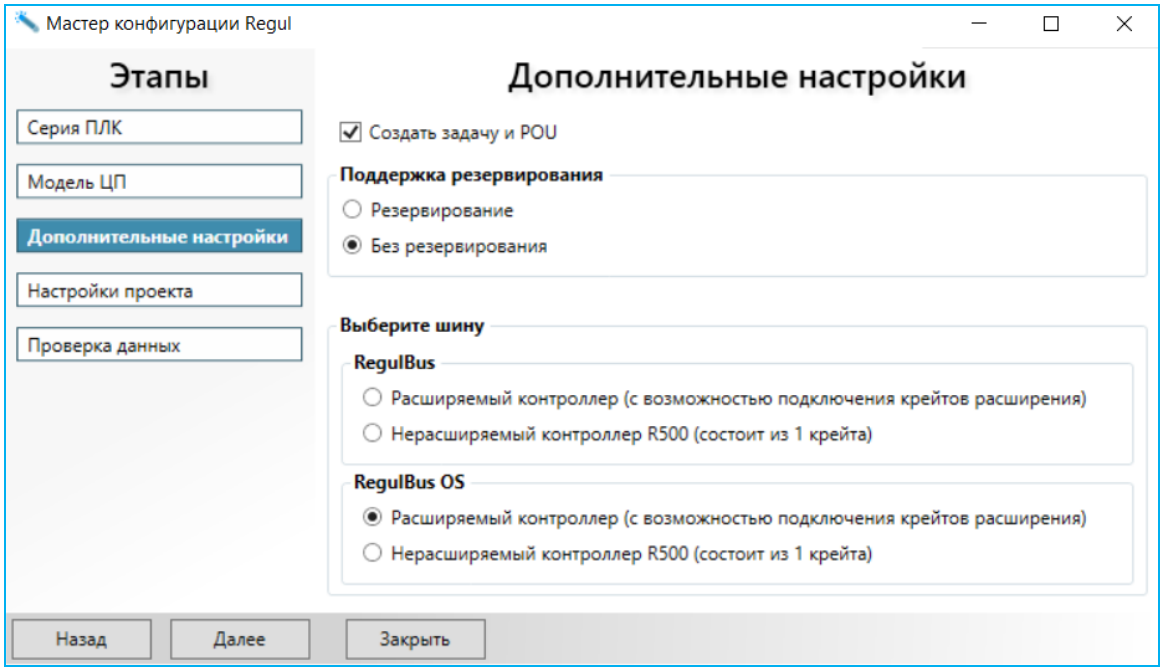


Рисунок 39 – Мастер конфигурации Regul. Дополнительные настройки

Флажок ☑ в поле **Создать задачу и POU** означает, что в проекте в структуре контроллера будут автоматически созданы шаблон пользовательской программы и шаблон задачи.

В блоке **Поддержка резервирования** укажите (поставив переключатель), предусмотрена ли в проекте поддержка резервирования. В дальнейшем этот параметр может быть изменен в редакторе шины (см. пункт «Редактор шины» подраздела «Конфигурирование крейтов»). Построение систем резервирования подробно описано в документе «Конфигурирование резервированной системы на контроллерах серии Regul RX00. Руководство пользователя». Для контроллеров серии R400 резервирование не предусмотрено.

i	**ИНФОРМАЦИЯ** Начиная с версии СПО 1.7.0.0, доступна нативная версия внутренней шины данных контроллера (**RegulBus OS)**, реализованная на уровне операционной системы. В **Приложении К** представлен перечень модулей, поддерживающих шину **RegulBus** и/или **RegulBus OS**

В следующем блоке выберите версию шины **RegulBus** или **RegulBus OS** (поставив переключатель). В дальнейшем этот параметр может быть изменен в редакторе шины (см. пункт «Редактор шины» подраздела «Конфигурирование крейтов»).

i	**ИНФОРМАЦИЯ** Начиная с версии СПО 1.7.1.0, по умолчанию выбирается версия шины **RegulBus OS**

Для контроллеров серии R500 выберите (поставив переключатель), является контроллер расширяемым или нет. Нерасширяемый контроллер состоит из одного крейта, другие крейты в

него добавить нельзя. Расширяемый контроллер может состоять из одного или нескольких крейтов. Контроллеры других серий всегда являются расширяемыми, выбирать не требуется.

Нажмите кнопку ***Далее***. Произойдет переход к настройкам проекта (Рисунок 40).

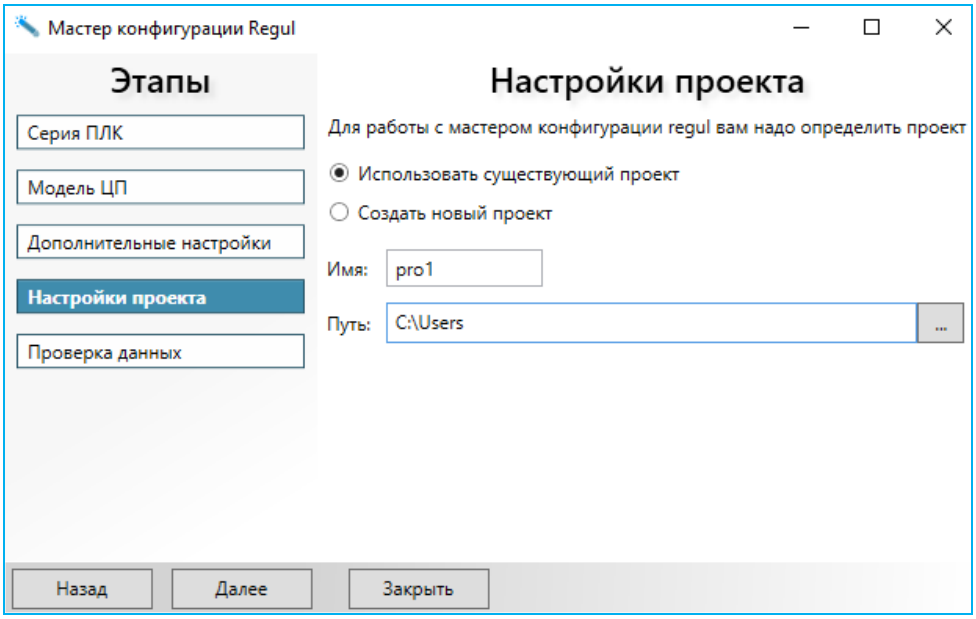


Рисунок 40 – Мастер конфигурации Regul. Настройки проекта

Если вы создаете конфигурацию контроллера в уже открытом, существующем проекте, то оставьте переключатель в поле **Использовать существующий проект**.

Если для создаваемой конфигурации контроллера требуется отдельный проект (отличный от открытого) или проект еще не был создан, то поставьте переключатель в поле **Создать новый проект**. В поле **Имя:** введите название проекта. В поле **Путь:** вручную или с помощью кнопки ... укажите путь к директории, где будет находиться файл проекта с расширением **.project*. При ошибочном вводе названия проекта, уже существующего в папке, появится информационное сообщение следующего вида: «*Проект с указанным именем уже существует*».

Для удобства работы рекомендуется создать отдельную папку, в которую в процессе работы будут сохраняться файлы, связанные с этим проектом.

Нажмите кнопку ***Далее***.

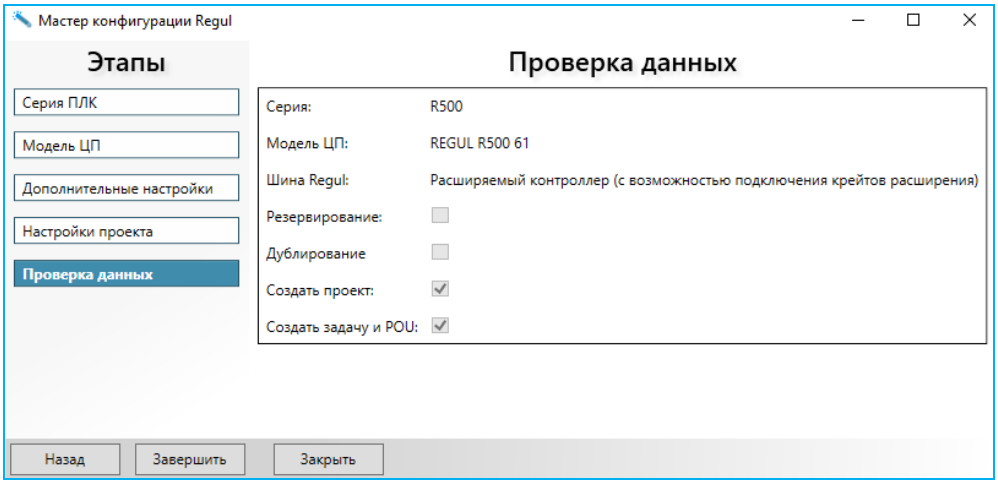


Рисунок 41 – Проверка данных

На этом шаге (Рисунок 41) проверьте, что все параметры указаны верно. При необходимости воспользуйтесь кнопкой ***Назад***. Чтобы закончить создание «базы» контроллера нажмите кнопку ***Завершить***. Окно мастера конфигурации автоматически закроется, а в окне дерева устройств появится созданная структура контроллера (Рисунок 42).

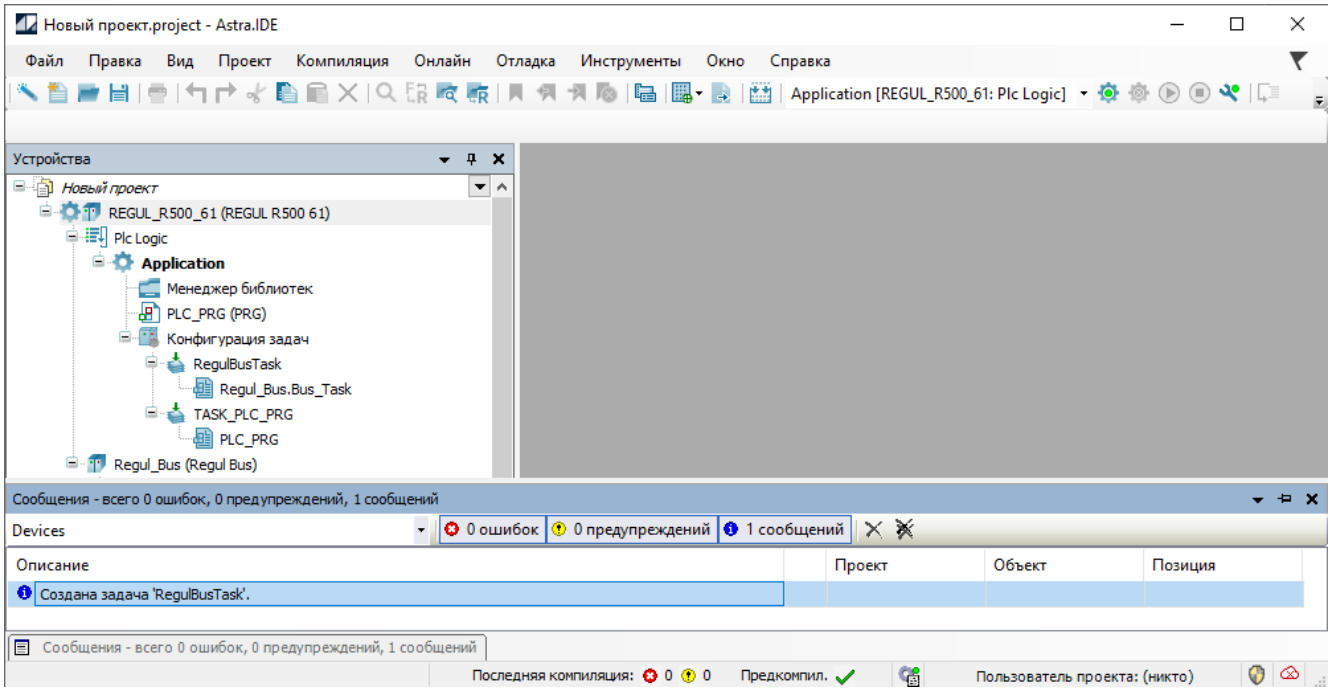


Рисунок 42 – В проект добавлен контроллер

В программе предусмотрены возможности сменить устройство, переименовать или удалить (подраздел «Описание интерфейса»).

Конфигурирование крейтов

Добавление крейтов в проект

Каждый контроллер должен иметь в своем составе **базовый крейт** – крейт с модулем центрального процессора (и другими модулями в соответствии с проектом). Для увеличения

канальной емкости контроллера к базовому крейту можно подключить до 255 **крейтов расширения** (крейты с различными модулями, но без модуля центрального процессора).

В серии R400 контроллер сам является базовым крейтом. В нерезервированном контроллере в сериях R600, R500, R200, R050 может быть только один базовый крейт, содержащий один (и только один) модуль центрального процессора. Особенности построения аппаратной конфигурации контроллеров в резервированной системе описаны в документе «Конфигурирование резервированной системы на контроллерах серии Regul RX00. Руководство пользователя».

Крейты расширения могут содержать модули той же серии, что и модули базового крейта, а могут состоять из модулей других серий Regul, но каждый крейт должен иметь в своем составе только модули одной серии. Так, например, контроллер с базовым крейтом R500 может иметь несколько крейтов расширения R500 и несколько крейтов расширения R200. Но недопустима ситуация, когда на борту одного крейта есть модули R500 и R200 одновременно.

При добавлении контроллера в проект с помощью мастера конфигурации Regul автоматически создается шина RegulBus (RegulBusOS), а на ней автоматически размещается базовый крейт (R050 CU Крейт, R200 CU Крейт, R400 ПЛК, R500 Крейт или R600 Базовый крейт) (Рисунок 43).

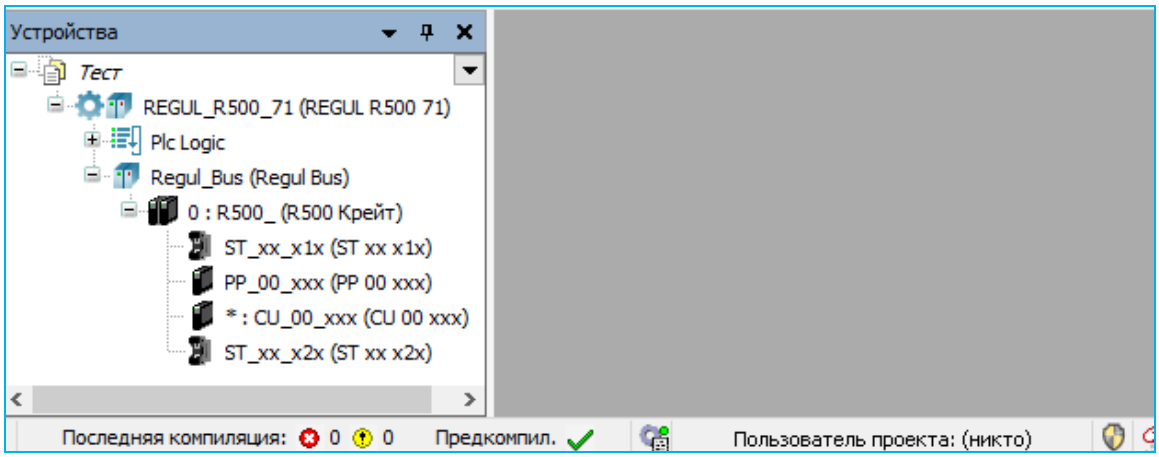


Рисунок 43 – Пример автоматического создания шины и базового крейта

Для добавления крейтов расширения в окне дерева устройств поместите курсор на название шины RegulBus(RegulBusOS), нажмите правую кнопку мыши. В появившемся контекстном меню выберите пункт **Добавить устройство…** Откроется окно, где, с помощью кнопки ⊞ раскрывая список устройств, выберите *Regul* → *Крейты* → *R__ Крейт*. Нажмите кнопку ***Добавить устройство***. Выбранный крейт появится в проекте в дереве устройств. Можно, не закрывая диалогового окна, добавить в проект еще несколько крейтов.

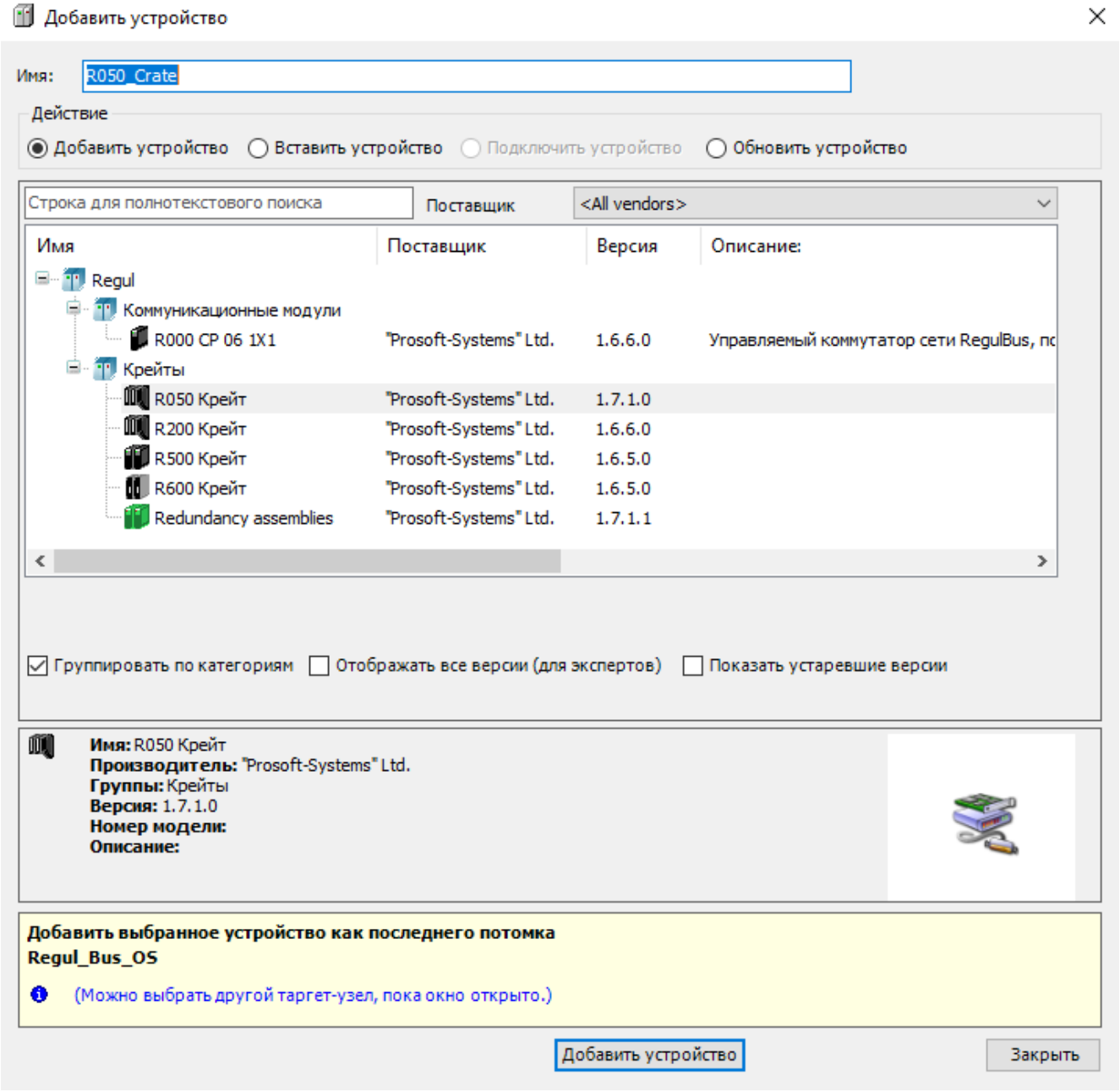


Рисунок 44 – Добавление крейта на внутреннюю шину RegulBus

Для построения разветвлённой и гибкой структуры сети применяют модуль управляемого коммутатора R000 CP 06 1X1, который добавляют в шину RegulBus (RegulBus_OS) аналогично крейтам и используют для подключения крейтов расширения всей модельной линейки Regul (R600, R500, R400, R200, R050). Модуль R000 CP 06 1X1 по функционалу идентичен коммуникационному модулю R500 CP 06 111 (см. «Задание параметров модулей коммуникационного процессора»), но предназначен для одиночной установки (не входит в состав крейта R500). При этом, с целью определения позиции данного модуля в составе контроллера, так же, как и отдельные крейты, он имеет уникальный адрес.

Редактирование конфигурации контроллера

В программе предусмотрена возможность изменения существующей конфигурации контроллера. Крейты можно удалять, добавлять новые (в том числе крейты других серий Regul), переименовывать и переопределять тип крейта (базовый или крейт расширения).

Для удаления крейтов нажмите клавишу ***Delete*** или в контекстном меню выберите команду **Удалить**. Крейт будет удален полностью, включая все модули и все настройки для модулей и крейта.

	**ИНФОРМАЦИЯ** Программа не запрашивает подтверждения на удаление крейта, но удаление можно отменить, если проект не сохранен

Переименование крейтов происходит также, как и переименование других объектов (см. «Описание интерфейса. Переименование объектов»).

Необходимость в переопределении типа крейта может возникнуть тогда, когда требуется передать роль базового крейта другому крейту. В обязательном порядке базовый крейт должен относиться к той же серии Regul, что и контроллер.

	**ИНФОРМАЦИЯ** Для корректного отображения результатов обновления рекомендуется проводить обновление устройства при закрытой вкладке редактора устройства

Если крейты и контроллер принадлежат одной серии, выполните следующие действия:

- смените тип текущего базового крейта:
  - для серии R500 – удалите модуль центрального процессора,
  - для серий R600, R200 и R050 – в контекстном меню выберите пункт **Обновить устройство…** Откроется окно **Обновить устройство**, где в списке крейтов выберите *R600 Крейт, R200 Крейт* или *R050 Крейт* соответственно, нажмите кнопку ***Обновить устройство***. Закройте окно. Состав крейта изменится – будет удален модуль центрального процессора;
- смените тип крейта, назначаемого базовым:
  - для серии R500 – добавьте модуль центрального процессора,
  - для серии R600, R200 и R050 – в контекстном меню выберите пункт **Обновить устройство…** Откроется окно **Обновить устройство**, где в списке крейтов выберите *R600 Базовый Крейт, R200 CU Крейт* или *R050 CU Крейт* соответственно, нажмите кнопку ***Обновить устройство***. Закройте окно. Состав крейта изменится – будет добавлен модуль центрального процессора.

Если крейт, назначаемый базовым, относится к другой серии, чем контроллер, выполните следующие действия:

– замените контроллер (головное устройство в дереве устройств) – в контекстном меню выберите пункт **Обновить устройство…** Откроется окно **Обновить устройство**, где в списке выберите новый контроллер, нажмите кнопку ***Обновить устройство*** (Рисунок 45). Закройте окно.

ВНИМАНИЕ!

При замене, для серии R050 – все прежние крейты (модули, шина) и настройки удаляются автоматически

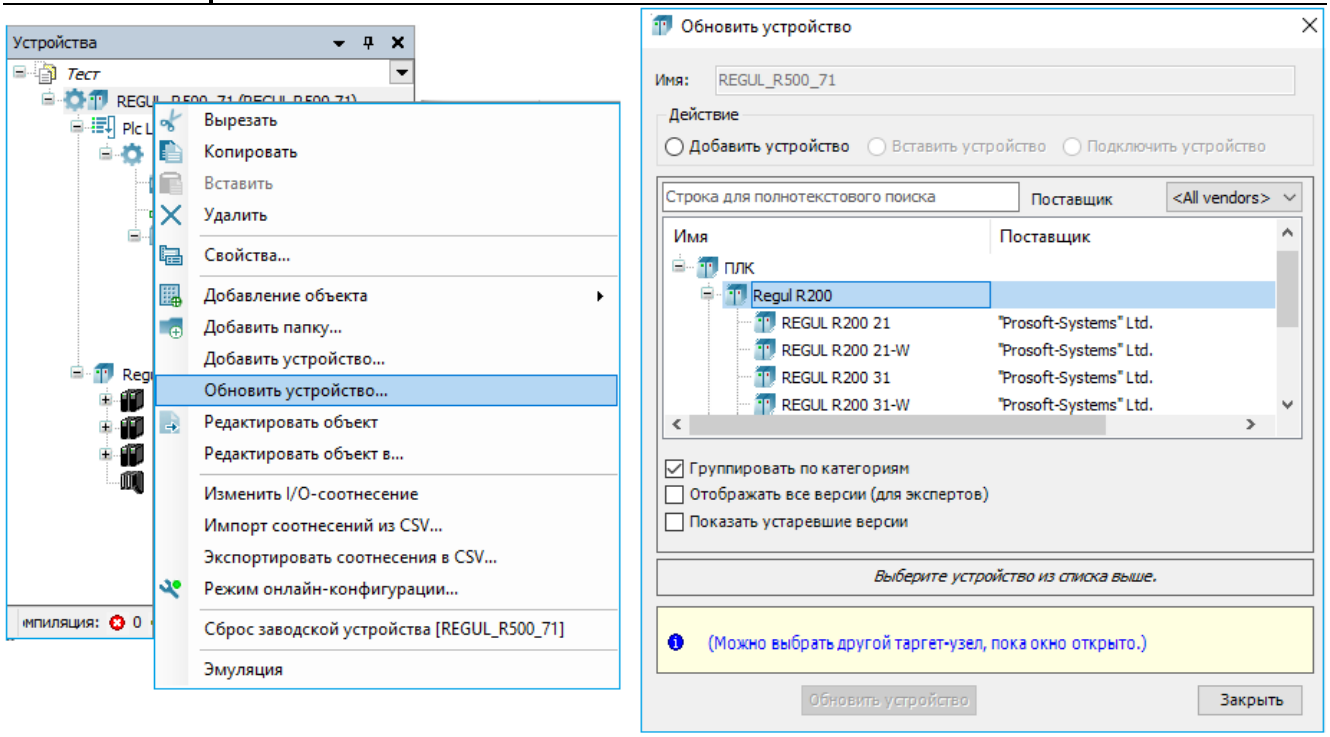


Рисунок 45 – Обновление устройства

– смените тип текущего базового крейта.

ВНИМАНИЕ!

При смене типа крейта все прежние модули и настройки этого крейта удаляются автоматически

В контекстном меню выберите пункт **Обновить устройство…** Откроется окно **Обновить устройство**, где в списке крейтов выберите соответственно *R200 CU Крейт (R200 Крейт)*, *R400 ПЛК*, *R500 Крейт* или *R600 Базовый крейт*, нажмите кнопку ***Обновить устройство***. Закройте окно;

– для серии R500 – добавьте в крейт модуль центрального процессора.

Редактор шины

При создании нового проекта с помощью **Мастера конфигурации Regul** на этапе **Дополнительные настройки** появилась возможность выбора шины, с дальнейшим автоматическим добавлением компонента **RegulBus / RegulBus OS** в дерево устройств.

Журналирование работы шины **RegulBus OS** производится в отдельный лог-файл regul-bus-driver.log (расположен в каталоге /logs/logger/user, рисунок 46) с выводом информации в оперативный журнал.

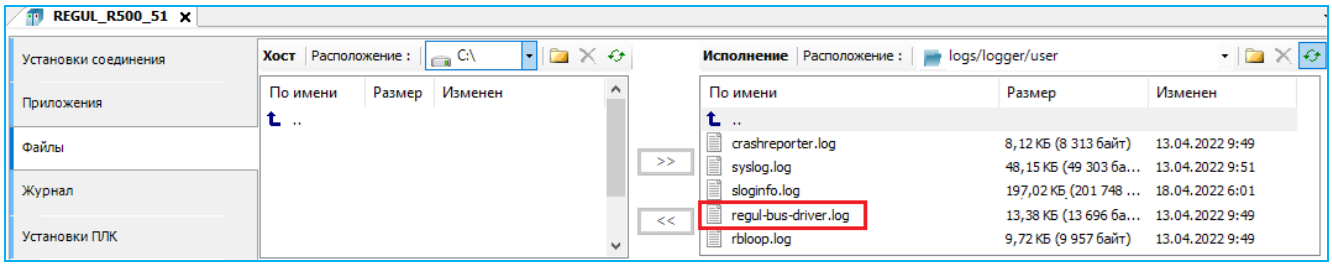


Рисунок 46 – Журналирование работы шины RegulBus OS

При добавлении устройства в конфигурацию (в дерево устройств) происходит журналирование его типа в следующем виде:

```
08.04.2024 08:29:46.903: [I] ======== Hardware configuration - BEGIN ========
08.04.2024 08:29:46.903: [I] Bus: type=RegulBus
08.04.2024 08:29:46.903: [I] R500_: type=R500_Crate, addr=0
08.04.2024 08:29:46.904: [I]   ST_xx_x1x: type=R500_ST_xx_x1x, pid=0401c001
08.04.2024 08:29:46.904: [I]   PP_00_xxx: type=R500_PP00xxx, pid=04014003
08.04.2024 08:29:46.904: [I]   CU_00_xxx: type=R500_CU00xxx, pid=0402c001, mbs=*
08.04.2024 08:29:46.904: [I]   ST_xx_x2x: type=R500_ST_xx_x2x, pid=0401c002
08.04.2024 08:29:46.904: [I]  Crate: type=R500_Crate, addr=1
08.04.2024 08:29:46.904: [I]   ST_xx_x1x_1: type=R500_ST_xx_x1x, pid=0401c001
08.04.2024 08:29:46.904: [I]   PP_00_xxx_1: type=R500_PP00xxx, pid=04014003
08.04.2024 08:29:46.904: [I]   AI_08_041: type=R500_AI08041_v2, pid=04050002
08.04.2024 08:29:46.904: [I]   CP_06_111: type=R500_CP061X1_v30, pid=04088003,
mbs=*
08.04.2024 08:29:46.904: [I]   ST_xx_x2x_1: type=R500_ST_xx_x2x, pid=0401c002
08.04.2024 08:29:46.905: [I]   CP_06_1X1: type=R000_CP061X1_v3, addr=5,
pid=0708c003
08.04.2024 08:29:46.905: [I] ======== Hardware configuration - END ========
```

Также предусмотрено журналирование причины последнего перезапуска (ошибка) крейта или модуля с версией СПО 1.0.33.0 и выше.

В логах наблюдаются записи следующего вида:

```
SDO optional Read I=0x2501 SI=0x01 Val=0x00000000-Ok

<Название модуля/крейт>: Reinit reason: Unknown (Master Request/Connection
error/Power loss/Internal error)
```

где:

– Unknown – неизвестно;

– Master Request-инициатива мастера;

– Connection error – потеря связи с мастером;

– Power loss – потеря питания;

– Internal error – внутренняя ошибка модуля.

Для модуля, не поддерживающего функционал (версия СПО ниже 1.0.33.0):

```
SDO optional Read I=0x2501 SI=0x01 — Fail

<Название модуля/крейт>: Reinit reason: Unknown
```

Например, при потере питания появляются сообщения следующего вида:

```
R050_Crate_ST(ET): Reinit reason: Power loss  /для крейта R050
DI_16_011_1: Reinit reason: Power loss / для модуля ввода
```

Например, при внутренней ошибке модуля CP 01 031:

```
CP_01_031(ET): Reinit reason: Internal error (Stack error)
```

Для перехода в редактор шины дважды щелкните левой кнопкой мыши по названию шины в окне дерева устройств. Откроется вкладка (окно) шины **RegulBus** (Рисунок 47).

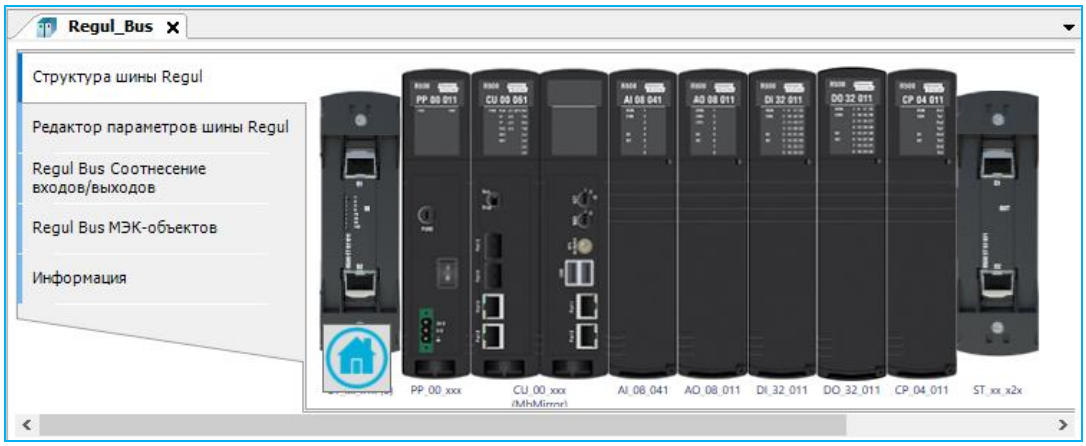


Рисунок 47 – Редактор шины контроллера R500

Перейти с одной версии шины на другую можно одним из следующих способов:

– через обновление. Поставьте курсор в проекте на название шины, нажмите правую кнопку мыши. Появится контекстное меню, в котором выберите **Обновить устройство…** Откроется окно **Обновить устройство**, где выберите шину и нажмите кнопку *Добавить* (Рисунок 48). Произойдет замена шины в дереве устройств;

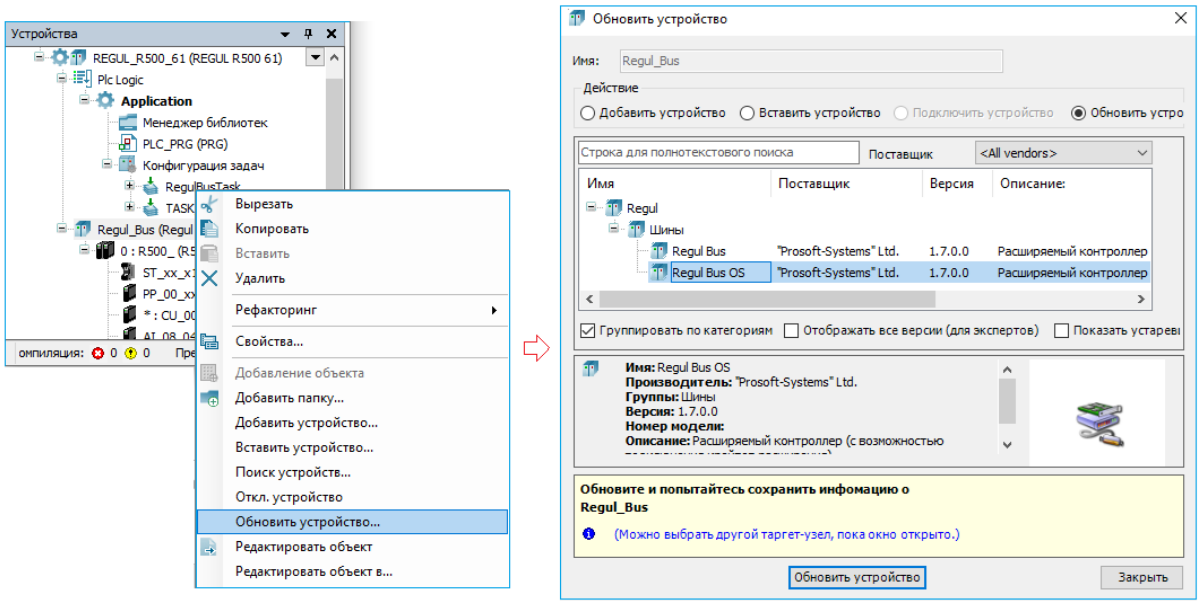


Рисунок 48 – Обновить устройство. Переход на новую версию шины RegulBus OS

– через удаление. Поставьте курсор в проекте на название шины, нажмите правую кнопку мыши. Появится контекстное меню, в котором выберите **Удалить**. Далее поставьте курсор в проекте на название контроллера, нажмите правую кнопку мыши. Появится контекстное меню, в котором выберите **Добавить устройство.** Откроется окно **Добавить устройство,** перейдите **Regul⇨Шины** и выберите необходимую, нажмите кнопку ***Добавить устройство*** (Рисунок 49).

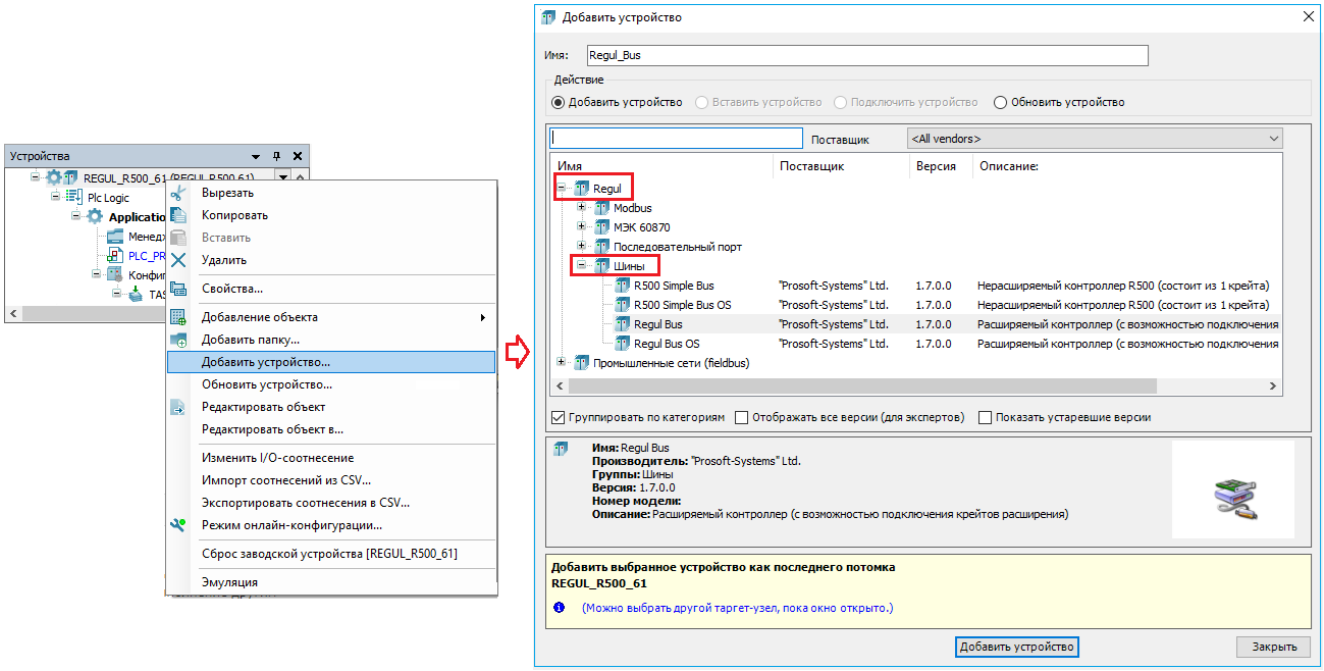


Рисунок 49 - Добавить устройство. Переход на соответствующую шину

Редактор шин содержит визуальный конструктор (вкладка **Структура шины Regul**), редактор параметров, вкладку соотнесения входов/выходов, МЭК-объектов, а также вкладки **Состояние** и **Информация** (данные о производителе, тип, ID, версия и так далее).

Визуальный конструктор шин отображает, какие крейты и каким образом располагаются на шине. На рисунке 47 приведен пример шины с одним крейтом контроллера R500. На рисунках 50 и 51 показаны примеры контроллеров других серий.

Если контроллер применяется в резервированной системе, то в структуре шины, наглядно видно резервированные крейты. Они отображаются в сером цвете и их нельзя редактировать непосредственно. Подробное описание приведено в документе «Конфигурирование резервированной системы на контроллерах серии Regul RX00. Руководство пользователя».

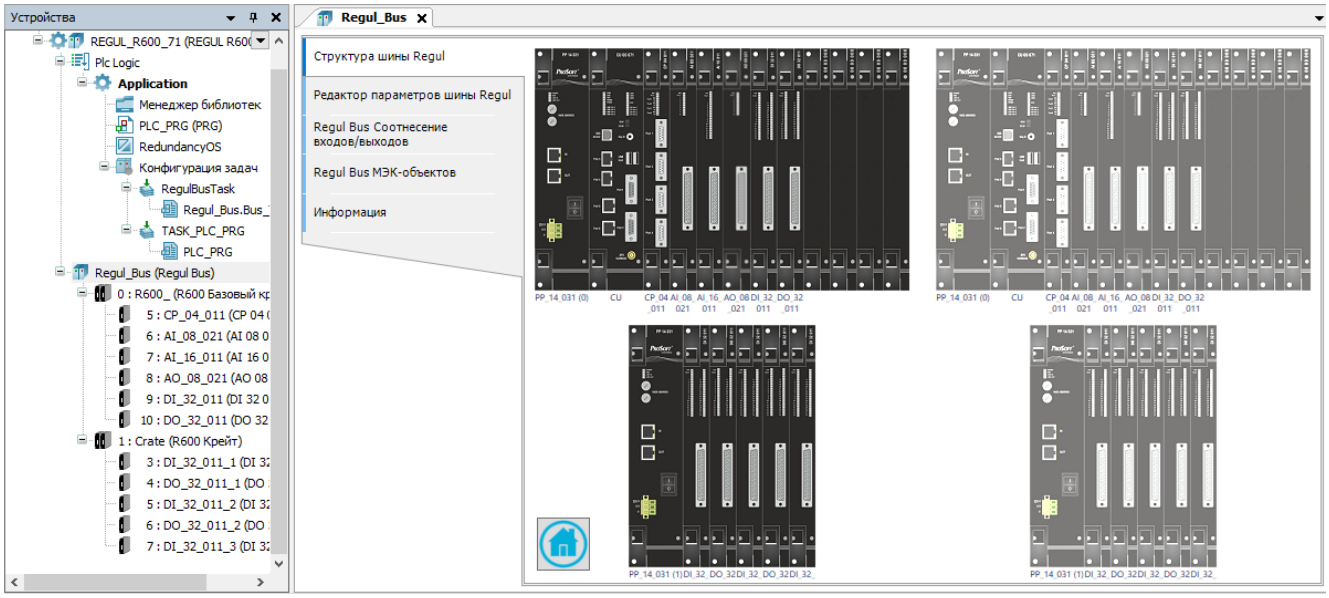


Рисунок 50 – Структура шины контроллера R600 (2 крейта, с полным резервированием)

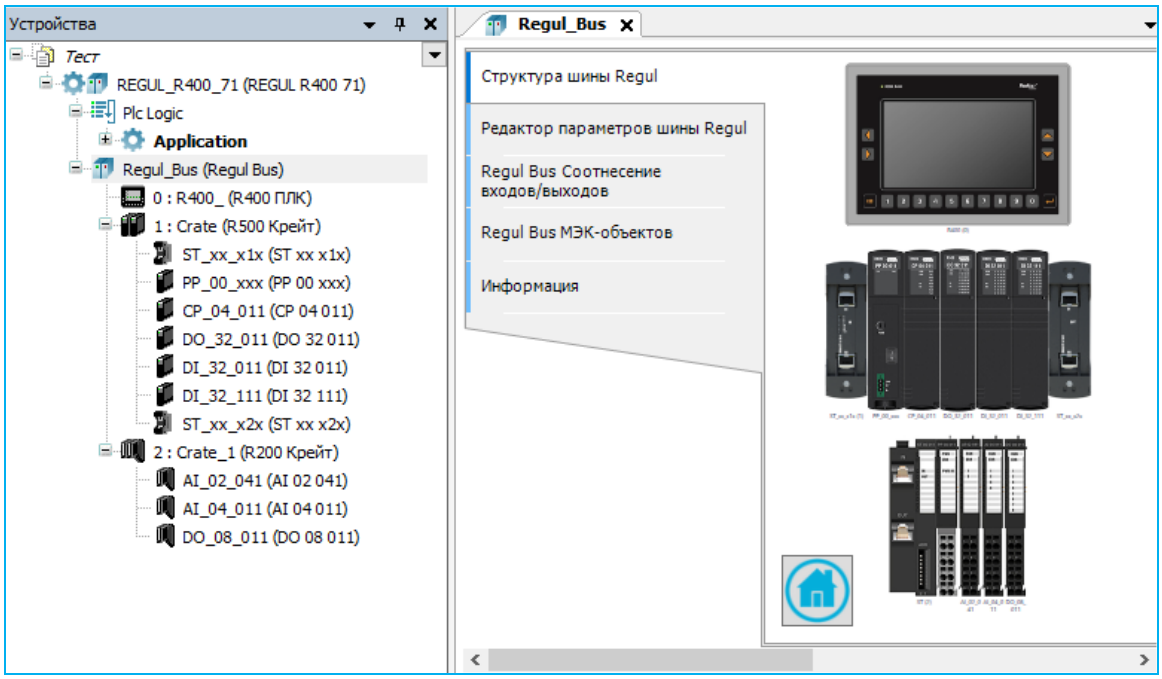


Рисунок 51 – Структура контроллера R400 с модулями R500 и R200

Из визуального конструктора шины удобно перейти в редактор крейта (двойной щелчок по изображению крейта).

Редактор параметров шины для модулей ЦП I/II -го типа

Редактор параметров шин (внутренняя вкладка **Редактор параметров шины Regul** для **RegulBus** – рисунок 52 и для **RegulBus OS** - рисунок 53.

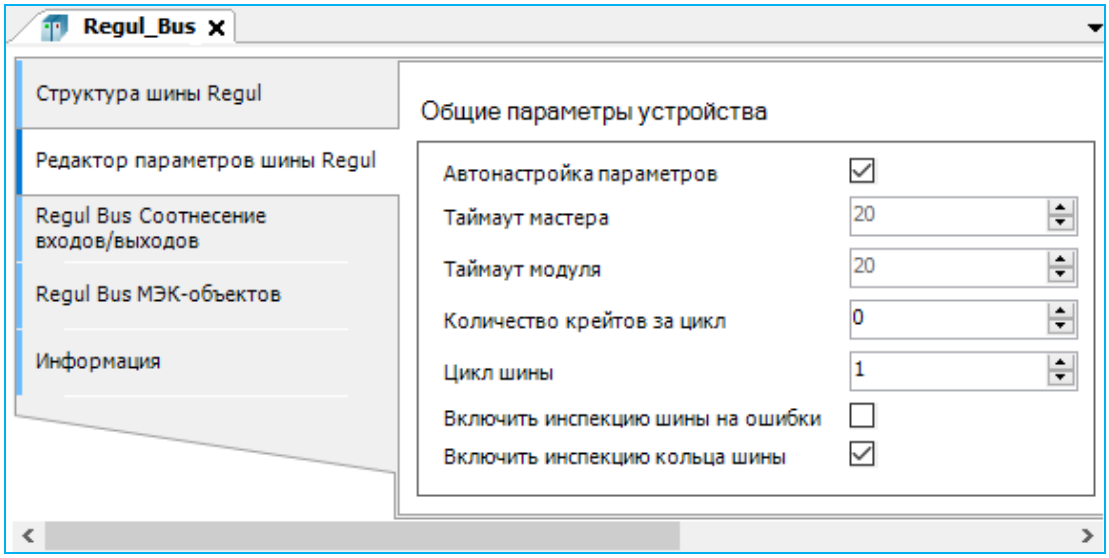


Рисунок 52 – Редактор параметров шины **RegulBus**

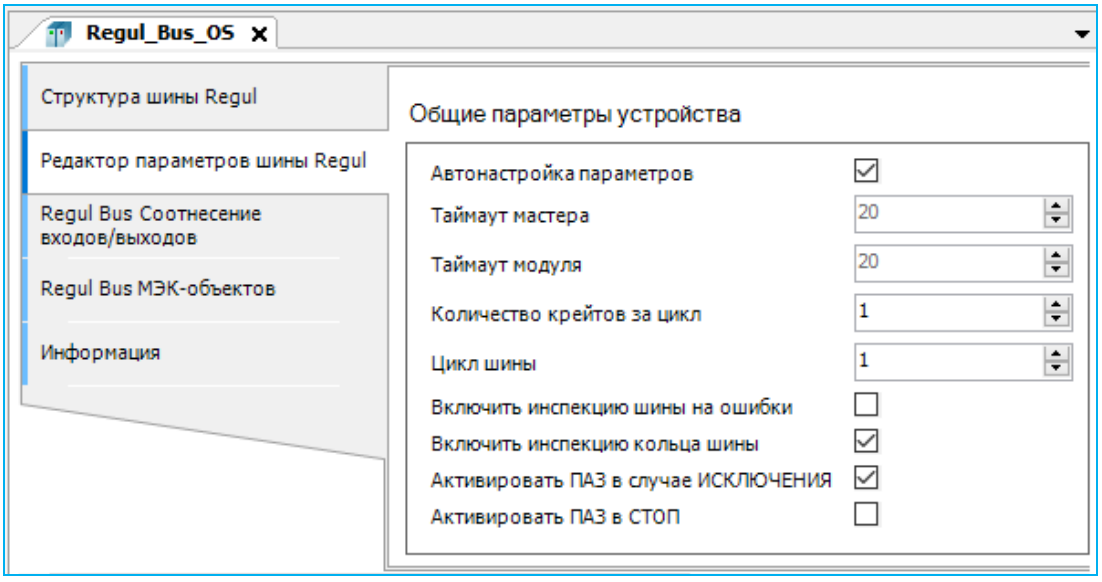


Рисунок 53 – Редактор параметров шины **RegulBus OS**

Редактор позволяет устанавливать следующие значения параметров:

– **Таймаут мастера** – временной интервал, по истечении которого модуль ввода / вывода бракует модуль ЦП (мастер) (максимально возможное – 6553 мс). Значение таймаута мастера определяет максимальный временной интервал, в течение которого модули вывода будут выдавать от ЦП не актуальный сигнал. Фактический временной интервал будет зависеть от того, в какой момент времени выполнения цикла прикладной задачи возникает неисправность. Если неисправность происходит в начале цикла, то период задержки управления будет минимальным (или его вообще не будет). Если же такая

неисправность возникнет в самом конце цикла прикладной задачи, то задержка управления будет максимальная, т.е. равная таймауту мастера.

- **Таймаут модуля** – временной интервал, по истечении которого модуль ЦП (мастер) бракует модуль ввода / вывода. Таймаут модуля, по аналогии с таймаутом мастера, определяет, какой максимальный интервал времени модуль ЦП будет использовать в прикладной программе неактуальное значение сигнала от модуля ввода/вывода, с которым уже потеряна связь. И так же, как и в случае с таймаутом модуля, реальная задержка будет зависеть от момента возникновения неисправности относительно цикла прикладной задачи.
- **Количество крейтов за цикл** – количество крейтов, опрошенных за один цикл шины (по умолчанию установлено значение *1*). Любое целое число, но не более количества крейтов на шине. При установке значения параметра равному «0» все крейты контроллера будут опрашиваться в одном цикле. Нагрузка на процессор напрямую зависит от количества одновременно опрашиваемых крейтов за цикл. Чем больше крейтов в проекте, тем реже опрашивается каждый из них.;
- **Цикл шины** – интервал работы шины, мс (максимальное *50 мс*, для модулей ЦП I-го типа, по умолчанию установлено *1 мс*. Для модулей ЦП II-го типа, по умолчанию установлено *5 мс* – минимально возможное значение, с шагом дискретизации *5 мс*). Разделение по типам модулей ЦП приведено в приложении Б.

ВНИМАНИЕ!

Если в контроллере установлен модуль аналогового ввода R500 AI 08 242/342, то должны быть заданы следующие значения:

- **Количество крейтов за цикл** равным *0*;
- **Цикл шины** равным *1 мс.*

Если выставить значение цикла шины больше, чем *1 мс*, то, при компиляции проекта произойдет автоматическая замена значения на *1 мс.*
В окне сообщений появится предупреждающая информация следующего вида:
«Шина ПЛК содержит модули AI 08 242/342, цикл шины установлен на 1 мс».

Модули аналогового ввода R500 AI 08 242/342 не поддерживают работу с модулями ЦП II-го типа

Установка флажка в поле **Включить инспекцию шины на ошибки** позволяет идентифицировать и фиксировать место ошибки, возникшей на шине, с формированием сообщений в журнал событий и журналированием в лог-файл информации о месте возникновения ошибки.

Включить инспекцию кольца шины позволяет идентифицировать и фиксировать место «обрыва» кольца шины, с формированием сообщений в журнал событий и журналированием в лог-файл информации о месте «обрыва».

ВНИМАНИЕ!

Шина RegulBus OS не поддерживает модули с соответствующей версией СПО (см. «Приложение К»). Модули с версией СПО, меньше указанной в «Приложении К», не проинициализируются на шине RegulBus OS. Для дальнейшей работы потребуется обновить программное обеспечение модулей с помощью специального приложения (см. подраздел «Обновление программного обеспечения модулей ввода/вывода»)

Для шины **RegulBus OS** предусмотрены дополнительные параметры (алгоритм ПАЗ применяется в модулях с каналами вывода аналогового/дискретного сигнала, описание см. в подразделах «Задание параметров алгоритма противоаварийной защиты» на соответствующий модуль):

- **Активировать ПАЗ в случае ИСКЛЮЧЕНИЯ** – при возникновении исключения (отсутствует ответ ППО), шина RegulBus OS позволяет активировать алгоритм ПАЗ (по умолчанию включен);
- **Активировать ПАЗ в СТОП** – при переходе в СТОП (останов приложения), шина RegulBus OS позволяет активировать алгоритм ПАЗ (по умолчанию выключен).

Установка флажка в поле **Автонастройка параметров** блокирует возможность изменять значения параметров **Таймаут мастера**, **Таймаут модуля**. При активированном параметре **Автонастройка параметров** автоматическая подстановка оптимальных настроек таймаут мастера и таймаут модуля для нерезервированного контроллера рассчитывается по следующим формулам:

$$TO\_ms = \max(17+3*CI, Tcmin), \quad (1)$$

$$TO\_mo = TO\_ms \quad (2)$$

где CI – интервал опроса модулей;

TO_mo – таймаут модуля;

TO_ms – таймаут мастера;

Tcmin - минимальное время цикла прикладной задачи.

В свою очередь, вычисление интервала опроса модулей производится по следующей формуле:

$$CI = BI*CN/CPC, \quad (3)$$

где BI (**Цикл шины**) – интервал работы шины, в мс (по умолчанию 1 мс);

CN (**Количество крейтов**) – количество крейтов в контроллере;

CPC (**Количество крейтов за цикл**) – количество крейтов, опрошенных за один цикл (по умолчанию 1).

Полученное значение CI, если оно является дробным, округляется до целого числа в большую сторону.

	**ВНИМАНИЕ!** Значение интервала опроса модулей CI не должно превышать 200 мс. Если значение интервала превысит 200 мс, то, после компиляции проекта, в окне сообщений появится информация об ошибке следующего вида: «Ошибка в структуре шины ПЛК: Параметр CI (205) превысил максимальное значение 200 мс»

Алгоритм расчета таймаутов для резервированного контроллера описан в документе «Конфигурирование резервированной системы на контроллерах серии REGUL RX00. Руководство пользователя».

Вы можете изменить параметр **Цикл шины** (только в сторону увеличения) в случае, если не требуется частого опроса модулей (прикладная задача имеет существенно больший цикл).

Также вы можете изменить параметр **Количество крейтов за цикл**. При увеличении этого параметра опрос всех модулей контроллера произойдёт за меньшее время. Так как увеличение параметра отразится на пиковой загрузке ЦП, то необходимо проконтролировать, чтобы задача RegulBusTask выполнялась за отведенное ей время.

При установке значения параметра **Количество крейтов за цикл** равному «0» все крейты контроллера будут опрашиваться в одном цикле, т.е. CI будет равняться BI.

Вы можете самостоятельно установить требуемые значения таймаута мастера и таймаута модуля.

Следует иметь в виду, что увеличение значения таймаутов уменьшает чувствительность контроллера к внешним воздействиям, приводящим к потерям связи по внутренней шине данных.

Напротив, уменьшение таймаутов ускоряет реакцию контролера на возникновение аппаратной ошибки, что бывает критически важно в ответственных системах с быстрым циклом управления.

Также следует учесть, что время исполнения цикла регулирования зависит от параметра CI. Так, максимальное время исполнения цикла регулирования ПЛК (Рисунок 54) определяется следующей формулой:

$$T_o = t_п+2*CI+t_и+2*\ CI\ +t_у, \quad (4)$$

где $t_п$ (**Время преобразования**) –время преобразования входных сигналов в модуле (данный параметр указан в таблице технических характеристик на модуль, см. в документе на «REGUL RX00.Системное руководство»);

CI (**Интервал опроса модулей**) – рассчитывается по формуле (3);

$t_и$ (**Интервал вызова задачи**) – время исполнения цикла задачи (см. Рисунок 33. Пункт «Конфигурация задач» подраздела «Основные понятия среды разработки»);

$t_у$ (**Время установления**) –время установления выходных сигналов в модуле (данный параметр указан в таблице технических характеристик на модуль, см. в документе на «REGUL RX00.Системное руководство»).

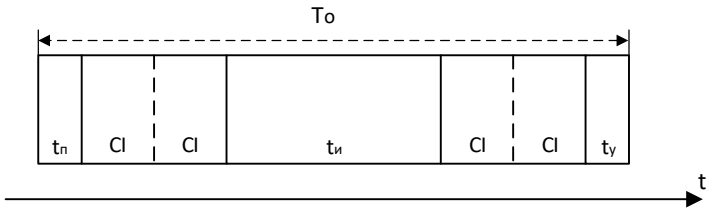


Рисунок 54– Максимальное время исполнения цикла регулирования ПЛК

Редактор параметров шины для модулей ЦП III-го типа

На модулях III-го типа доступна только версия внутренней шины данных RegulBus OS. Для модулей этого типа *интервал опроса модулей* (*CI*) в большинстве случаев не превышает 1 мс. В таблице 5 приведен пример зависимости интервала опроса модулей от объема входных-выходных данных и количество модулей, соответствующее этому объему данных (исходя из среднестатистического соотношения модулей разного типа в составе одного контроллера).

Таблица 5 – Соотношения количества модулей с объемом данных и интервалом опроса

Объем входных-выходных данных, байт	**Количество модулей в контроллере**	**Интервал опроса модулей, мс**
1058	37	0,15946
10533	370	1,064
31587	1110	3,151

Редактор параметров шины представлен на рисунке 55 (внутренняя вкладка **Редактор параметров шины Regul**). Параметры шины настраиваются аналогично описанию, приведенному выше, за исключением значений параметров **Таймаут мастера**, **Таймаут модуля** и параметра **Скорость инициализации модулей.**

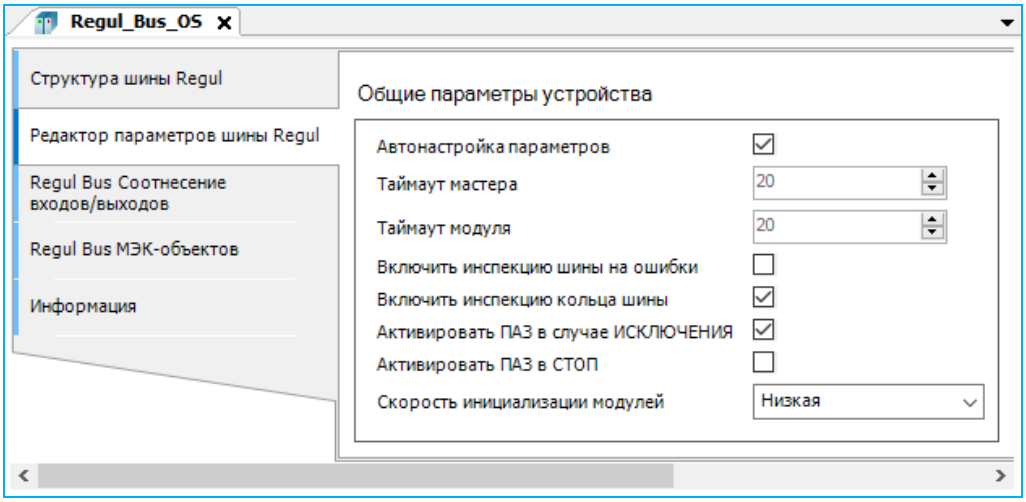


Рисунок 55 – Редактор параметров шины **RegulBus OS** (для III-го типа модулей)

Установка флажка в поле **Автонастройка параметров** блокирует возможность изменять значения параметров **Таймаут мастера**, **Таймаут модуля**. При активированном параметре **Автонастройка параметров** автоматическая подстановка следующих значений:

$$TO\_ms = \max(20, Tcmin), \quad (5)$$

$$TO\_mo = TO\_ms \quad (6)$$

где TO_mo – таймаут модуля;

TO_ms – таймаут мастера;

Tcmin - минимальное время цикла прикладной задачи.

На шине RegulBus OS предусмотрен дополнительный параметр:

– **Скорость инициализации модулей** – позволяет ограничить потребление ресурсов процессора в момент инициализации модулей (по умолчанию ***Низкая***). Возможные значения:
  - ***Минимальная***;
  - ***Низкая***;
  - ***Средняя***;
  - ***Высокая***;
  - ***Максимальная***.

Редактор параметров шины крейта модели R050

Для крейта модели R050 присутствует индивидуальная настройка временных параметров внутренней шины, в зависимости от количества и типа добавленных в конфигурацию модулей. Настроечные параметры шины для каждого крейта приведены в редакторе интерфейсного модуля (Рисунок 56**Ошибка! Источник ссылки не найден.**).

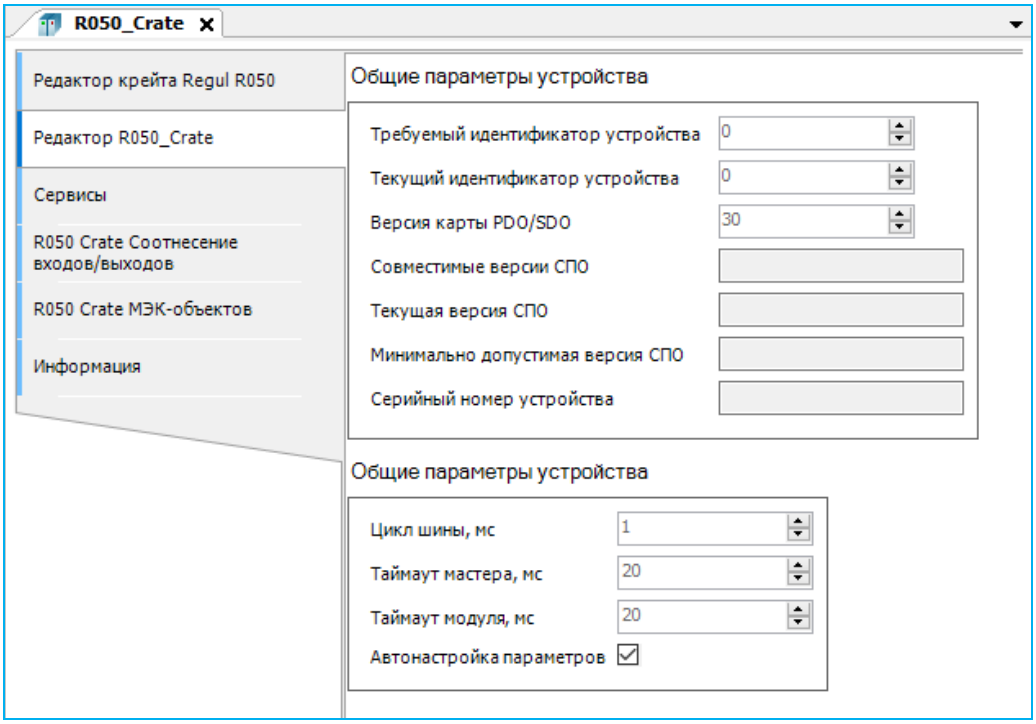


Рисунок 56 - Настроечные параметры внутренней шины крейта R050

Редактор позволяет устанавливать следующие значения параметров:

- **Таймаут мастера** – временной интервал, по истечении которого модуль ввода/вывода бракует модуль ЦП (мастер), в диапазоне от 20 до 10000 мс;
- **Таймаут модуля** – временной интервал, по истечении которого модуль ЦП (мастер) бракует модуль ввода / вывода, в диапазоне от 20 до 10000 мс;
- **Цикл шины** – интервал работы шины, мс (максимальное *1000 мс*, по умолчанию установлено *1 мс).*

Установка флажка в поле **Автонастройка параметров** блокирует возможность изменять значения параметров **Таймаут мастера**, **Таймаут модуля, Цикл шины**, при этом производится автоматическая установка оптимальных настроек в зависимости от количества и типа модулей в крейте.

	**ВНИМАНИЕ!** Изменять параметры, установленные по умолчанию, рекомендуется только в обоснованных случаях и при полной уверенности в работоспособности контроллера с измененными параметрами, так как неверно установленные параметры могут привести к некорректной работе контроллера. При наличии сбоев в работе контроллера необходимо вернуться к параметрам по умолчанию!

Редактор крейта, установка адреса крейта

Откройте редактор крейта (Рисунки 57, 58) одним из двух способов: в дереве устройств двойным щелчком мыши по названию крейта, или в редакторе шины RegulBus двойным щелчком мыши по изображению крейта.

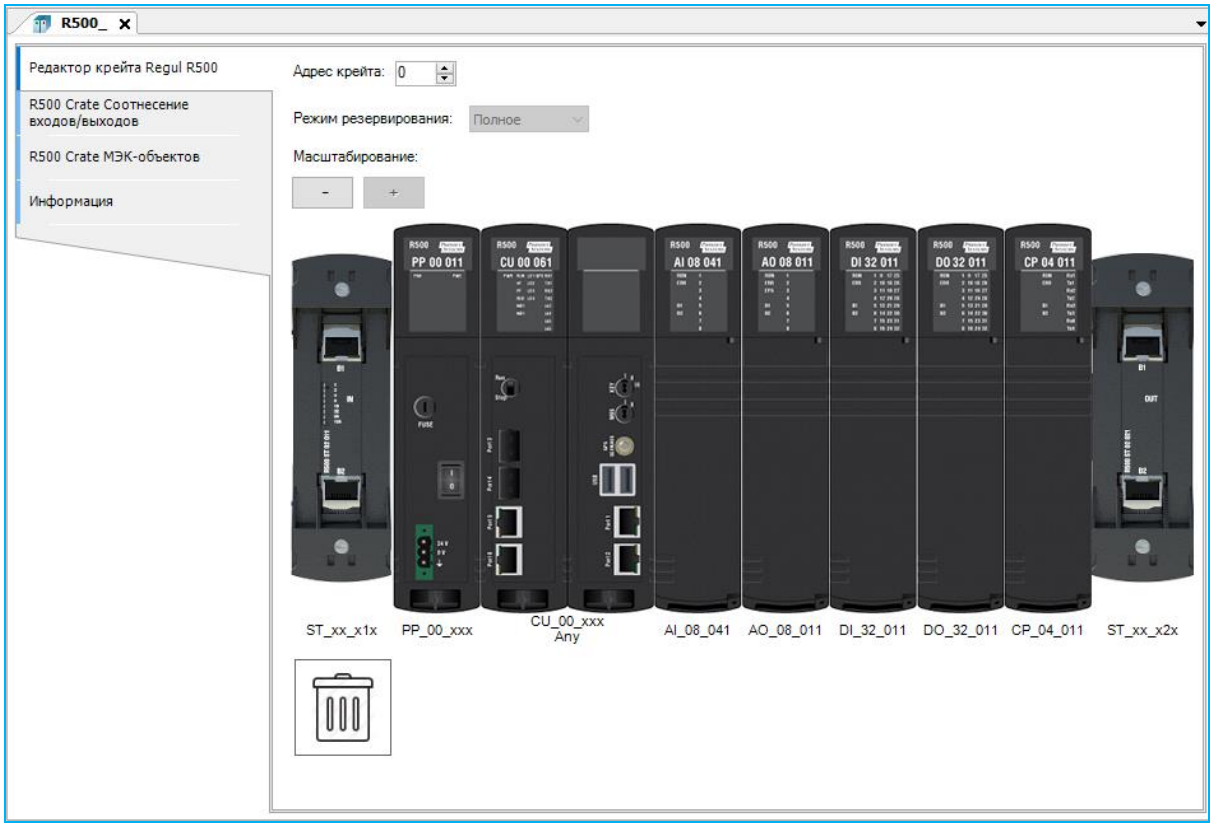


Рисунок 57– Пример редактора крейта (контроллер Regul R500)

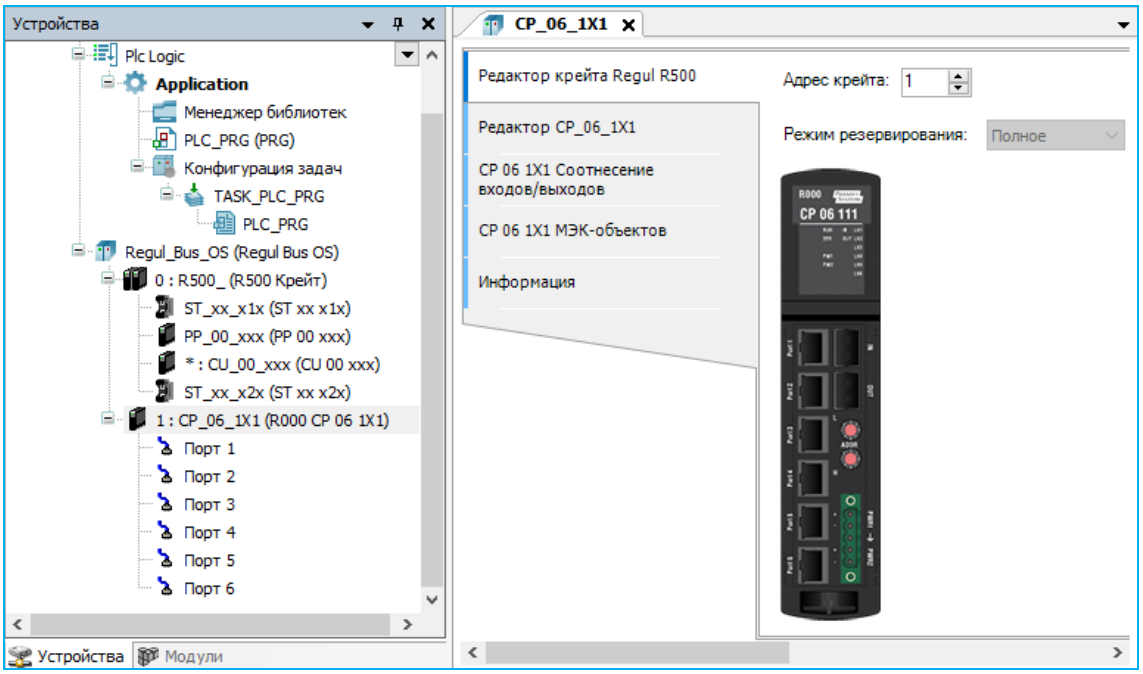


Рисунок 58– Пример редактора Regul R000

В редакторе крейта можно добавлять модули в крейт, менять их местами, удалять из крейта (см. подраздел «Размещение модулей в крейте»).

В поле **Режим резервирования** отображается, какой выбран режим резервирования. Если для контроллера резервирование не предусмотрено, то это поле неактивно. В случае, когда

резервирование есть, в этом поле можно выбрать значение из раскрывающегося списка: *Полное* или *Частичное*. Сам факт наличия/отсутствия резервирования определяется программой автоматически по наличию объекта Redundancy в первом приложении проекта. Подробное описание систем резервирования приведено в документе «Конфигурирование резервированной системы на контроллерах серии Regul RX00. Руководство пользователя».

В поле **Размер крейта** (для Regul R600) выберите значение из раскрывающегося списка: *14 слотов* или *7 слотов*.

Поле **Адрес крейта** в редакторе позволяет задать адрес выбранного крейта в распределенной системе управления.

Каждому крейту, входящему в состав контроллера REGUL RX00, сопоставляется уникальный адрес:

- для Regul R600 этот адрес задается двумя поворотными переключателями на лицевой панели модуля блока питания, соответствующего крейта. Адресные переключатели проградуированы от 1 до F. Можно задать адрес крейта в диапазоне от 00 до FF в шестнадцатеричной системе счисления, при этом нижний адресный переключатель отвечает за младший разряд в значении адреса, а верхний - за старший;
- для Regul R500 этот адрес задается адресным переключателем на передней панели оконечного модуля IN соответствующего крейта. Адресный переключатель имеет в своем составе 8 DIP-ключей. Включение ключа добавляет к значению адреса крейта соответствующую величину (от 1 до 128), указанную рядом с ним;
- для крейта расширения Regul R200 этот адрес задается адресным переключателем на передней панели интерфейсного модуля. Адресный переключатель имеет в своем составе 8 DIP-ключей. Включение ключа добавляет к значению адреса крейта соответствующую величину (от 1 до 128), указанную рядом с ним;
- для крейта расширения Regul R050 этот адрес задается адресным переключателем на передней панели интерфейсного модуля. Адресный переключатель имеет в своем составе 8 DIP-ключей. Включение ключа происходит переводом справа налево (ON→OFF). Включение ключа добавляет к значению адреса крейта соответствующую величину (от 1 до 128), указанную рядом с ним;
- для Regul R000 этот адрес задается двумя поворотными переключателями ADDR(L/H) на лицевой панели модуля. Переключатели проградуированы 0|2|4|6|8|A|C|E|. Можно задать адрес в диапазоне от 00 до FF в шестнадцатеричной системе счисления, при этом верхний адресный переключатель отвечает за младший (L) разряд в значении адреса, а нижний - за старший (H).

Диапазон допустимых адресов от 0 до 255. Адрес крейта можно задавать произвольно, не ориентируясь на физический порядок соединений крейтов между собой, но он обязательно

должен совпадать с адресом, присвоенным данному крейту в среде разработки Astra.IDE. Адрес крейта отображается в дереве устройств перед именем крейта.

Для Regul R400 и базового крейта Regul R200 не предусмотрен аппаратный задатчик адреса. Для них всегда зарезервирован неизменяемый адрес 0.

Вкладка соотнесение входов/выходов шины и крейта

В редакторах шины и крейта, на вкладке **Соотнесение входов/выходов,** присутствует параметр HwError, информирующий о статусе состояния шины RegulBus (RegulBusOS) и крейта соответственно (Рисунок 59).

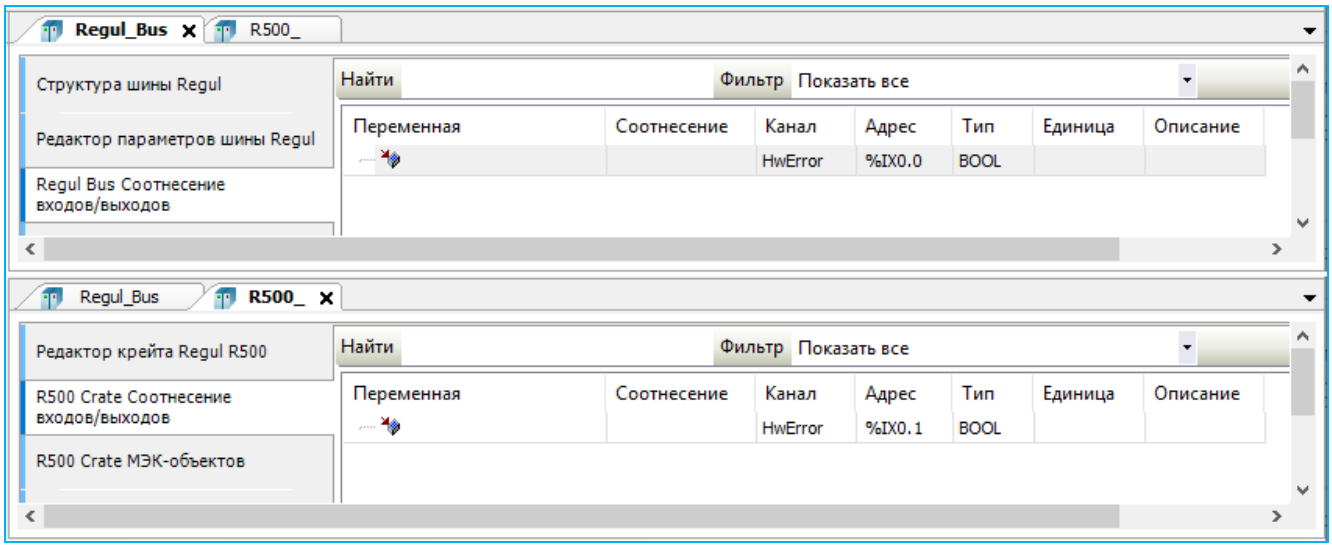


Рисунок 59– Пример вкладок соотнесение входов/выходов шины RegulBus и Crate

Данный параметр доступен для конфигурирования и привязке к переменной прикладной программы (см. подраздел «Привязка каналов к переменным программы»).

Значение параметра отображается в онлайн-режиме. Параметр принимает значение *TRUE* (истина) или *FALSE* (ложь), т.е. при наличии ошибки отобразится: *TRUE*, а при отсутствии *FALSE* (Рисунок 60).

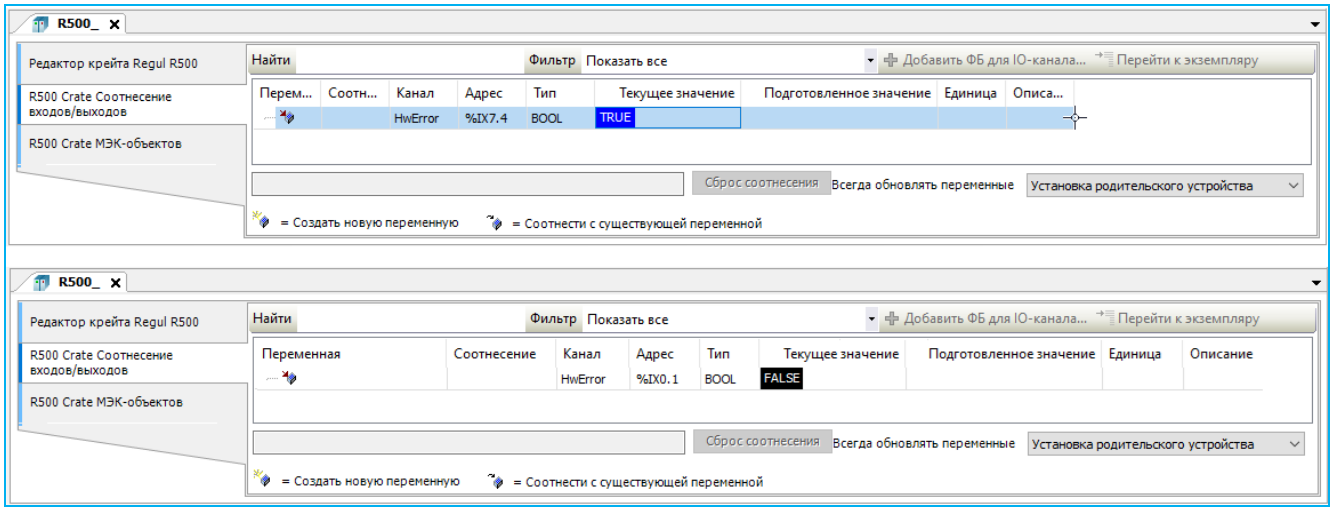


Рисунок 60– Пример отображения наличия и отсутствия ошибки в крейте

Также, могут отображаться биты состояния соединения для коммуникационных портов с интерфейсом Ethernet и коммуникационных портов IN и OUT (для организации связи по шине), ниже приведены примеры на рисунках: 61 (крейт расширения) и 62 (базовый крейт).

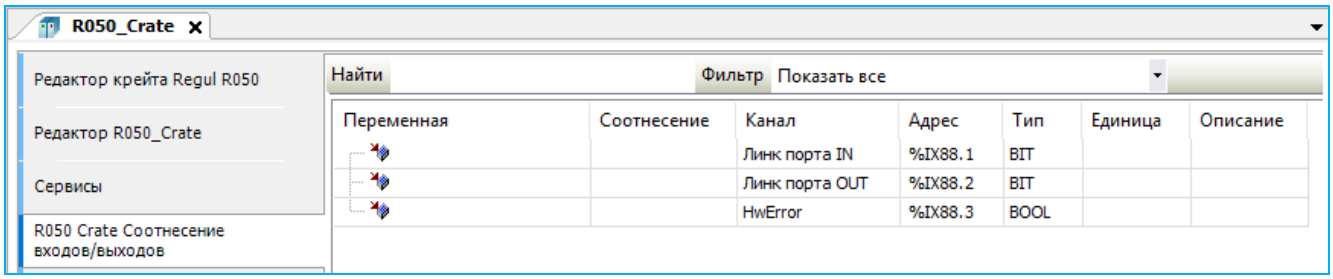


Рисунок 61 – Пример вкладки соотнесение входов/выходов шины R050 Crate

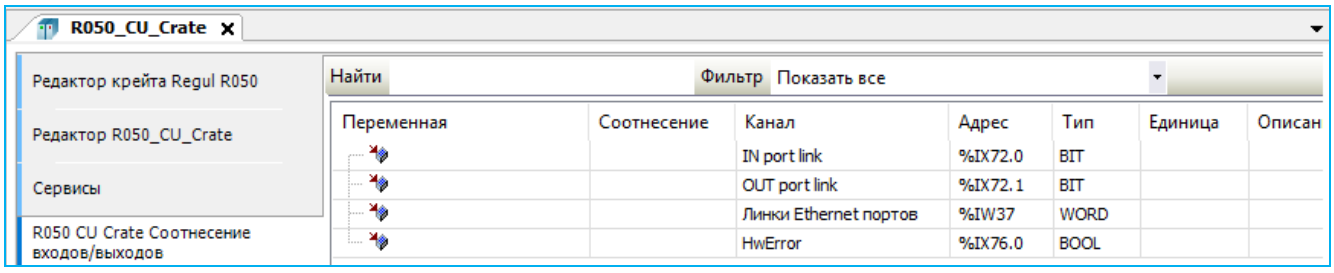


Рисунок 62 – Пример вкладки соотнесение входов/выходов шины R050 CU Crate

Размещение модулей в крейте

Добавление модуля в крейт

Существует два варианта добавления модулей в крейт: через дерево устройств или с помощью редактора крейта. Добавление через редактор крейта описано ниже для каждой серии контроллера.

Для добавления модулей в крейт через дерево устройств: в дереве устройств поместите курсор на название крейта, нажмите правую кнопку мыши. В появившемся контекстном меню выберите пункт **Добавить устройство…** (Рисунок 63).

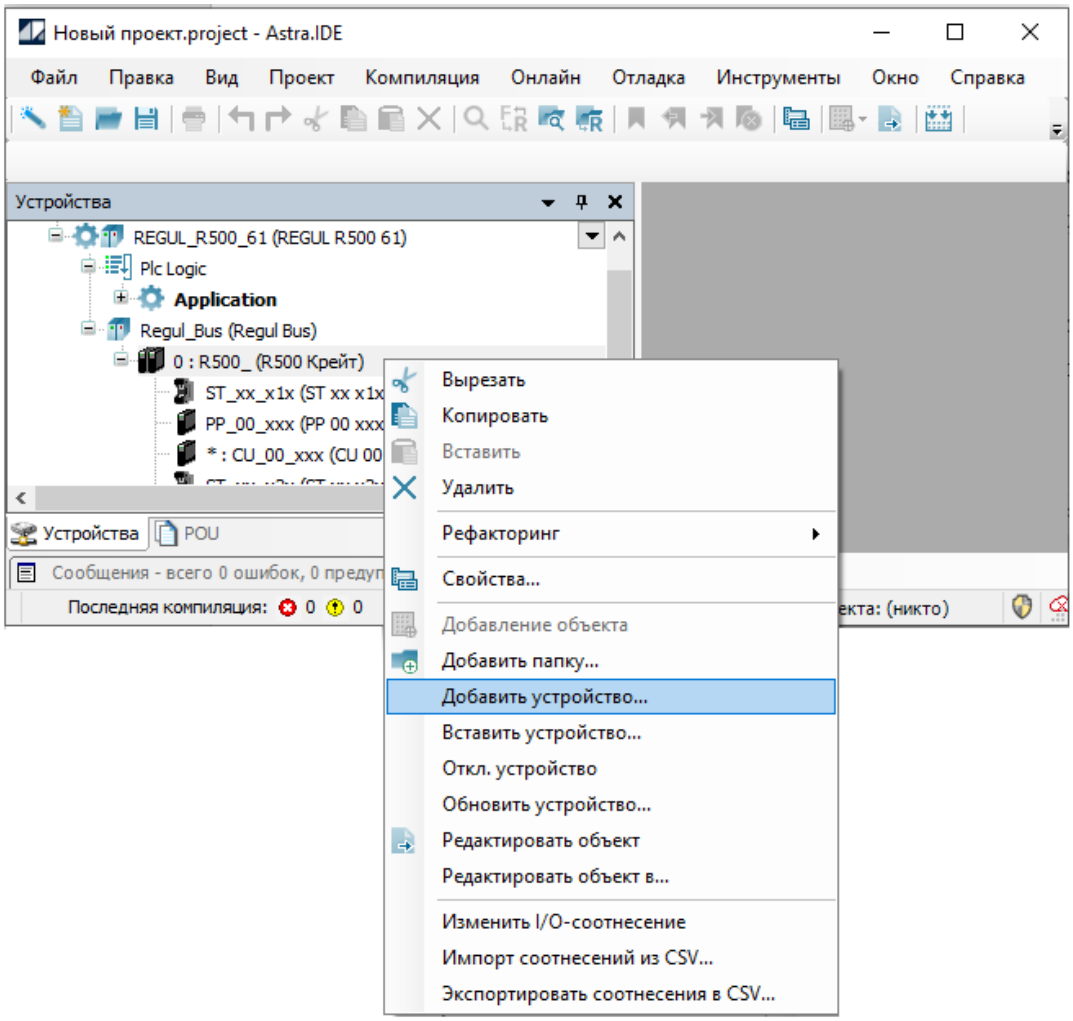


Рисунок 63– Добавление модулей в крейт через дерево устройств

Откроется окно с перечнем модулей для выбора (Рисунок 64).

По умолчанию установлен флажок в поле **Группировать по категориям**. Поэтому для выбора модуля раскрывайте список с помощью кнопки .

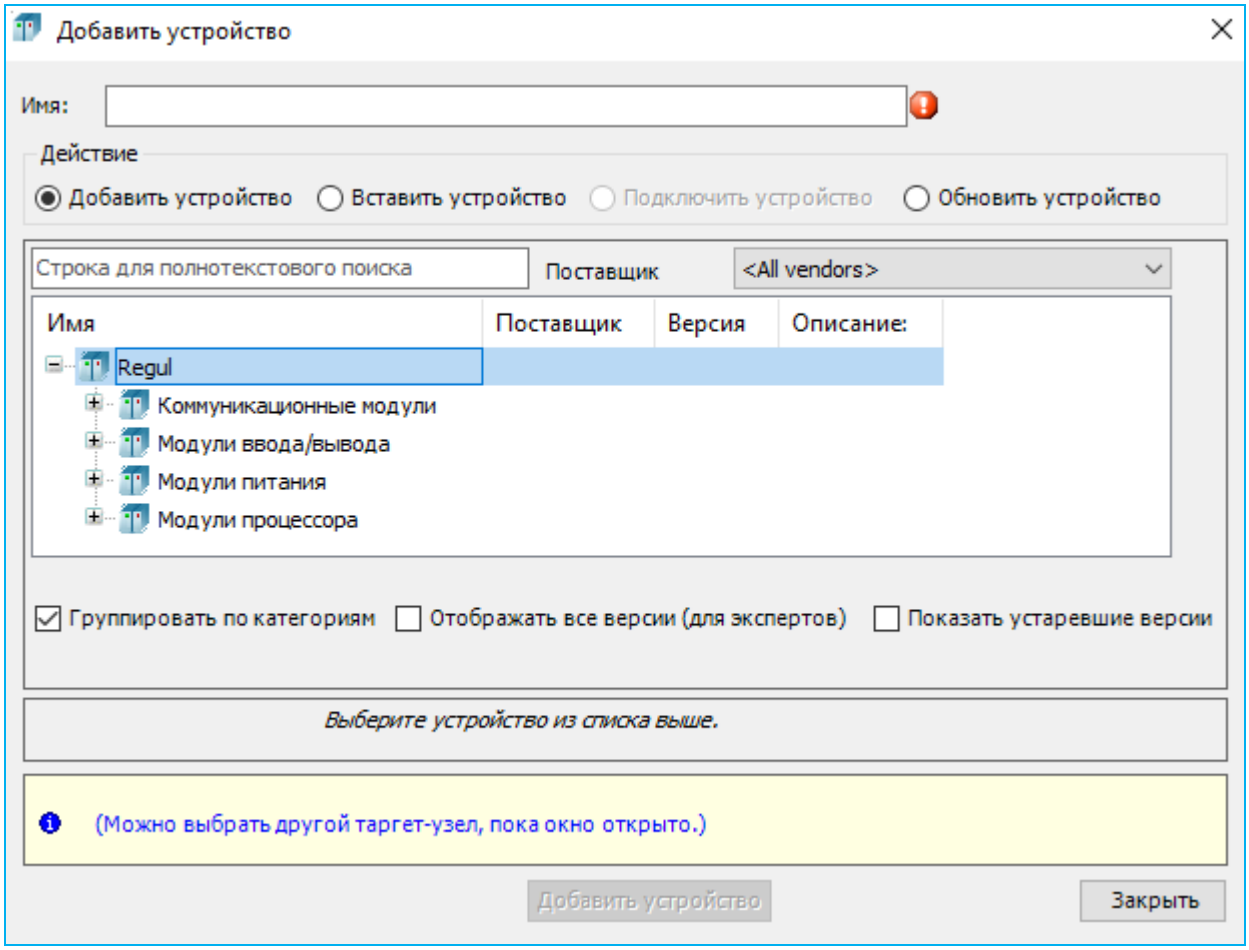


Рисунок 64 – Добавление модулей в крейт контроллера

Если снять флажок в поле **Группировать по категориям**, то список модулей принимает следующий вид (Рисунок 65).

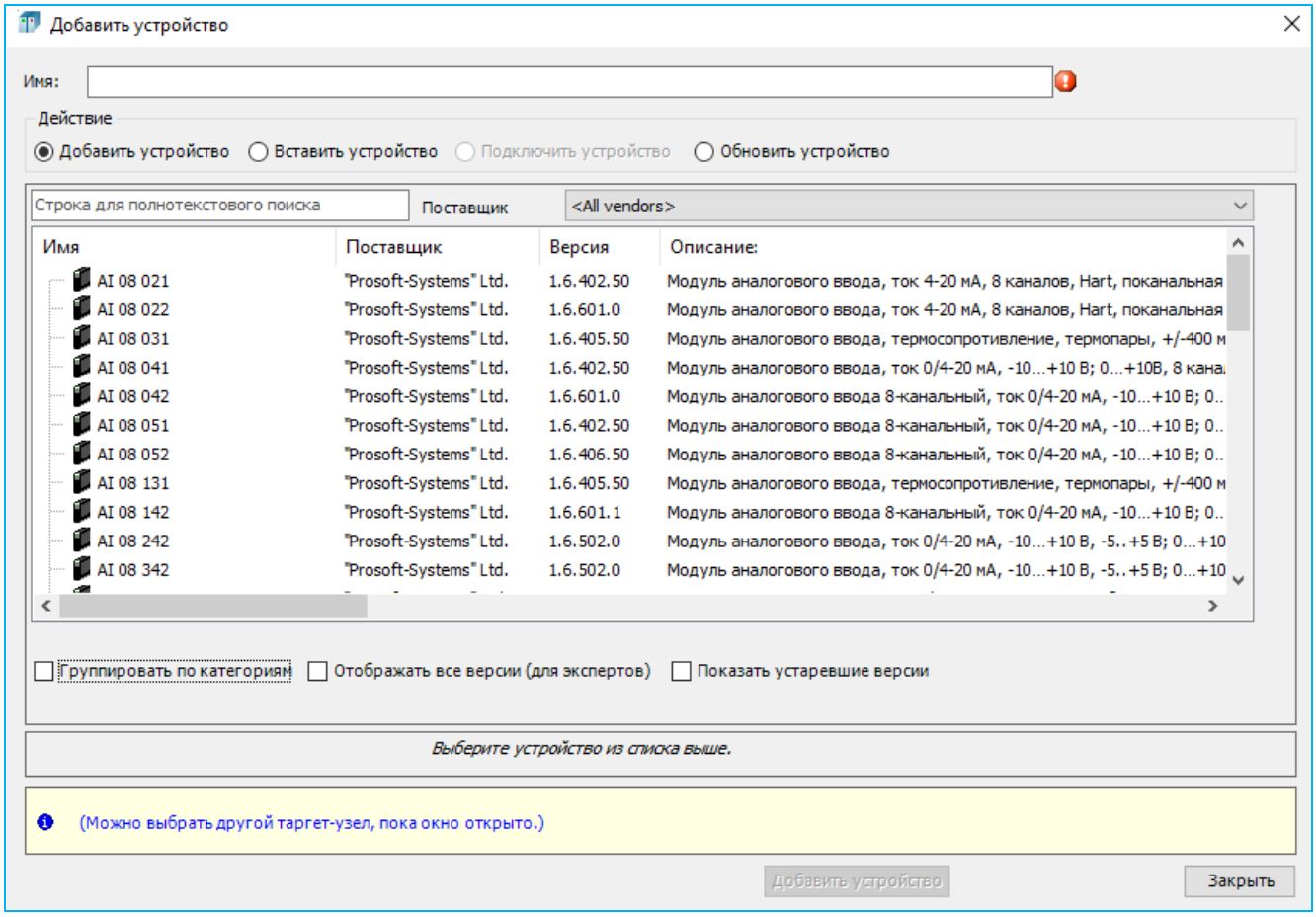


Рисунок 65 – Список модулей для добавления в крейт

Для каждого модуля указано его имя, производитель и номер версии ПО.

Предусмотрена возможность использовать в одном проекте не только разные модули, но и модули одного типа с разными версиями ПО. Для корректной работы модулей необходимо, чтобы версия ПО модуля совпадала с версией ПО из файла описания в пакете Astra.IDE.

Установите флажок в поле **Отображать все версии (для экспертов)**. В списке будут отображены все модули всех возможных совместимых версий (Рисунок 66).

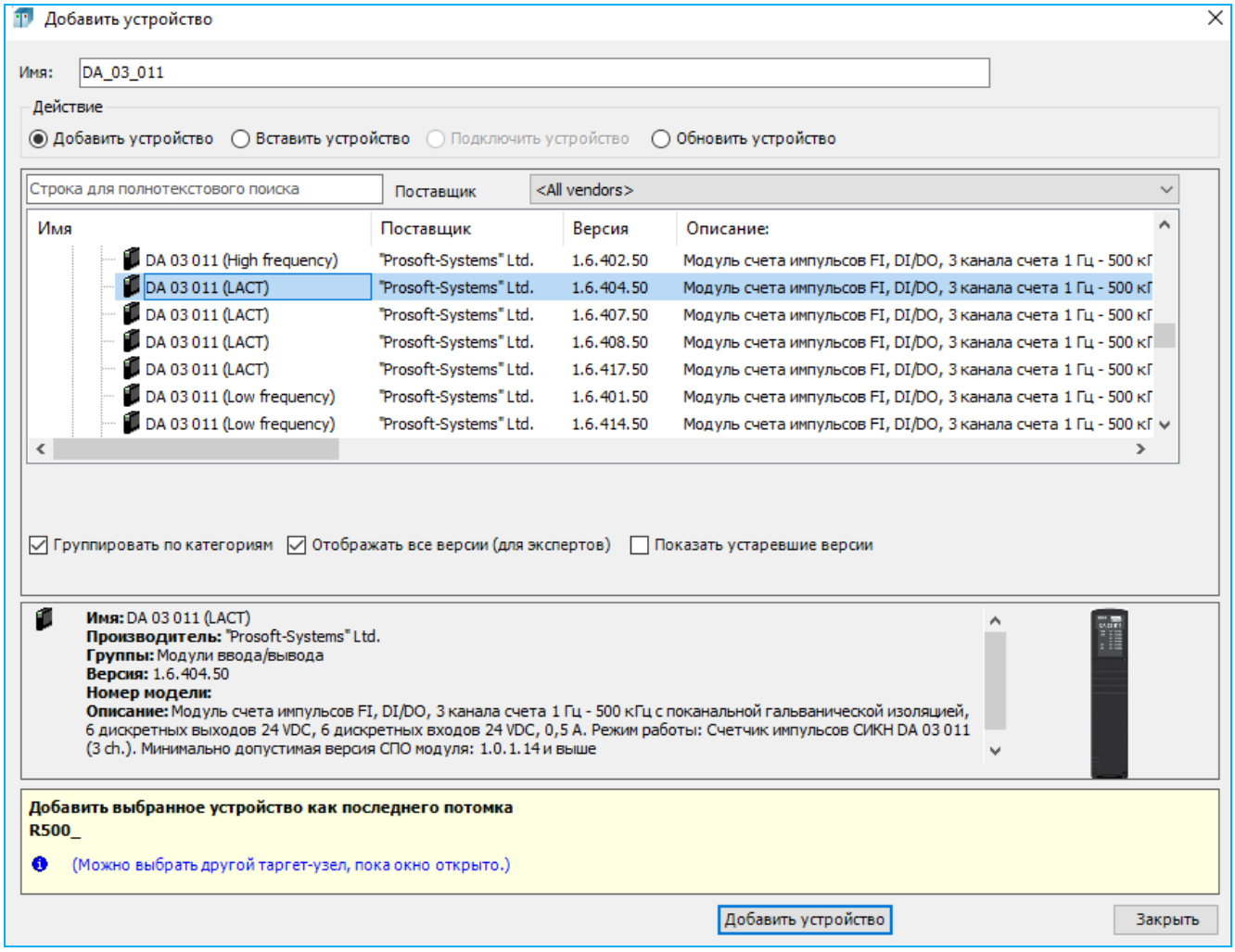


Рисунок 66 – Список модулей всех версий

Выберите нужный модуль, обращая внимание на версию ПО, дважды щелкните мышью по названию модуля или в нижней части окна нажмите кнопку ***Добавить устройство***. Модуль будет добавлен в крейт, выбранный в дереве проекта.

Подробное описание совместимости версий ПО модулей приведено в пункте «Об обратной совместимости» подраздела «Обновление ПО контроллера».

Особенности размещения модулей в крейте контроллера Regul R600

Редактор крейта контроллера Regul R600 имеет следующий вид (Рисунок 67).

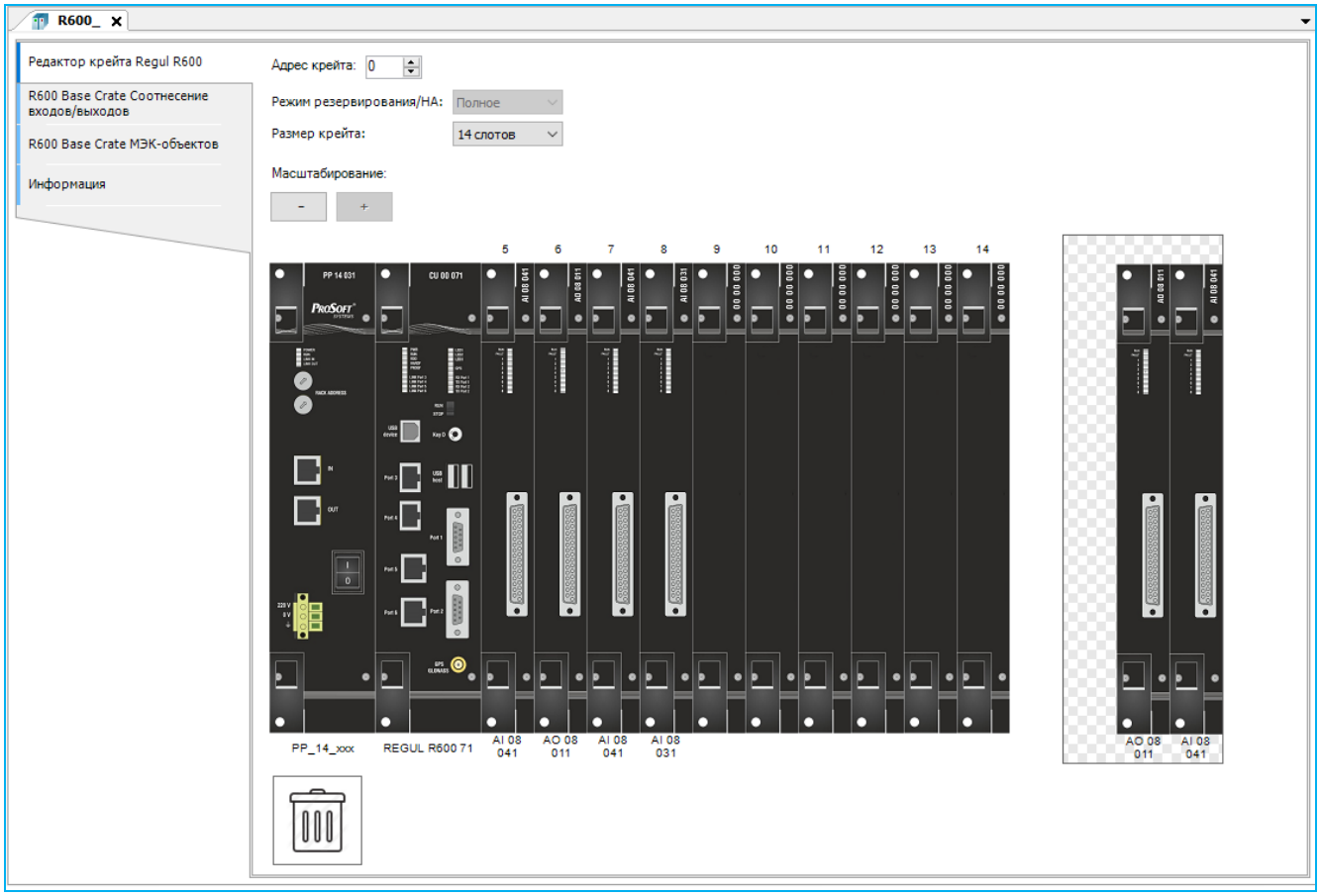


Рисунок 67 – Пример редактора крейта контроллера Regul R600

В левой части редактора отображается новый крейт с пустыми слотами, то есть посадочными местами для модулей. В правой части – рабочая область, куда временно помещаются выбранные модули. В первый слот всегда устанавливается модуль источника питания. Так как модули источника питания и центрального процессора занимают два слота, то, говоря про номер слота установки, подразумевается номер левого из двух слотов, занимаемых модулем. В последующие слоты в зависимости от функциональности крейта можно устанавливать модули любого типа в следующем порядке:

– если в крейте один источник питания (PP) и один модуль центрального процессора (CU), то модуль PP занимает слот 1, а модуль CU должен занимать слот 3;
– если в крейте два источника питания и два модуля центрального процессора (CU), то первый модуль PP занимает слот 1, второй модуль PP – слот 3, модули CU должны занимать слот 5 и слот 7;
– остальные слоты могут быть заняты модулями ввода/вывода и модулями коммуникационного процессора в любом порядке.

Добавление модуля через редактор крейта: щелкните левой кнопкой мыши по пустому слоту. Откроется окно **Добавить устройство** (Рисунок 64).

Выберите нужный модуль. Двойной щелчок левой кнопкой мыши добавит этот модуль в крейт. Если повторно щелкнуть дважды, то этот модуль будет добавлен в рабочую область. Можно, не закрывая окна, в рабочую область добавлять другие модули. Как только в ней закончится место, в окне **Добавить устройство** автоматически будет предложено выбрать новый крейт, а не модуль.

Перемещение модулей из рабочей области в крейт выполняется перетаскиванием. Также перетаскиванием можно менять модули местами или удалять (переносить в корзину).

Все модули, добавленные в редактор крейта, отображаются в дереве устройств: черным цветом – модули в составе крейта, бледно-серым – модули в рабочей области.

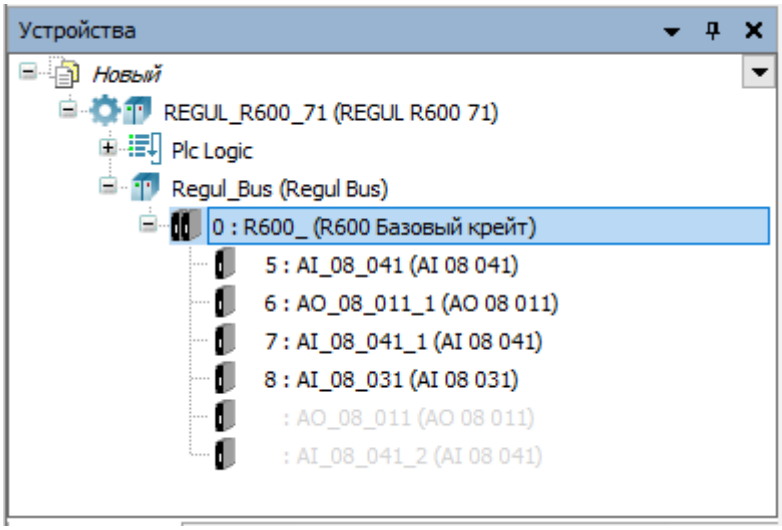


Рисунок 68 – Модули в крейте – отображение в дереве устройств

Особенности размещения модулей в крейте контроллера Regul R500

В начале работы в редакторе контроллера Regul R500 отображается крейт с оконечными модулями, модулем источника питания и, если крейт базовый, модулем центрального процессора (Рисунок 69). По мере добавления в контроллер крейтов, а в крейты – модулей, они отображаются в дереве устройств.

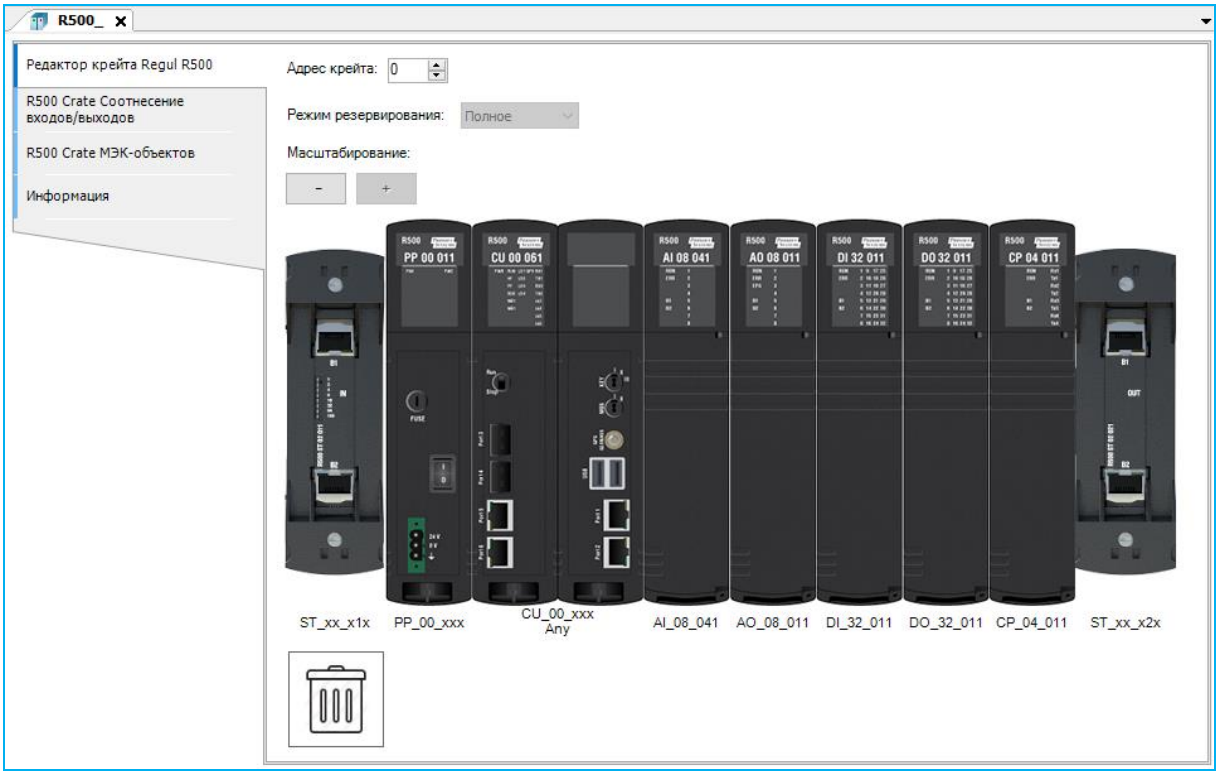


Рисунок 69 – Редактор крейта контроллера Regul R500

Аппаратные и программные решения, реализованные в контроллере, не накладывают ограничения на размещения модулей как в рамках одного крейта, так и в рамках нескольких крейтов, входящих в один контроллер, за исключением следующих правил:

- с обеих сторон крейта должны быть установлены оконечные модули: слева от крейта - оконечный модуль IN, справа – оконечный модуль OUT;
- в составе одного крейта количество модулей разного типа (исключая оконечные модули) не должно превышать 40 штук;
- в случае установки дополнительных модулей источников питания следуют их распределить по крейту;
- в контроллере может присутствовать не более двух модулей центрального процессора, которые физически работают на разных аппаратных шинах, что также должно быть указано в виртуальном редакторе (в параметрах модуля ЦП, раздел «Настройка параметров модулей»);

ИНФОРМАЦИЯ

На модулях ЦП I/II типа реализована защита от подключения двух модулей ЦП на одну шину. Защита сработает на модуле ЦП2, при его установке в работающий контроллер с идентичным положением ключа MBS на модуле ЦП1. Она исключает отказ работающего ЦП1 контроллера, в случае конфликта ключа MBS. При подключении средой разработки к модулю ЦП2, система предупреждает, что находится в несогласованном состоянии и записью сообщений в файл sloginfo.log (путь …logs/logger/user/sloginfo.log):
[E] 11.0 rbutil enable_bus:regul bus 1 is busy
[I] 11.0 rbutil set_ecatbus_activity:regul bus 1 is disabled

– для модулей ЦП III-го типа можно установить блоки расширения EU, чтобы организовать четыре дополнительных независимых каналов связи по интерфейсу Ethernet. Блоки расширения устанавливаются справа от модуля ЦП и не более двух штук на один ЦП (Рисунок 70).

ИНФОРМАЦИЯ

Добавить блок расширения EU к модулям ЦП III-го типа можно только через дерево устройств. В крейте поместите курсор на название модуля ЦП (**CU 00 xxx)**, нажмите правую кнопку мыши. В появившемся контекстном меню выберите пункт **Добавить устройство…** Откроется окно **Добавить устройство** ⇨Выберите блок расширения, дважды щелкните мышью по названию или в нижней части окна нажмите кнопку **Добавить устройство**. Блок расширения EU будет добавлен к модулю ЦП в крейт

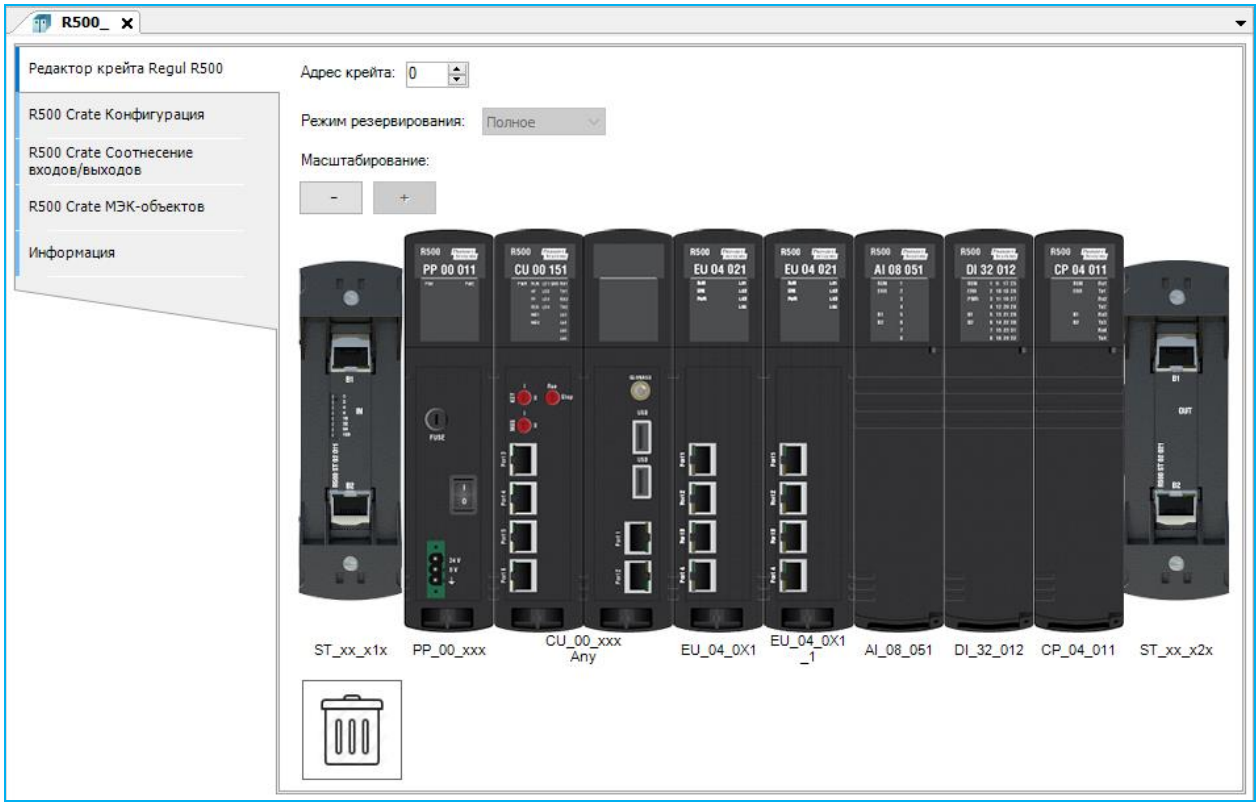


Рисунок 70 – Редактор крейта контроллера Regul R500 с блоком расширения EU

Для добавления модуля в крейт наведите курсор на имеющиеся модули, зеленым цветом подсвечивается область, куда будет вставлен новый модуль. Нажмите левую кнопку мыши. Откроется окно **Вставить устройство**, аналогичное окну **Добавить устройство**. Выберите нужный модуль. Двойной щелчок левой кнопкой мыши добавит этот модуль в крейт. Перетаскиванием можно менять модули местами или удалять (переносить в корзину).

Особенности размещения модулей в крейте контроллера Regul R200

Редактор контроллера Regul R200 выглядит следующим образом (Рисунок 71).

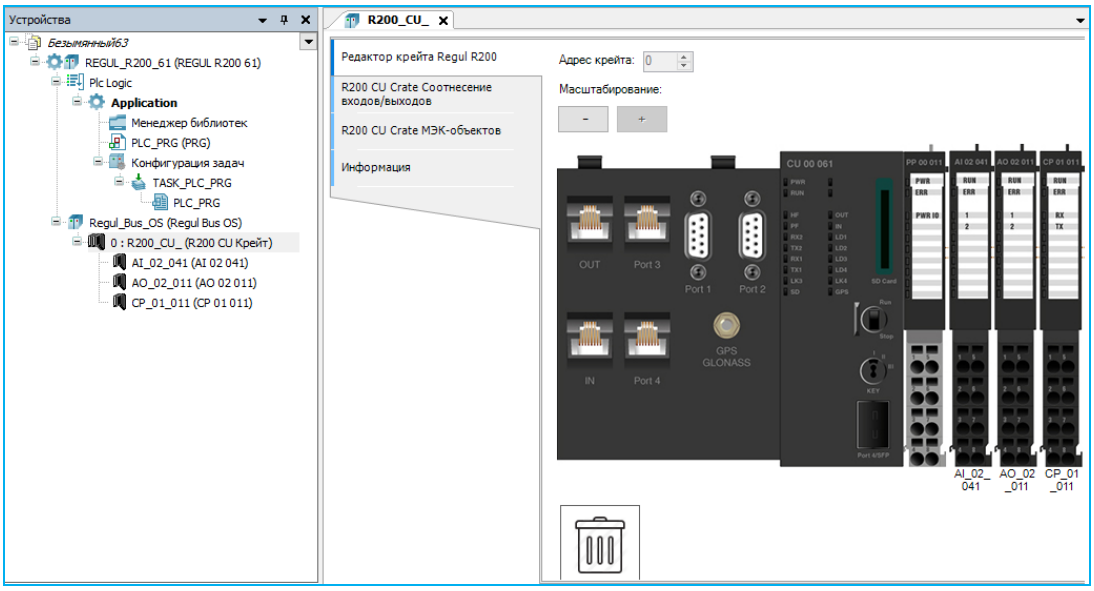


Рисунок 71 – Редактор крейта контроллера Regul R200

Крейт в обязательном порядке должен содержать в своем составе один (и только один) модуль центрального процессора (в случае базового крейта) или интерфейсный модуль (в случае крейта расширения), который устанавливается в крайнее левое положение. С правой стороны к нему подсоединяются остальные модули (ввода/вывода, коммуникационного процессора) в свободном порядке. В один крейт можно установить до 70 модулей различного типа: ввода/вывода, коммуникационного процессора или источника питания.

Модули источника питания физически добавляются в крейт согласно правилам, прописанным в документе «Regul R200. Системное руководство». В программе Astra.IDE добавление этих модулей в проект не предусмотрено и не требуется.

Для добавления модуля в крейт наведите курсор на имеющиеся модули, зеленым цветом подсвечивается область, куда будет вставлен новый модуль. Нажмите левую кнопку мыши. Откроется окно **Добавить устройство**. Выберите нужный модуль. Двойной щелчок левой кнопкой мыши добавит этот модуль в крейт. Перетаскиванием можно менять модули местами или удалять (переносить в корзину).

ИНФОРМАЦИЯ

Если в проекте присутствует крейт без модулей ввода/вывода, он будет отображаться со значком ⚠. Данное состояние считается рабочим и является исключением из правил, описанных в подразделе «Устройство, дерево устройств»

Особенности размещения модулей в крейте контроллера Regul R050

Редактор контроллера Regul R050 выглядит следующим образом (Рисунок 72).

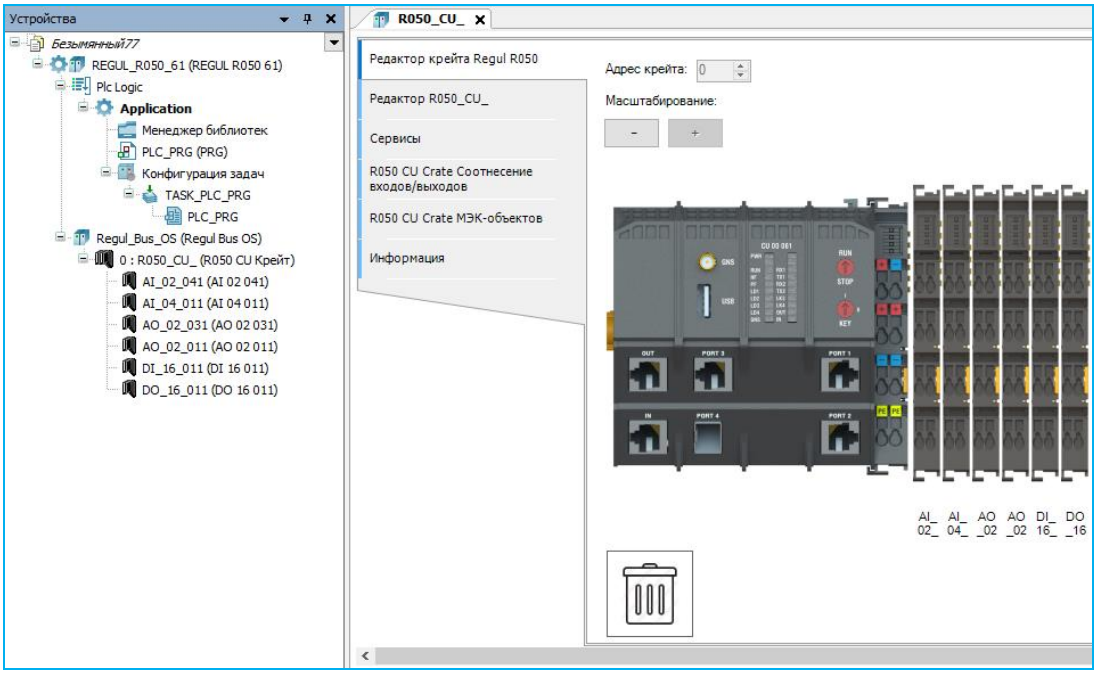


Рисунок 72 – Редактор крейта контроллера Regul R050

Крейт в обязательном порядке должен содержать в своем составе один (и только один) модуль центрального процессора (в случае базового крейта) или интерфейсный модуль (в случае крейта расширения), который устанавливается в крайнее левое положение. С правой стороны к нему подсоединяются остальные модули (ввода/вывода, коммуникационного процессора) в свободном порядке. В один крейт можно установить до 70 модулей ввода/вывода.

Для добавления модуля в крейт наведите курсор на имеющиеся модули, зеленым цветом подсвечивается область, куда будет вставлен новый модуль. Нажмите левую кнопку мыши. Откроется окно **Добавить устройство**. Выберите нужный модуль. Двойной щелчок левой кнопкой мыши добавит этот модуль в крейт. Перетаскиванием можно менять модули местами или удалять (переносить в корзину). Выберите модуль и с зажатой левой клавишей мыши перетащите его.

Особенности размещения модулей контроллера Regul R400

Добавление модулей контроллера Regul R400 (Рисунок 73) происходит через дерево устройств – нужно добавить крейт R200, R500 или R600. Размещение модулей в крейты этих серий по правилам, описанным выше.

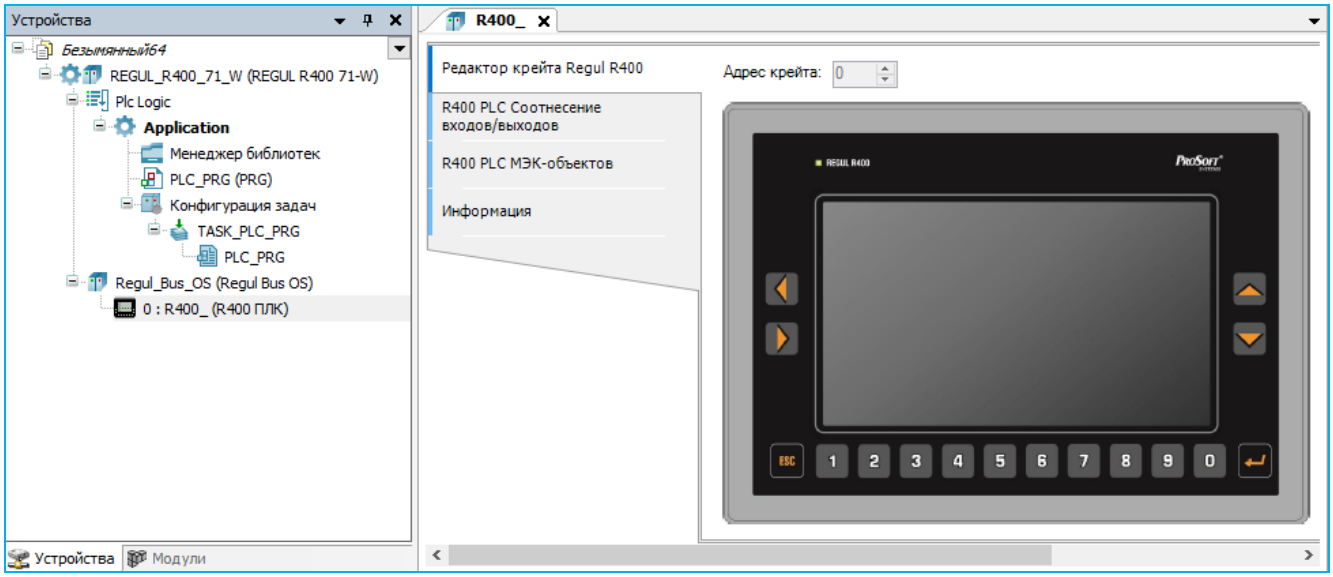


Рисунок 73 – Редактор крейта контроллера Regul R400

Добавление устройств к модулям

При добавлении HART-устройств к модулям аналогового ввода/вывода и использовании интерфейсных портов RS-485 на модулях ЦП и коммуникационного процессора (CP 0x 011), необходимо учитывать общее количество последовательных портов и Hart-мастеров, добавленных в конфигурацию контроллера, а именно, всего не более 1000 (ExtSerialPort протокол Modbus и IEC 60870-5-101).

Более подробное описание работы приведено в документах «Настройка обмена данными по протоколу HART на контроллерах серии REGUL RX00. Руководство пользователя», «Настройка обмена данными по протоколу Modbus на контроллерах серии REGUL RX00. Руководство пользователя», «Настройка обмена данными по протоколу IEC 60870-5-101/IEC 60870-5-104 на контроллерах серии REGUL RX00. Руководство пользователя», «Настройка обмена данными по протоколу PROFIBUS DP на контроллерах серии REGUL RX00. Руководство пользователя», «Настройка обмена данными по протоколу FIELDBUS FOUNDATION H1 на контроллерах серии REGUL RX00. Руководство пользователя».

.

Резервированные сборки модулей ввода/вывода контроллера Regul R500

ИНФОРМАЦИЯ

Начиная с версии СПО 1.7.1.0, реализована поддержка резервированных сборок модулей ввода/вывода R500 с версией СПО 1.0.30.0 на шине RegulBus OS. Поддержка осуществлена для всех модулей (AI, DI, AO, DO, AS, DS), за исключением:

- модулей R500 AI 08 142, R500 AI 08 342, R500 DI 16 032;
- на модулях R500 DO 32 041 работает только для каналов в режиме «верхний ключ» и без режима широтно-импульсной модуляции (ШИМ) выходного сигнала на канале;
- на модулях R500 AO XX 0X1 и R500 AS 08 011 работает только для каналов в режиме по току и без поддержки HART.

Начиная с версии СПО 1.7.2.0, реализована поддержка резервированной сборки модуля DA (только в режиме High frequency) с версией СПО 1.0.31.0.

Примеры схем подключения приведены в документе «Regul R500. Системное руководство», раздел «Приложение В. Схемы подключения полевых устройств к резервированным каналам»

Резервированная сборка представляет собой набор из двух или трех модулей одного типа, которая предназначена для объединения нескольких физических каналов обработки сигнала в одной переменной.

Одной входной или выходной переменной могут соответствовать два или три физических канала. Например, у резервированной сборки из трех модулей аналогового ввода для обслуживания одной переменной используются три физических канала. Такая избыточность позволяет исключить ложный останов управляемого оборудования, при отказе одного из элементов резервированной подсистемы. При отказе одного элемента система деградирует, но продолжает работать, сигнализируя об отказе резервированного компонента.

Создание резервированной сборки

В дереве устройств поместите курсор на название модуля, который необходимо резервировать и нажмите правую кнопку мыши. В появившемся контекстном меню выберите пункт **Создать резервированную сборку** ▸, далее количество модулей в сборке: *Из двух* либо *Из трех* модулей (Рисунок 74).

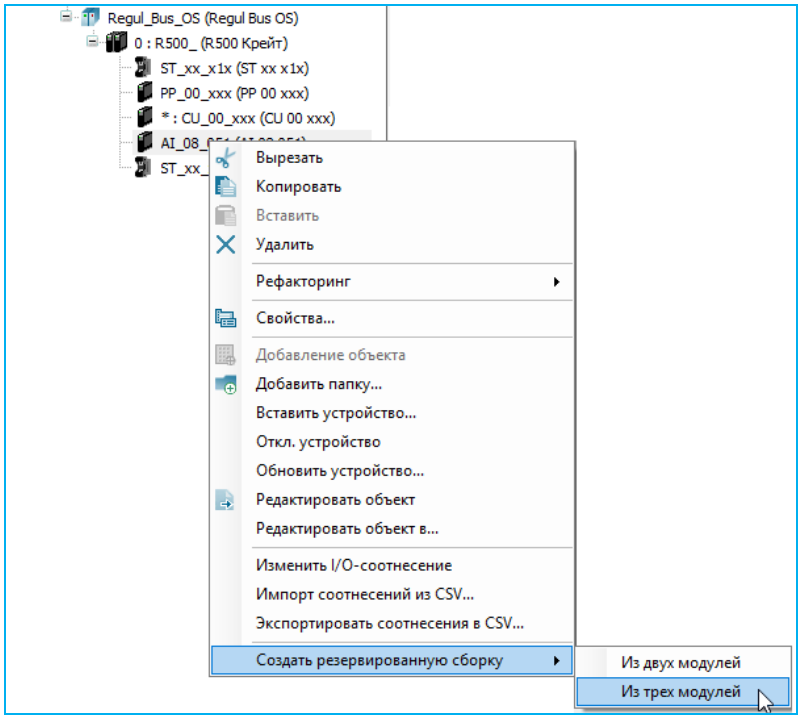


Рисунок 74 – Добавление резервированной сборки

В дереве устройств появится объект **Redundancy_assemblies (Резервированная сборка)** и модули выбранного типа со знаком R, участвующие в резервированной сборке (Рисунок 75).

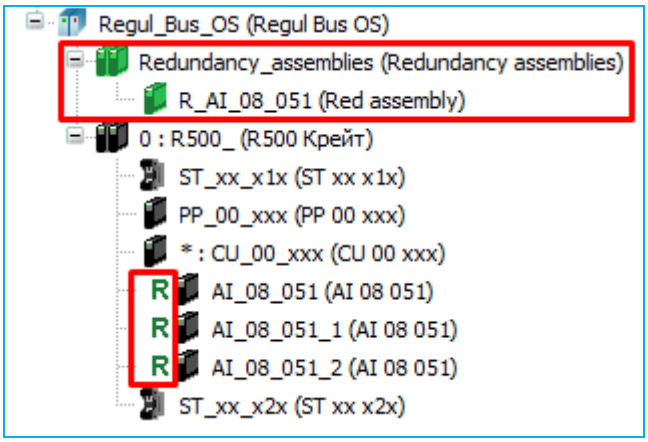


Рисунок 75 - Объекты резервированной сборки

После создания резервированной сборки модули, входящие в ее состав, находятся рядом друг с другом, но в последствии могут быть перераспределены, как внутри одного крейта, так и перемещены в другие. Изменить позицию модуля из сборки можно перетаскиванием с зажатой клавишей мыши (Рисунок 76).

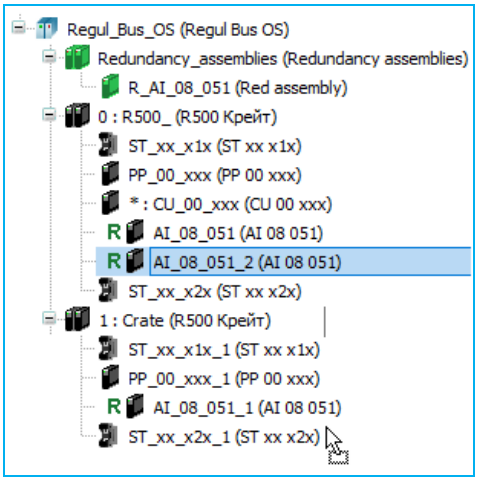


Рисунок 76 – Размещение модулей в составе резервированной сборки в разных крейтах

Для редактирования резервированной сборки выберите в дереве устройств из объекта **Redundancy assemblies** необходимый тип модуля **R_...(Red assembly)**. По двойному щелчку левой кнопки мыши открывается вкладка **Настройка резервированных сборок**.

Наведите курсор мыши на название модуля, подсветится область синим цветом с кнопками: редактировать  или удалить  модуль из сборки.

Для редактирования нажмите на кнопку  и появится окно **Выберите модуль**. Выберите необходимый модуль и нажмите на кнопку ***OK.*** (Рисунок 77).

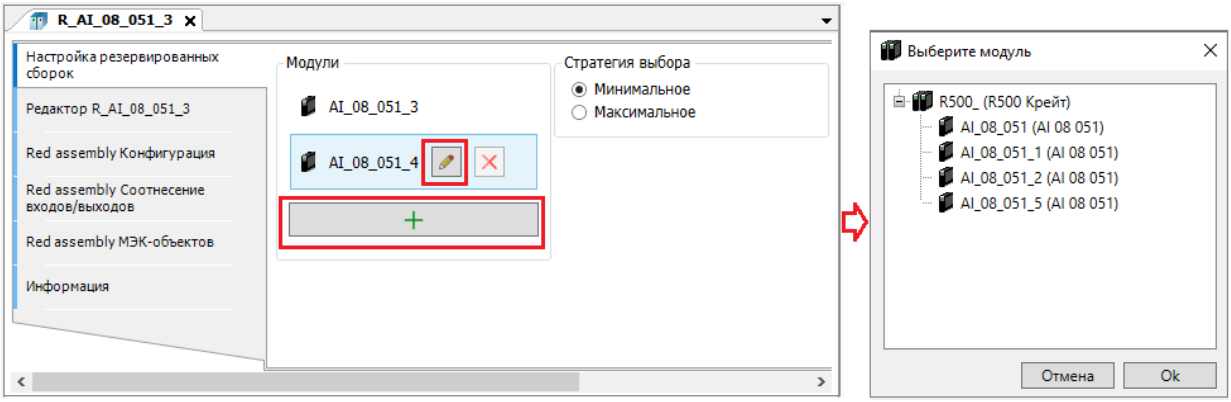


Рисунок 77 – Редактирование резервированной сборки

Для добавления третьего модуля в созданную ранее сборку из двух модулей необходимо предварительно добавить в крейт однотипный модуль, нажать на кнопку  и в открывшемся окне **Выберите модуль** выбрать его из представленного списка. Далее нажмите на кнопку ***OK*** (Рисунок 78).

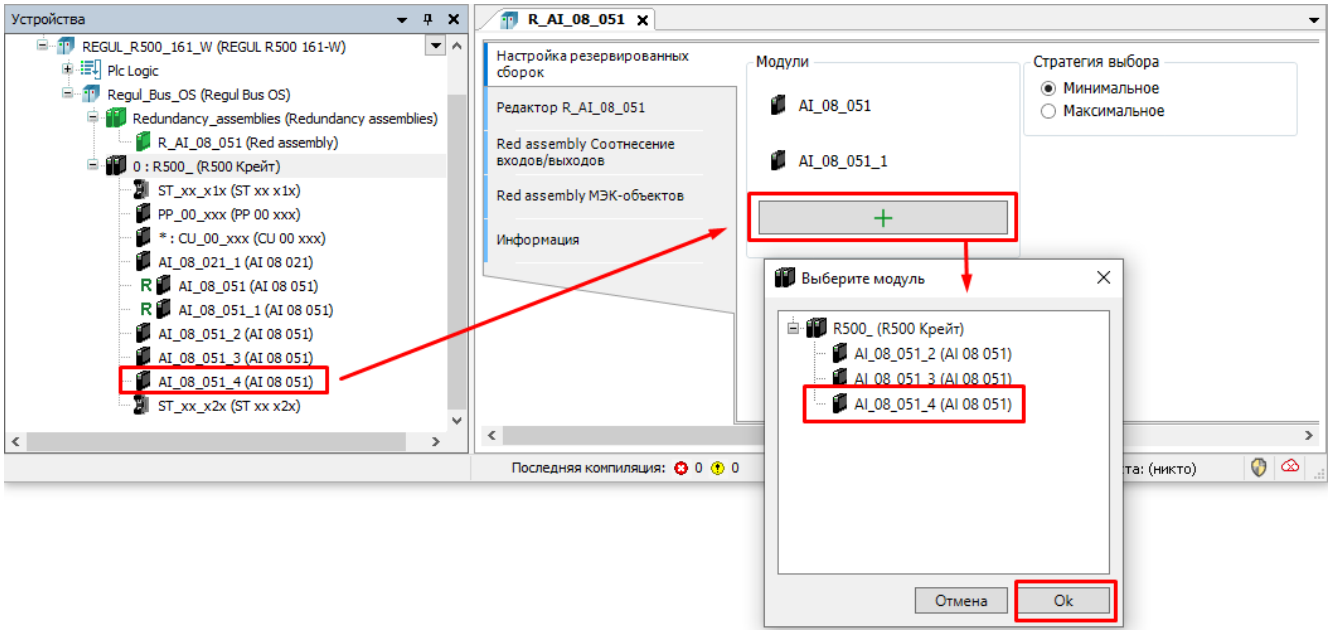


Рисунок 78 - Добавление третьего модуля в сборку

Если в крейте отсутствует однотипный модуль («свободный»), то при нажатии на кнопку + в окне **Выберите модуль** появится предупреждение об отсутствии необходимого типа модуля (Рисунок 79).

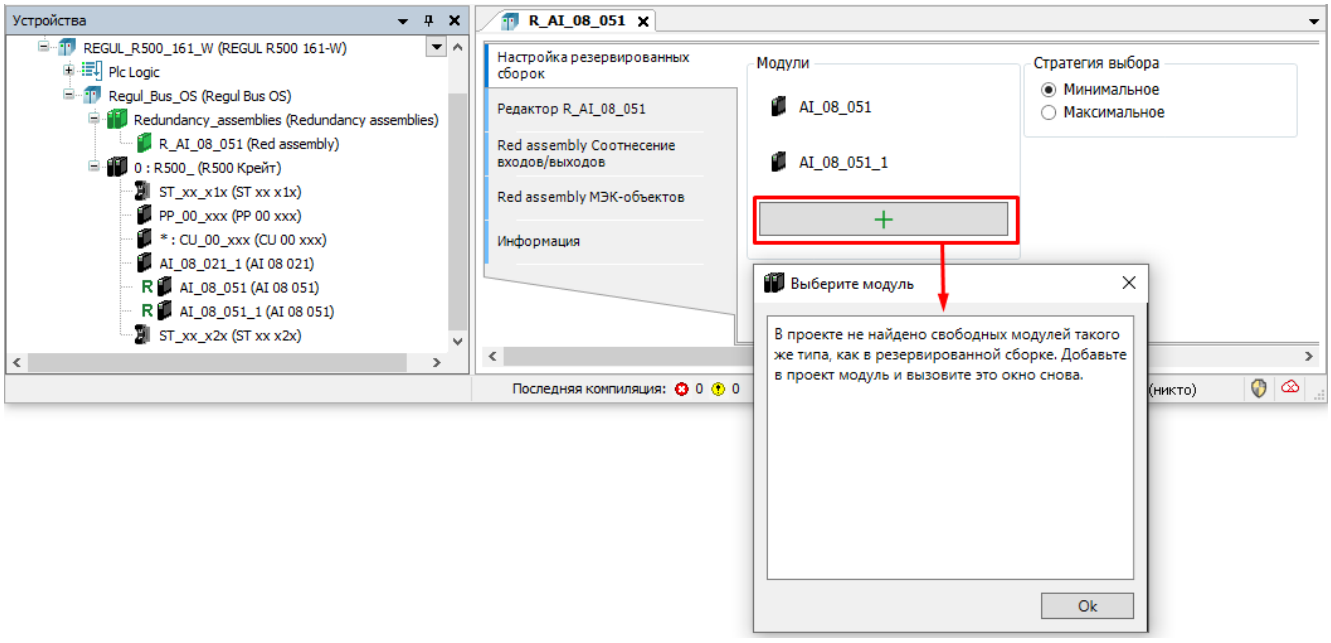


Рисунок 79 – Предупреждение

Конфигурирование стратегий выбора резервированных сборок

Резервированные сборки модулей аналогового ввода

На вкладке **Настройка резервированных сборок** необходимо определить стратегию выбора значений сигналов (Рисунок 80):

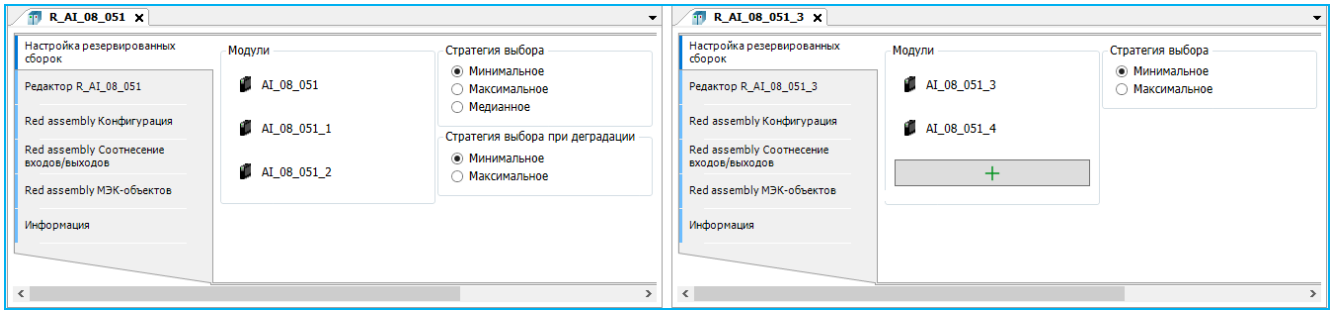


Рисунок 80 - Конфигурирование резервированных сборок модулей аналогового ввода

– при сборке из ***двух модулей***:
  - **стратегия выбора:** *Минимальное* или *Максимальное* значение. Результирующее значение принимается равным минимальному/максимальному значению из исходных. При отказе модуля значение принимается с исправного модуля.
– при сборке из ***трех модулей***:
  - **стратегия выбора:** *Минимальное*, *Максимальное* или *Медианное* значение. Результирующее значение принимается равным минимальному/максимальному значению, либо медианное (максимальное и минимальное значение исключаются) из исходных. Стратегия применяется при исправной работе всех модулей сборки.
  - **стратегия выбора при деградации:** *Минимальное* или *Максимальное* значение. Результирующее значение принимается равным минимальному/максимальному значению из исходных. Стратегия применяется при деградации сборки по причине отказа одного модуля из трех. При отказе двух модулей из трех значение принимается с исправного модуля.

Резервированные сборки модулей дискретного ввода

На вкладке **Настройка резервированных сборок** необходимо определить стратегию выбора объединения значений сигналов по «типу логики» (Рисунок 81):

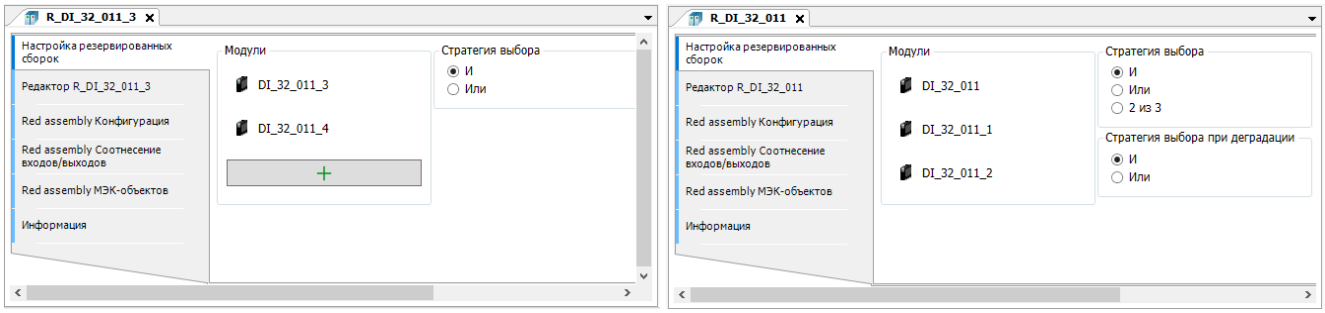


Рисунок 81 - Конфигурирование резервированных сборок модулей дискретного ввода

– при сборке ***из двух модулей:***
  - **стратегия выбора**: *И* или *ИЛИ*. Результирующее значение принимается равным операции логического объединения «И» (AND)/«ИЛИ» (OR) над исходными

значениями (см. таблицу 6). При отказе одного модуля значение принимается с исправного модуля.

Таблица 6 – Результирующее значение стратегии выбора «И»/«ИЛИ»

Дискретное значение с первого модуля	Дискретное значение со второго модуля	Результирующее значение логической операции «И»	Результирующее значение логической операции «ИЛИ»
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

– при сборке из ***трех модулей***:
  - **стратегия выбора:** *И*, *ИЛИ* или *2 из 3*. Результирующее значение принимается равным операции логического объединения «И» (AND)/«ИЛИ» (OR)/«2 из 3» над исходными значениями (см. таблицу 7). Стратегия применяется при исправной работе всех модулей сборки.

Таблица 7 – Результирующее значение стратегии выбора «И»/«ИЛИ»/ «2 из 3»

Дискретное значение с первого модуля	Дискретное значение со второго модуля	Дискретное значение с третьего модуля	Результирующее значение логической операции «И»	Результирующее значение логической операции «ИЛИ»	Результирующее значение логической операции «2 из 3»
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	0	1	1
1	1	1	1	1	1

  - **стратегия выбора при деградации:** *И* или *ИЛИ.* Результирующее значение принимается равным операции логического объединения «И» (AND)/«ИЛИ» (OR) над исходными значениями (см. таблицу 6). Стратегия применяется при деградации сборки по причине отказа одного модуля из трех. При отказе двух модулей из трех результирующее значение принимается с исправного модуля.

Резервированные сборки модулей дискретного/аналогового вывода

В резервированных сборках модулей дискретного/аналогового вывода команда поступает одновременно на все модули, входящие в сборку (Рисунок 82).

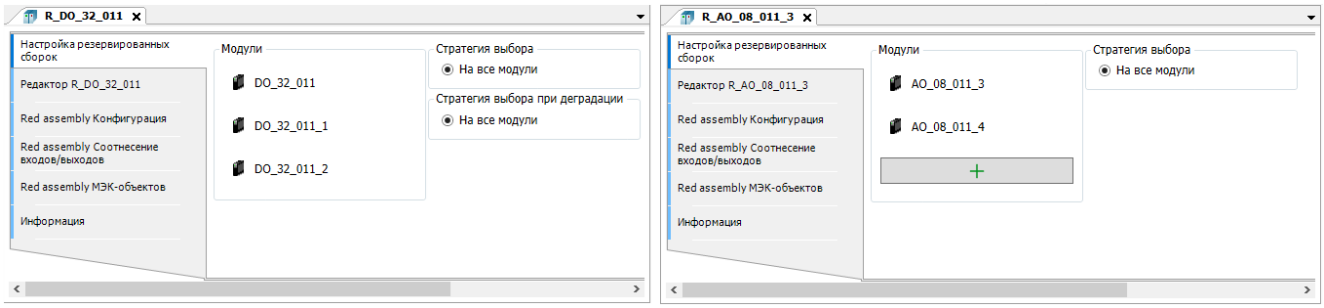


Рисунок 82 - Вкладка **Настройка резервированных сборок** модулей аналогового/дискретного вывода

Задание параметров модуля резервированной сборки

Перейдите на вкладку **Редактор R_...** объекта **Redundancy_assemblies** (Рисунок 83). На вкладке все параметры идентичны применяемому типу модуля в сборке (см. раздел «Настройка параметров модулей. Задание параметров модулей…»). В резервированной сборке осуществляется единая настройка параметров, которая наследуется для всех модулей в сборке, поэтому помодульная настройка заблокирована.

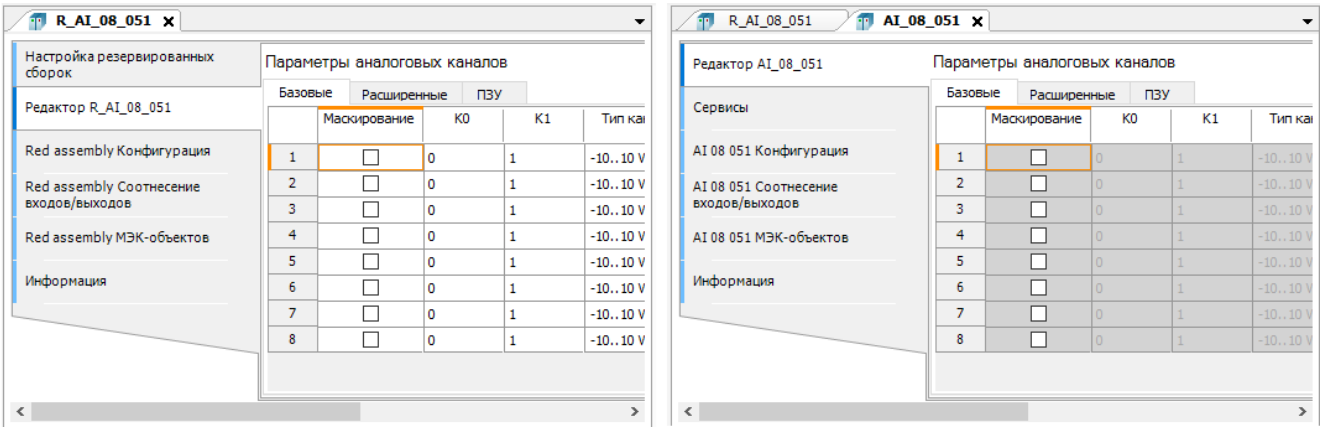


Рисунок 83 – Редактор параметров аналоговых каналов модуля для резервированной сборки на примере модуля аналогового ввода AI 08 051

Резервированные сборки комбинированных модулей сочетают в себе параметры каналов дискретных/аналоговых модулей ввода и вывода.

Привязка переменных резервированной сборки

Логика обработки сигналов в резервированных сборках учитывает результаты диагностики отдельных модулей. Соответственно, при аппаратных проблемах с модулем или трактом сигнала он бракуется, и сборка исключает его из логики обработки данных. Эта ситуация приводит к деградации сборки, однако, сигнал на ее выходе остается достоверным. Учитывая

возможности модулей по самодиагностике, резервированные сборки обеспечивают достоверный выходной сигнал и оповещение пользователя.

ИНФОРМАЦИЯ

Программное обеспечение резервированной сборки формирует единую переменную, по результатам обработки данных нескольких каналов. Вместе с тем, каждый канал сборки остается доступным индивидуально. К нему и его статусу можно привязать отдельную переменную (вкладка **<Наименование модуля> Соотнесение входов/выходов,** см. рисунок 84)

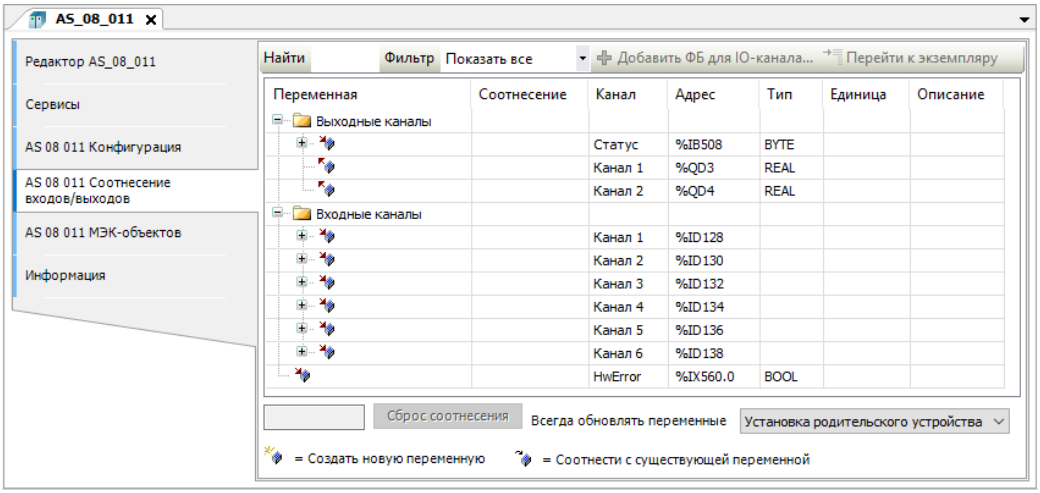


Рисунок 84 – Вкладка соотнесения входов/выходов на примере модуля R500 AS 08 011

На вкладке **Red assembly Соотнесение входов/выходов** (Рисунок 85) осуществляется привязка переменных резервированной сборки аналогично описанию, приведенному в разделе «Привязка каналов к переменным программы».

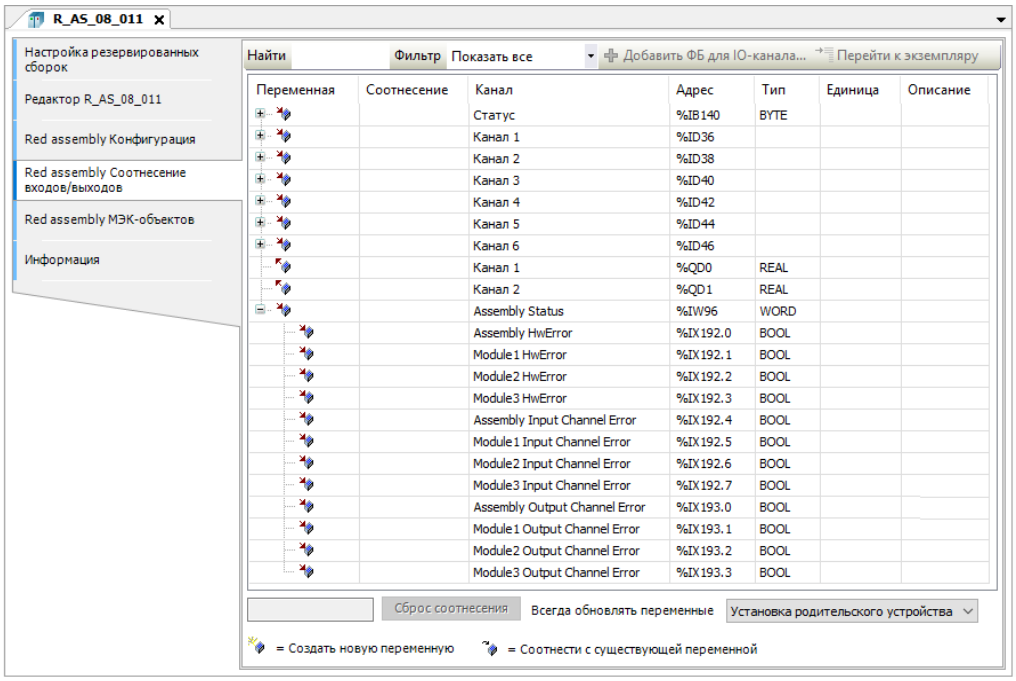


Рисунок 85 – Вкладка соотнесения входов/выходов резервированной сборки на примере модуля R500 AS 08 011

Дополнительно, к основным переменным (в зависимости от типа модуля, входящего в резервированную сборку), на вкладке будут присутствовать:

- *Assembly HwError* - ошибка сборки (параметр принимает значение «0», когда все модули в резервированной сборке имеют ошибку HwError);
- *ModuleN_HwError* - ошибка модуля **N** (где **N** – номер модуля, от 1 до 3);
- *InputChannelError* - ошибка входного канала резервированной сборки (параметр принимает значение «0», когда для всех модулей в сборке существует входной канал с ошибкой);
- *ModuleN_InputChannelError* - ошибка какого-либо из входных каналов модуля **N** (где **N** – номер модуля, от 1 до 3). Для определения, на каком из каналов появилась ошибка, и детальной проработки причин ее возникновения, следует обратиться к статусным данным модуля;
- *OutputChannelError* - ошибка выходного канала резервированной сборки (параметр принимает значение «0», когда для всех модулей в сборке существует выходной канал с ошибкой);
- *ModuleN_OutputChannelError* - ошибка какого-либо из выходных каналов модуля **N** (где **N** – номер модуля, от 1 до 3). Для определения, на каком из каналов появилась ошибка, и детальной проработки причин ее возникновения, следует обратиться к статусным данным модуля.

НАСТРОЙКА ПАРАМЕТРОВ МОДУЛЕЙ

Модули контроллеров серии Regul RX00 имеют ряд параметров, которые используются для того, чтобы настроить работу модуля под конкретную прикладную задачу. К таким параметрам, относятся, например, маскирование каналов ввода/вывода, коэффициенты калибровки аналогового канала ввода/вывода, скорость обмена по каналам коммуникационных модулей. Конкретный список доступных для задания параметров модуля зависит от его типа.

Настройку параметров можно производить как при подключении к контроллеру (онлайн режим), так и в офлайн режиме. В первом случае новые параметры будут применены сразу, но в проекте сохранены не будут. Соответственно, при следующей перезагрузке контроллера или прикладной программы все новые настройки будут потеряны, так как будут заменены значениями из текущего проекта.

Во втором случае, то есть в офлайн режиме, параметры контроллера сохраняются в проекте и будут применены при следующей загрузке программы в контроллер.

Редактор модуля

Выберите нужный модуль в дереве устройств. По двойному щелчку левой кнопки мыши открывается вкладка редактора модуля (Рисунок 86). Для модулей Regul R600 можно перейти в этот редактор из редактора крейта (двойной щелчок мыши по изображению модуля).

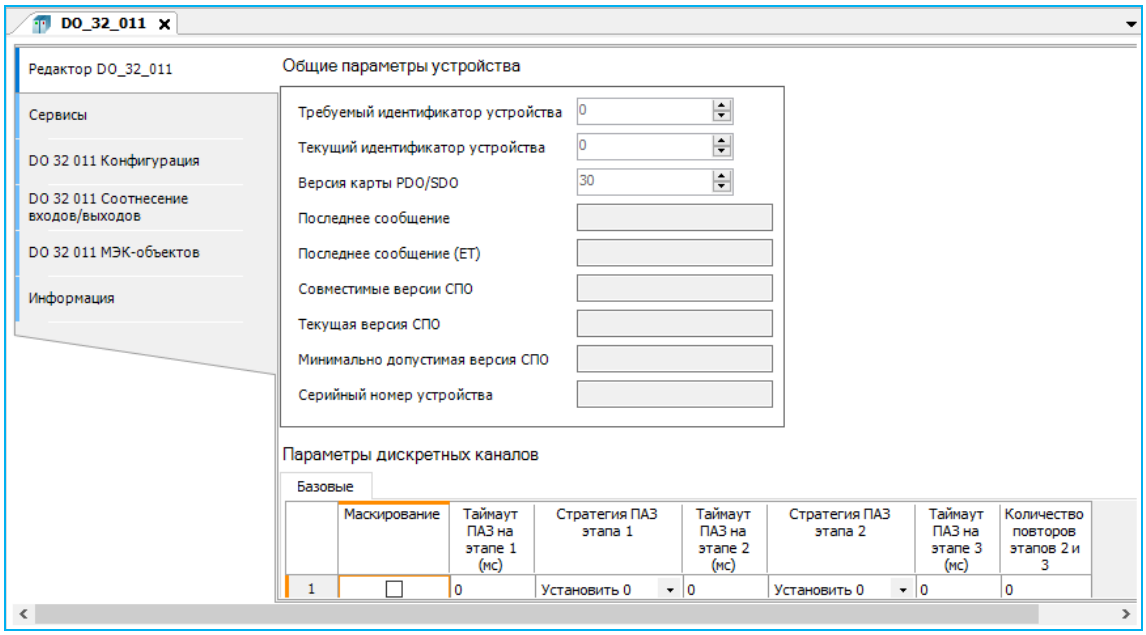


Рисунок 86 – Пример редактора модуля в офлайн режиме

Все редакторы модулей имеют типовую структуру и состоят из следующих групп:

– базовые параметры, общие для всех модулей;

– параметры, специфичные для данного модуля;

– таблица параметров каналов с возможностью чтения/записи значений параметров как в офлайн режиме, так и при подключении к контроллеру;

– отдельная вкладка редактора, где осуществляется привязка переменных к входам/выходам модуля.

Параметры, общие для всех модулей:

– Требуемый идентификатор устройства (Device ID required);

– Текущий идентификатор устройства (Device ID current);

– Версия карты PDO/SDO (PDO/SDO map version);

– Последнее сообщение (Last message);

– Последнее сообщение (ET) (Last message (ET));

– Совместимые версии СПО (Compatible FW versions);

– Текущая версия СПО (FW version current);

– Минимально допустимая версия СПО (Minimal FW version);

– Серийный номер устройства (в шестнадцатеричном представлении)

Например, значение серийного номера устройства на вкладке: 0x00AB20B1, что в переводе на десятичную систему счисления равно **11215025**, которое соответствует фактическому серийному номеру, указанному на корпусе модуля.

Это системные параметры имеют атрибут *Только для чтения* и используются средой исполнения контроллера для идентификации самого модуля и его типа.

Редактор модуля ввода/вывода в режиме онлайн будет выглядеть следующим образом (Рисунок 87).

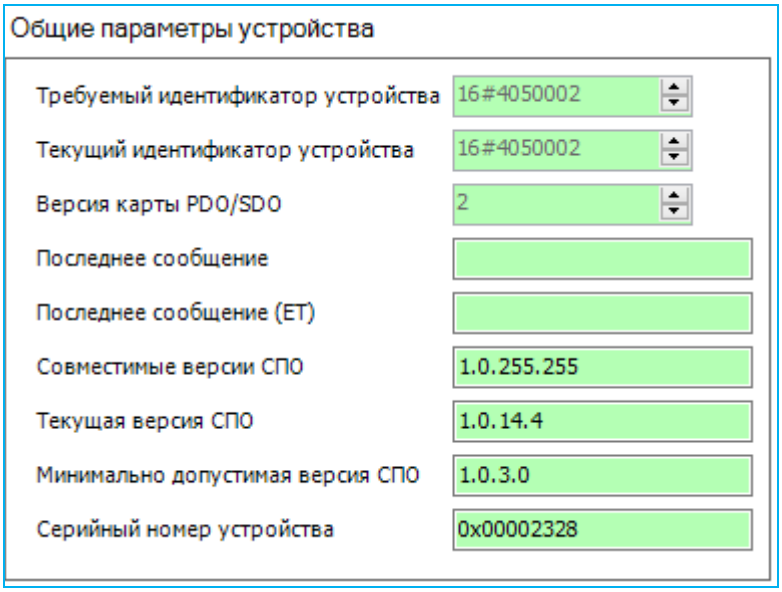


Рисунок 87 - Пример отображения редактора модуля в режиме онлайн

ИНФОРМАЦИЯ

Вывод информации в полях с «версиями» в режиме онлайн, производится в десятеричном представлении

Если в онлайн-режиме изменить доступные для редактирования параметры модуля ввода/вывода (либо шины), то фон измененного параметра сменится с зеленого на желтый цвет (Рисунок 88)

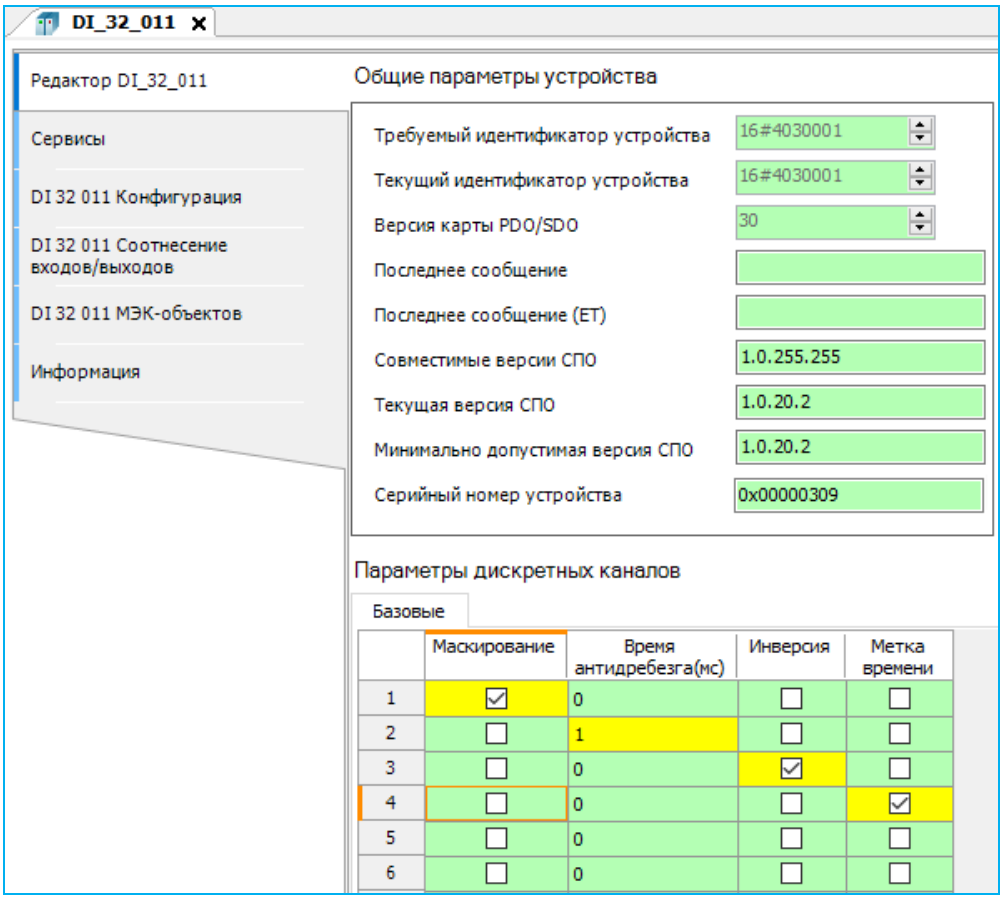


Рисунок 88 - Пример отображения измененного параметра модуля в онлайн-режиме

Ниже описана настройка параметров конкретных модулей в среде разработки Astra.IDE. Подробное описание алгоритма работы каждого модуля, его технические характеристики, другие параметры приведены в документах «Regul R200. Системное руководство», «Regul R500. Системное руководство», «Regul R600. Системное руководство».

Сохранение и восстановление настроек, хранящихся в ПЗУ

Для модулей ввода/вывода предусмотрена возможность сохранения настроек, хранящихся в ПЗУ, в отдельный файл.

Установите нужные параметры аналоговых/дискретных каналов (см. подразделы «Задание параметров модулей»). Правой клавишей мыши вызовите контекстное меню (Рисунок 89).

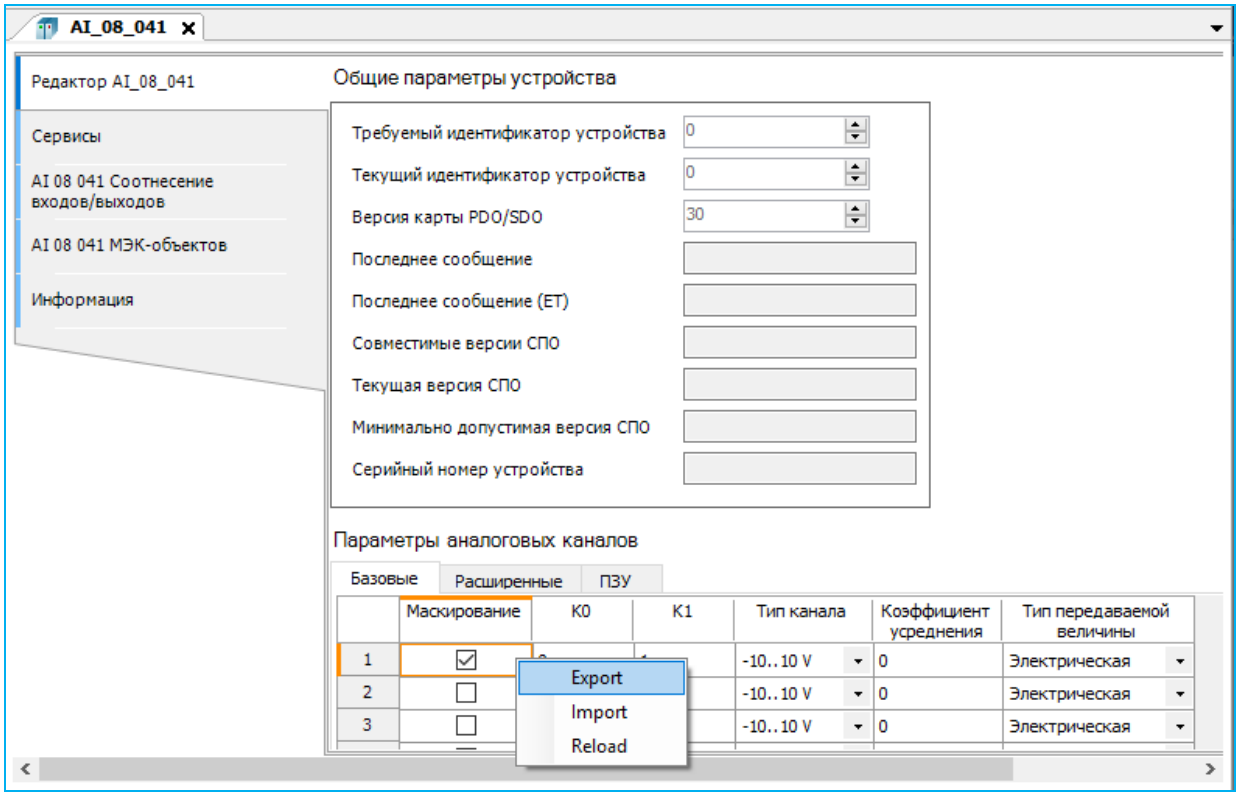


Рисунок 89 – Контекстное меню для сохранения/восстановления настроек

Для сохранения параметров выберите пункт контекстного меню **Export**. Откроется окно **Export params data**, где укажите имя файла, который будет содержать настройки, и местоположение этого файла на ПК пользователя. Нажмите кнопку ***Сохранить***. Параметры каналов будут сохранены на ПК (то есть вне модуля) в файле с расширением csv.

В процессе работы с контроллером при изменении значений параметров каналов в таблице может возникнуть необходимость восстановить параметры, указанные ранее. Это возможно, если прежние настройки были экспортированы в файл. Выберите пункт контекстного меню **Import**. Откроется окно **Import params data**. Выберите файл с расширением csv, в котором хранятся значения параметров модуля, нажмите кнопку ***Открыть***. Параметры модуля будут восстановлены из файла.

Задание параметров модулей источника питания

Интеллектуальные модули источника питания (модуль источника питания R500 PP 00 051 и модуль источника внешнего питания R500 PO 08 041) осуществляют обмен информацией с модулем ЦП по шине RegulBus. На вкладке редактора модулей присутствуют системные параметры (Рисунок 90).

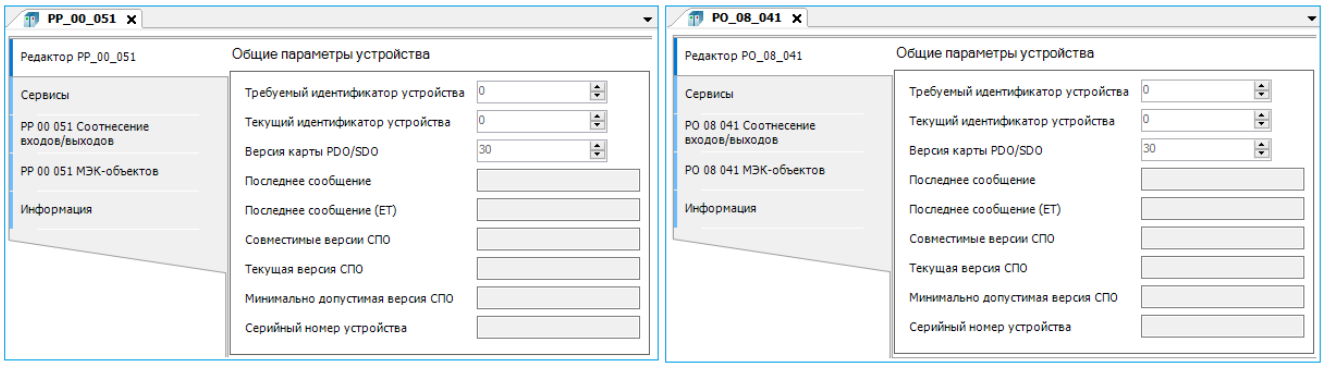


Рисунок 90 - Пример отображения редактора модуля R500 PP 00 051 и R500 PO 08 041

Базовые параметры

Для настройки каждого канала модуля источника внешнего питания R500 PO 08 041 доступен следующий параметр (Рисунок 91):

– **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается (не выдается выходное значение). По замаскированным каналам нет индикации статуса ошибки. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован.

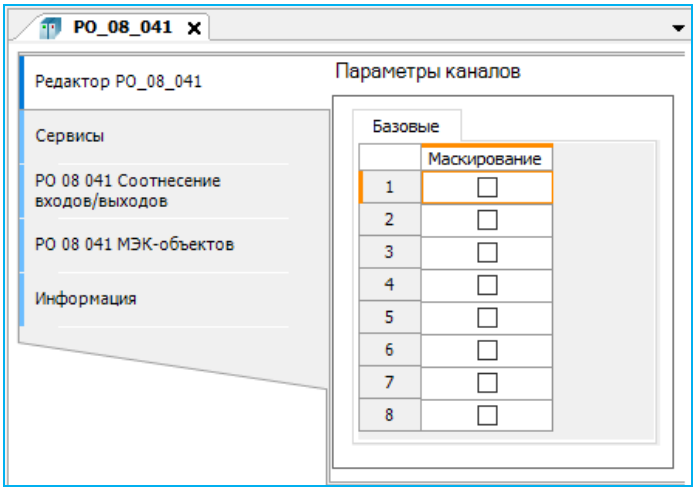


Рисунок 91 – Базовые параметры модуля R500 PO 08 041

Задание параметров модулей аналогового ввода

Базовые параметры

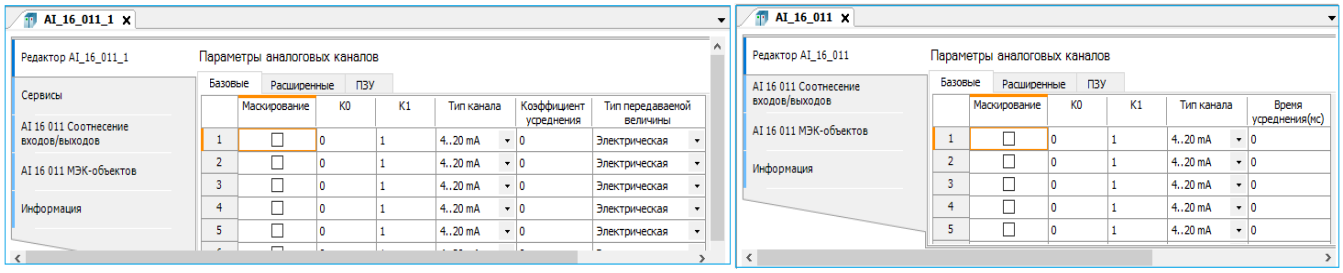


Рисунок 92 – Базовые параметры аналоговых каналов на примере модуля R500 AI 16 011 и модуля R600 AI 16 011

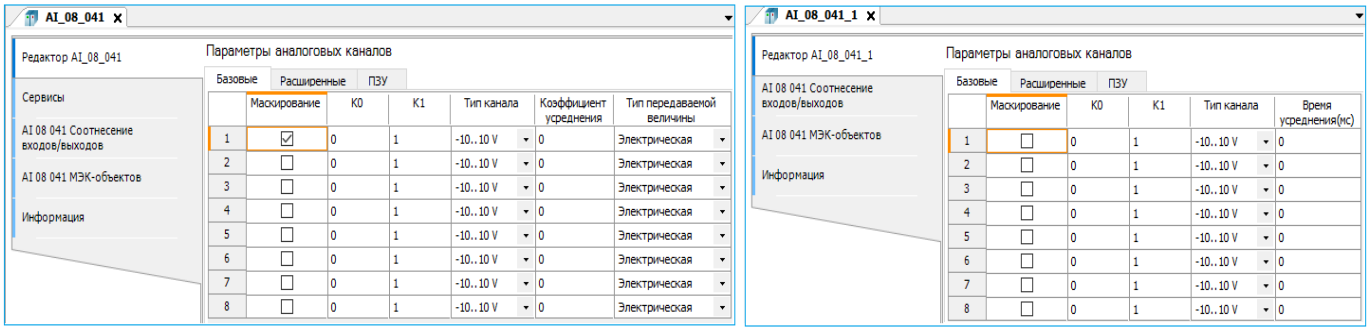


Рисунок 93 – Базовые параметры аналоговых каналов на примере модуля R500 AI 08 041 и модуля R600 AI 08 041

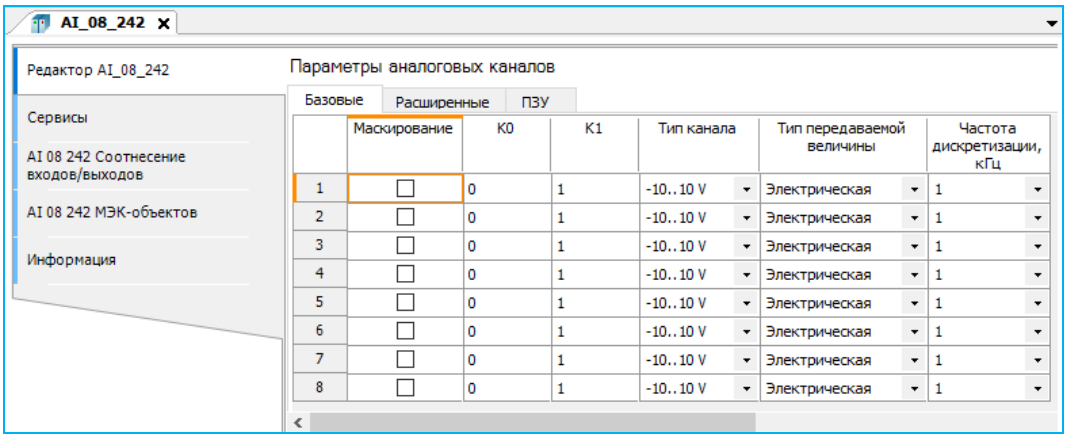


Рисунок 94 – Базовые параметры аналоговых каналов на примере модуля R500 AI 08 242/342

ВНИМАНИЕ!

Модули аналогового ввода R500 AI 08 242/342 не поддерживают работу с модулями ЦП II-го типа. Присутствуют ограничения при настройке параметров шины RegulBus (см. описание в разделе «Редактор шины»)

Для настройки модулей аналогового ввода доступны следующие основные параметры каждого канала:

– **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По замаскированным каналам нет индикации обрыва. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован;

ИНФОРМАЦИЯ

Для модуля AI 16 011, имеющего один АЦП, замаскированный канал исключается из цикла опроса. Таким образом полный опрос всех входов происходит быстрее

– **K_0** и **K_1** – коэффициенты преобразования электрической величины в инженерную. Они индивидуальны для каждого диапазона измерений каждого аналогового канала. Зависят от датчика, используемого для измерений, и задаются пользователем при конфигурировании модуля. Эти коэффициенты сохраняются в проекте;

i	**ИНФОРМАЦИЯ** Если использование встроенного функционала модуля по бракованию и линеаризации сигнала не требуется, или если датчик обладает нелинейной характеристикой, то значения этих коэффициентов необходимо оставить неизменными. В этом случае в качестве измеренных значений модуль будет передавать в прикладную программу значение электрической величины

- **Тип канала** – выбор диапазона измерения. Возможные значения (в зависимости от модуля): ***-10...10 V; 0…10 V; 4...20 mA; 0...20 mA***;
- **Коэффициент усреднения λ** - коэффициент усреднения α в диапазоне значений [0…1]. По умолчанию установлено значение *0* - усреднение выключено;
- **Время усреднения (мс)** - задает время в миллисекундах, за которое усредненное значение достигнет 95% от новой величины сигнала, при его одномоментном изменении, при условии, что значение сигнала и усредненной величины до этого были равны. Усреднение измеряемого значения производится с использованием функции экспоненциально взвешенного скользящего среднего;
- **Тип передаваемой величины** – формат, в котором предоставляется измеряемая величина. Три варианта значений:
  - *Коды АЦП* – непосредственно код аналого-цифрового преобразователя;
  - *Электрическая* – значение электрической величины входного сигнала (мА, В);
  - *Инженерная* – значение инженерной величины, измеренной первичным преобразователем (давление, температура, масса, уровень и так далее).

Дополнительные базовые настройки для модуля R500 AI 08 242/342 (СПО 1.0.30.0 и выше):

- **Частота дискретизации, кГц** – частота преобразования непрерывного во времени аналогового сигнала в последовательность отсчетов с определенным временным шагом.
- За цикл преобразования формирует от 1 до 8 массивов данных (замаскированные каналы не участвуют в опросе). Все сформированные за очередной цикл преобразования массивы помечаются в MCU одной меткой времени.

Пример кода для получения данных с модуля R500 AI 08 242/342 смотри в приложении И.

Дополнительные настройки для модулей аналогового ввода RX00 AI 0X X31

Компенсация температуры холодного спая задается следующим образом (Рисунок 95):

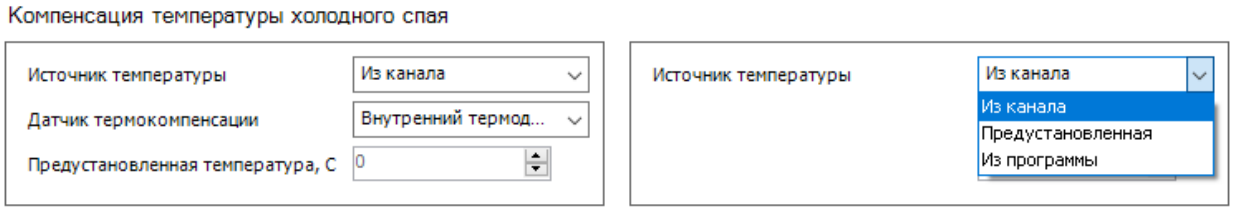


Рисунок 95 – Задание параметров источника компенсации температуры холодного спая

– **Источник температуры** – выбор способа получения данных о температуре:
  - *Из канала* – выделение отдельного канала, к которому подключается термосопротивление, измеряющее температуру в точке холодного спая (при этом отключается настройка параметра «Предустановленная температура»);
  - *Предустановленная* – значение температуры задается пользователем (при этом отключается настройка параметра «Канал термодатчика»);
  - *Из программы* – использование значения температуры через прикладную программу (при этом отключается настройка параметров «Канал термодатчика» и «Предустановленная температура»);
– **Датчик термокомпенсации** (при выборе способа получения данных о температуре холодного спая «*Из канала*»):
  - *Внутренний термодатчик* – использование встроенного в модуль датчика температуры;
  - *Канал 1…8* – выделение отдельного канала, к которому подключается термосопротивление, измеряющее температуру в точке холодного спая;
– **Предустановленная температура** – задается пользователем температура холодного спая при выборе способа получения данных о температуре холодного спая «*Предустановленная*».

Параметры аналоговых каналов задаются следующим образом (Рисунок 96):

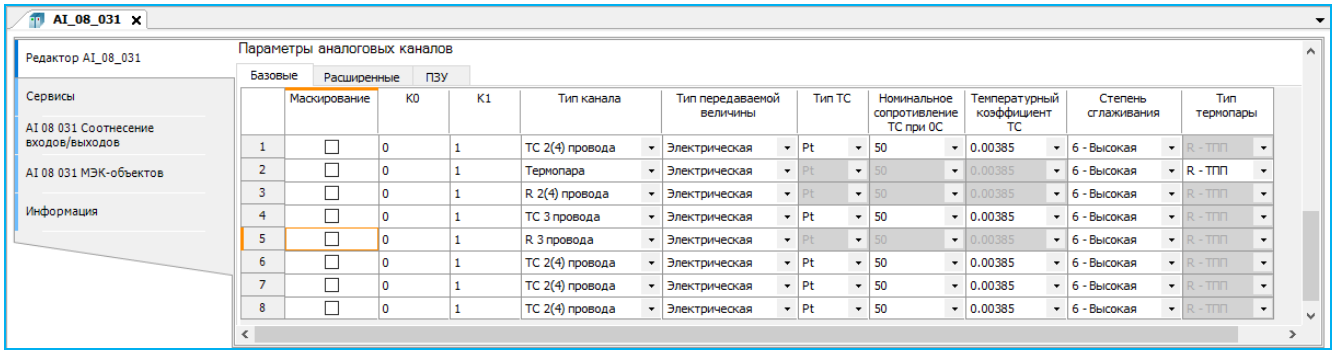


Рисунок 96 Параметры аналоговых каналов на примере модуля R500 AI 08 031

– **Тип канала** – выбор преобразователя и типа его подключения, со следующими возможными вариантами:

- *TC 2(4), TC 3* - подключен термопреобразователь сопротивления по двух-/трех-/четырехпроводной схеме;
- *Термопара* - подключена термопара;
- *R 2(4), R 3* - подключен датчик с выходом в виде сопротивления по двух-/трех-/четырехпроводной схеме;
- *U, mV* - измерение напряжения, в диапазоне: - *400 mV…400 mV*;

– **Тип ТС** – тип материала термопреобразователя сопротивления. Возможные варианты: *Pt* – платина, *CU* – медь, *Ni* – никель;

– **Номинальное сопротивление ТС** – нормированное изготовителем сопротивление при 0 °С, со следующими возможными вариантами:

- *Pt* – 46, 50, 100*;*
- *CU* – 50, 53, 100;
- *Ni* – 50, 100;

– **Температурный коэффициент ТС** – изменение величины сопротивления от температуры. Возможные варианты:

- *Pt* – при **Номинальное сопротивление ТС=46** устанавливается только 0.00385, при значениях 50 или 100 – 0.00385 либо 0.00391;
- *CU* – при **Номинальное сопротивление ТС=53** устанавливается только 0.00426, при значениях 50 или 100 – 0.00426 либо 0.00428;
- *Ni* – значение всегда 0.00617;

– **Степень сглаживания** – величина, определяющая усреднение измеренных значений, для ослабления высокочастотных составляющих. Возможные варианты: *1 – Низкая* (16,7 Гц)*…6 – Высокая* (4,17 Гц);

– **Тип термопары** – тип термопары. Возможные варианты: *R-ТПП, S-ТПП, B-ТПР, J-ТЖК, T-ТМК, E-ТХКн, K-ТХА, N-ТНН, A(A-1, A-2, A-3)-ТВР, L-ТХК.*

Расширенные параметры

Для модулей аналогового ввода предусмотрены дополнительные параметры, позволяющие применять автоматическое бракование сигналов, когда необходимо автоматически обнаруживать аномальные значения или скорости изменения измеряемых сигналов. При браковании сигнала, модуль автоматически устанавливает общий флаг бракования канала и флаг, соответствующий критерию, по которому сигнал был отбракован. Поддерживаются два типа бракования: по выбросу сигнала и по выходу за границу.

Окно дополнительных параметров модулей аналогового ввода открывается при выборе вкладки **Расширенные** (Рисунки 97, 98).

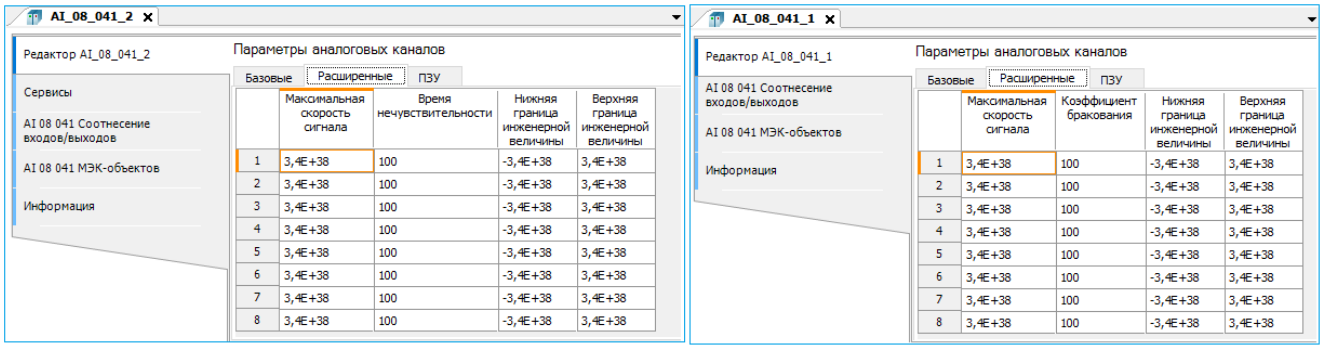


Рисунок 97 - Расширенные параметры модулей аналогового ввода на примере R500 AI 08 041 и R600 AI 08 041

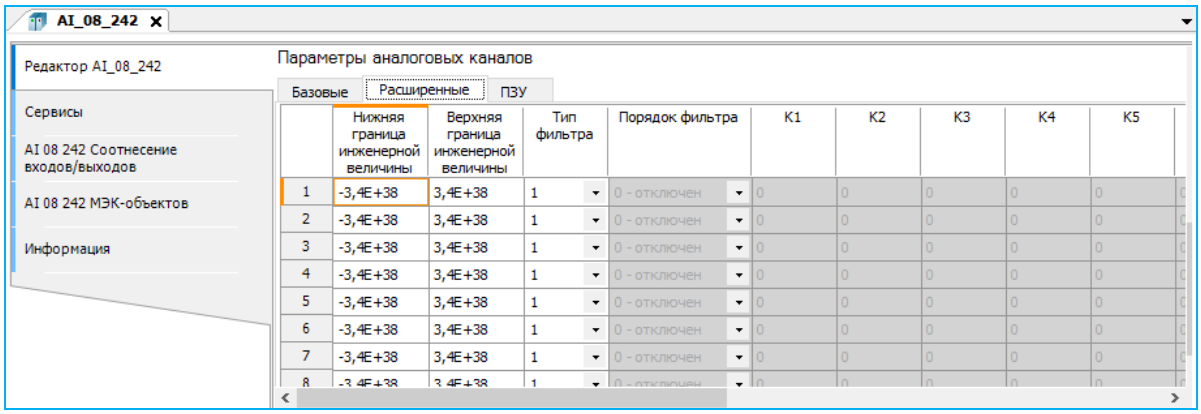


Рисунок 98 - Расширенные параметры модуля аналогового ввода на примере R500 AI 08 242

Сигнал бракуется при слишком резком изменении инженерной величины (выброс сигнала). Для конфигурирования алгоритма бракования сигнала используются следующие параметры:

– **Максимальная скорость сигнала**. Максимальная скорость изменения инженерной величины за цикл опроса незамаскированных каналов модуля;

– **Время нечувствительности**. Интервал времени в мс, по истечении которого происходит установка статуса бракования канала. По умолчанию равен 100 – статус бракования канала установится по истечении 100 мс (для модулей серии R200/R500);

– **Коэффициент бракования**. Интервал времени в мс (задается как процент от величины усреднения), по истечении которого происходит установка статуса бракования канала. По умолчанию равен 100 – статус бракования канала установится по истечении времени, заданного параметром «Время усреднения, мс» (для модулей серии R600).

Бракование сигнала по выходу за границы производится при выходе за следующие пределы:

– пределы измерений АЦП модуля - зависит от установленного АЦП и не может быть сконфигурирован пользователем;

– пределы возможного изменения «электрического» сигнала - зависит от типа электрического сигнала, измеряемого данным каналом. Например, если канал работает в режиме измерения тока 4 – 20 мА, границы этого интервала будут равны, соответственно 4 и 20 мА. При значении тока ниже 4 мА или выше 20 мА канал будет забракован. Значения границ этого интервала также не могут быть сконфигурированы пользователем;

– пределы возможного изменения инженерной величины. Задаются пользователем и определяются спецификой технологического процесса: нижняя граница инженерной величины, верхняя граница инженерной величины.

Дополнительные расширенные настройки для модуля R500 AI 08 242/342

Пользователь может задать порядок цифрового фильтра и определить его коэффициенты, либо выбрать из трех вариантов с предустановленными значениями фильтра (КИХ, 15-го порядка). Параметры задаются следующим образом:

– **Тип фильтра**. Доступны фильтры со следующей степенью сглаживания:
  - **1** – низкая степень (Fd = 100 кГц; Fc = 100 Гц);
  - **2** – средняя степень (Fd = 100 кГц; Fc = 10 Гц);
  - **3** – высокая степень (Fd = 100 кГц; Fc = 1 Гц);
  - **Пользовательский**. Пользователь задает сам: *Порядок фильтра* – от 0 до 15 (0-отключен) и *K1…K15* – коэффициенты фильтра.

Степень сглаживания определяется отношением частоты среза (Fc) к частоте дискретизации (Fd). Частотные составляющие, превышающие частоту среза, подавляются.

Калибровочные коэффициенты

Окно калибровки каналов модулей аналогового ввода открывается при выборе вкладки **ПЗУ** (Рисунок 99).

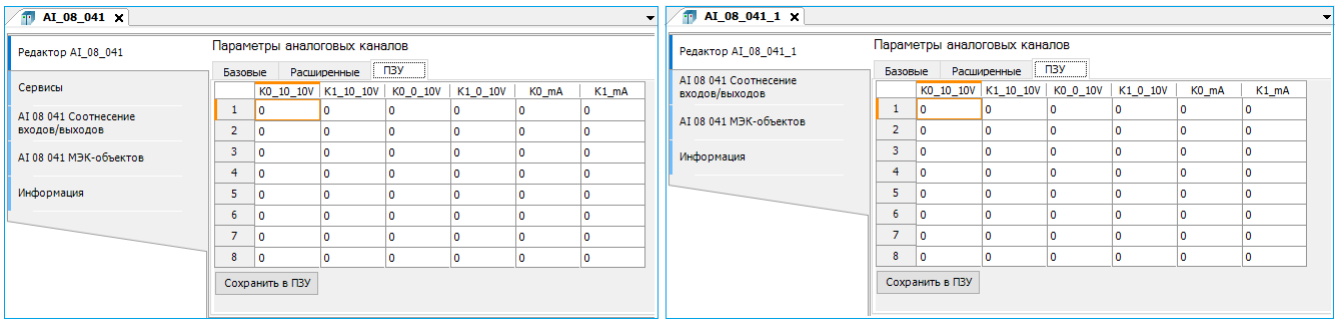


Рисунок 99 – Установка калибровочных коэффициентов каналов на примере модулей R600 AI 08 041 и R600 AI 08 031

Коэффициенты **K0** и **K1** отличаются от описанных выше коэффициентов $\mathbf{K_0}$ и $\mathbf{K_1}$. По сути, **K0** и **K1** являются калибровочными коэффициентами канала, они принадлежат конкретному физическому каналу конкретного модуля и хранятся только в нем (а не в проекте). Их значение вычисляется и записывается в ПЗУ модуля на стадии его производства. В последующем, в процессе эксплуатации модуля, при необходимости перекалибровать канал, значения этих коэффициентов могут быть заново вычислены и перезаписаны.

Изменение значений таких параметров может осуществляться только в онлайн-режиме (при подключении к контроллеру) и для сохранения новых значений нужно нажать кнопку ***Сохранить в ПЗУ***.

i	**ИНФОРМАЦИЯ** Если в системе с частичным резервированием (с ведущего модуля ЦП) произвести запись калибровочных коэффициентов в ПЗУ, то, возможна кратковременная, на период записи, потеря связи ведомого модуля ЦП с модулем ввода/вывода, при условии, что таймаут модуля был меньше времени записи коэффициентов

Задание параметров модулей аналогового вывода

Базовые параметры

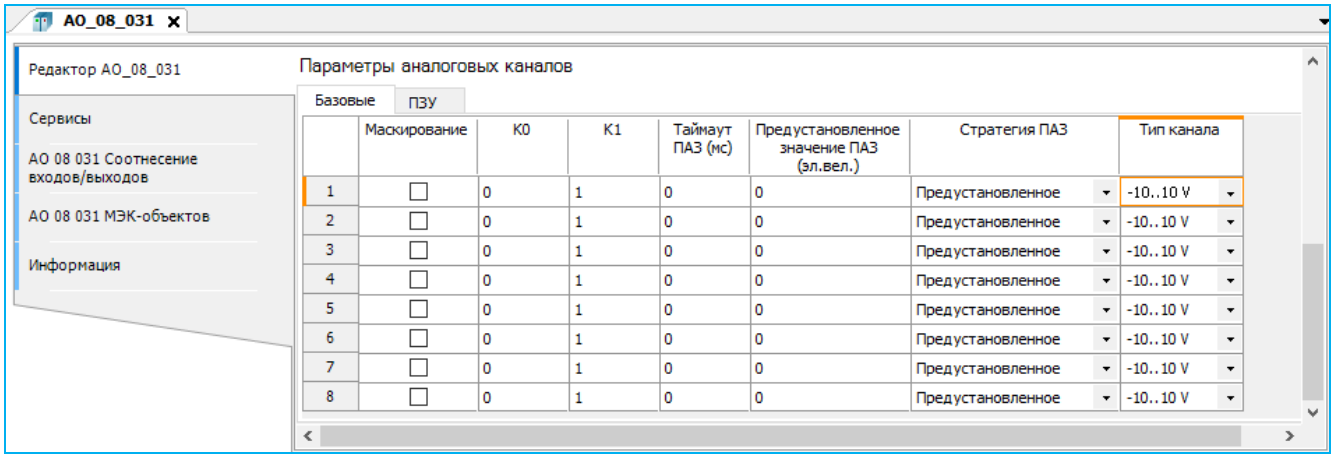


Рисунок 100 – Базовые параметры аналоговых каналов на примере модуля R500 AO 08 031

Для настройки доступны следующие основные параметры каждого канала:

- **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По замаскированным каналам нет индикации обрыва. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован;
- **K_0** и **K_1** - коэффициенты преобразования сигнала из инженерной величины в электрический сигнал. По умолчанию эти коэффициенты равны «0» и «1» соответственно. То есть без настройки каналов из прикладной программы в модуль передается управляющий сигнал в виде значения силы тока на выходе. При желании пользователя коэффициенты K_0, K_1 могут быть изменены индивидуально для каждого канала как при конфигурации контроллера, так и в процессе его работы. Эти коэффициенты сохраняются в проекте;
- **Тип канала** – выбор диапазона воспроизведения. Возможные значения (в зависимости от модуля): *-10..10 V*; *0..10 V*; *4..20 mA*; *0..20 mA*.

	**ИНФОРМАЦИЯ** Для работы функции HART в модулях аналогового ввода/вывода требуется устанавливать диапазон от 4 до 20 мА. Если выбранный диапазон не будет соответствовать режиму HART, то при компиляции, в окне сообщений, отобразится сообщение об ошибке: «Для HART требуется режим 4-20 mA, канала…»

Задание параметров алгоритма противоаварийной защиты

Модули аналогового вывода имеют возможность реализовать противоаварийную защиту при наступлении одного из следующих событий:

- потеря связи с модулем центрального процессора;
- аппаратная неисправность центрального процессора;
- ошибка в системном и/или прикладном программном обеспечении модуля центрального процессора.

В рамках выполнения противоаварийной защиты модуль осуществляет управление аналоговыми каналами в зависимости от выбранной конфигурации (стратегии управления и времени на ее исполнение).

- **Таймаут ПАЗ (мс)** – отрезок времени, мс, в течение которого держится стратегия ПАЗ. Диапазон [0 – 65535] (0 – бесконечность). По истечении таймаута, на всех каналах вывода будет установлено значение «0»;
- **Предустановленное значение ПАЗ (эл. вел.)** – значение, которое будет выставлено на канале, в случае потери связи (вне зависимости от исходных значений, установленных на модуле);
- **Стратегия ПАЗ** - определение сценария изменения состояния канала при потере связи. Возможные значения: *предустановленное* – установить предустановленное значение, *не изменять* – не изменять состояние канала.

Если в момент выполнения алгоритма ПАЗ (после потери связи с ЦП) связь с модулем ЦП будет восстановлена, то модуль безударно вернется в состояние выполнения заложенного программой функционала.

	**ИНФОРМАЦИЯ** На шине **RegulBus OS** доступна возможность активации ПАЗ в случае ИСКЛЮЧЕНИЯ и/или в режиме СТОП, но только для модулей с соответствующей версией СПО (см. раздел «Конфигурирование крейтов. Редактор шины»)

Калибровочные коэффициенты

Перейдите на вкладку ПЗУ (Рисунок 101).

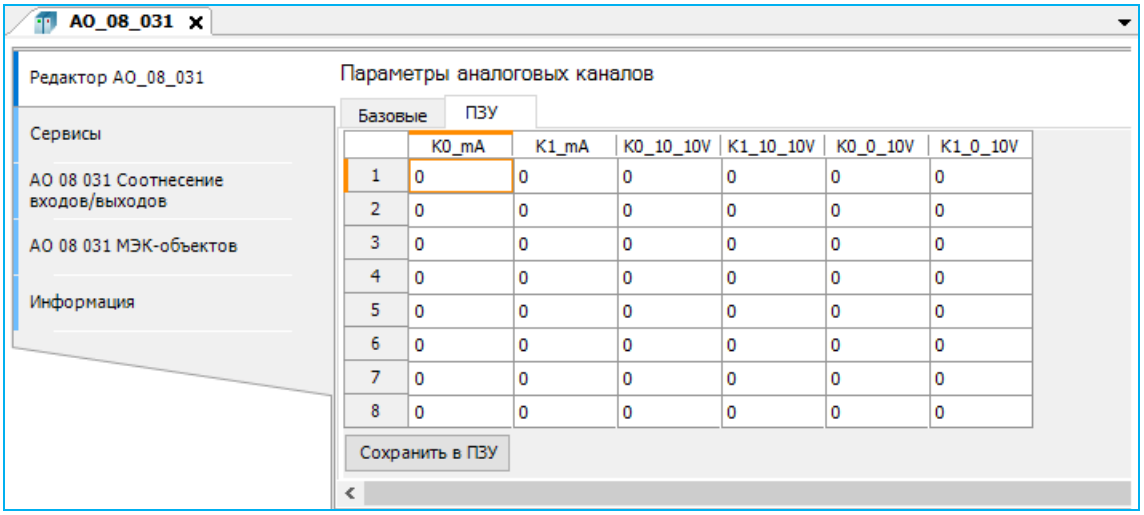


Рисунок 101 – Калибровочные коэффициенты аналоговых каналов на примере модуля R500 AO 08 031

Коэффициенты **K0** и **K1** отличаются от коэффициентов $\mathbf{K_0}$ и $\mathbf{K_1}$. По сути, **K0** и **K1** являются калибровочными коэффициентами канала, они принадлежат конкретному физическому каналу конкретного модуля и хранятся только в нем (а не в проекте). Первично они прописываются при заводской калибровке модуля, хранятся в ПЗУ модуля.

Изменения значений таких параметров может осуществляться только в онлайн-режиме (при подключении к контроллеру) и для сохранения новых значений нужно нажать кнопку ***Сохранить в ПЗУ***.

i	**ИНФОРМАЦИЯ** Если в системе с частичным резервированием (с ведущего модуля ЦП) произвести запись калибровочных коэффициентов в ПЗУ, то, возможна кратковременная, на период записи, потеря связи ведомого модуля ЦП с модулем ввода/вывода, при условии, что таймаут модуля был меньше времени записи коэффициентов

Задание параметров модулей аналоговых комбинированных

Комбинированные аналоговые модули сочетают в себе параметры каналов аналоговых модулей ввода и вывода, которые описаны выше (Рисунок 102).

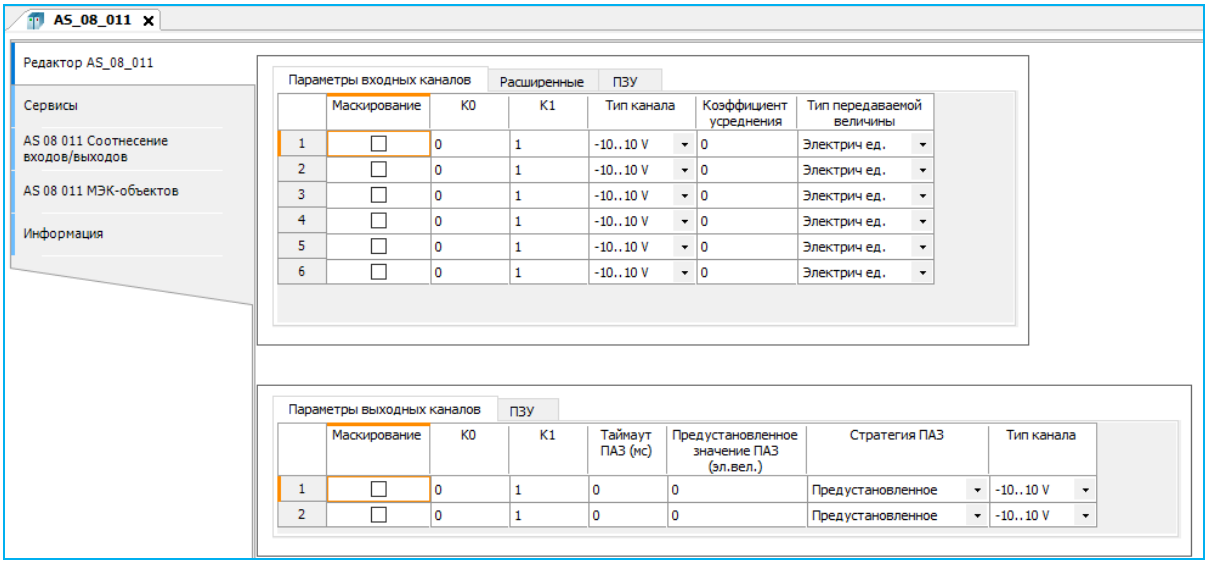


Рисунок 102 – Параметры аналоговых каналов ввода и вывода на примере модуля R500 AS 08 011

Задание параметров модулей дискретного ввода

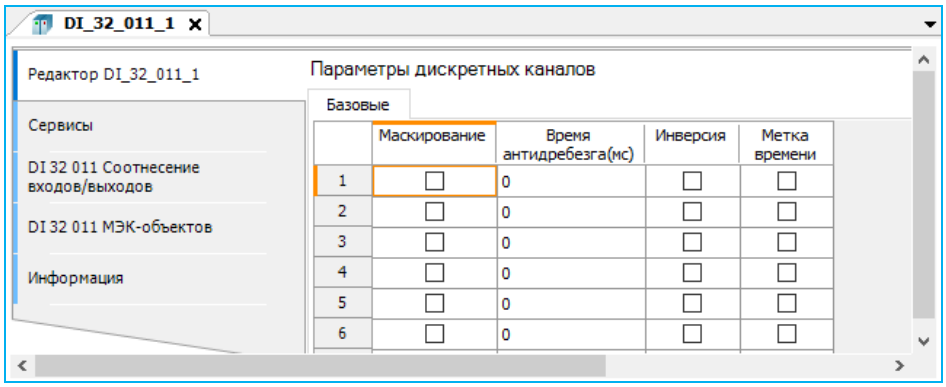


Рисунок 103 – Параметры дискретных каналов на примере модуля R500 DI 32 011

Для настройки доступны следующие основные параметры каждого канала:

– **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован;

– **Время антидребезга (мс)** – минимальное время между сменами состояния $0 \leftrightarrow 1$, допустимое для регистрации смены состояния. Задается в мс;

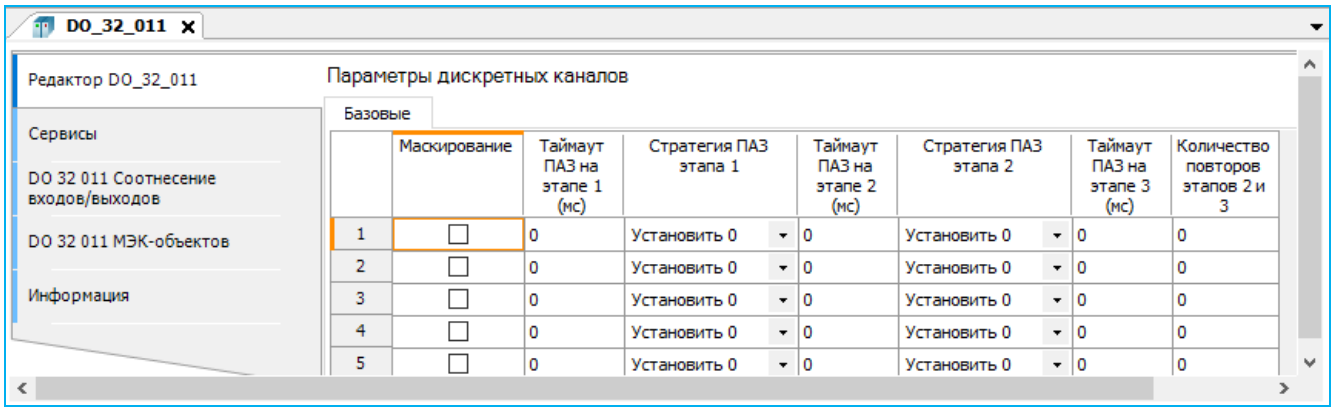
– **Инверсия** - установка флажка в этом поле включает инверсию канала. По умолчанию установлено значение 0 (пустое поле) – инверсии нет;

– **Метка времени** – фиксация краткосрочных дискретных событий. Установка флажка в этом поле означает, что будет вестись архив последних 10 событий. По умолчанию установлено значение 0 (пустое поле) – архив не ведется.

Порядок работы с метками времени на канале приведен в приложении И.

Задание параметров модулей дискретного вывода

Для каждого канала модуля есть возможность активировать параметр **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается, не выдаются выходные значения. По умолчанию установлено значение *0* (пустое поле) – канал не замаскирован.



	Маскирование	Таймаут ПАЗ на этапе 1 (мс)	Стратегия ПАЗ этапа 1	Таймаут ПАЗ на этапе 2 (мс)	Стратегия ПАЗ этапа 2	Таймаут ПАЗ на этапе 3 (мс)	Количество повторов этапов 2 и 3
1	☐	0	Установить 0	0	Установить 0	0	0
2	☐	0	Установить 0	0	Установить 0	0	0
3	☐	0	Установить 0	0	Установить 0	0	0
4	☐	0	Установить 0	0	Установить 0	0	0
5	☐	0	Установить 0	0	Установить 0	0	0

Рисунок 104 – Параметры дискретных каналов на примере модуля R500 DO 32 011

Задание параметров алгоритма противоаварийной защиты

Модули дискретного вывода имеют возможность реализовать алгоритм противоаварийной защиты при наступлении одного из следующих событий:

– потеря связи с модулем центрального процессора;

– аппаратная неисправность центрального процессора;

– ошибка в системном и/или прикладном программном обеспечении модуля центрального процессора.

В рамках выполнения алгоритма противоаварийной защиты модуль осуществляет управление выходными дискретными каналами в несколько конфигурируемых этапов (максимально – 3, с возможностью цикличного повторения этапов) с разными временными отрезками (максимально 65 535 мс на отрезок) и разными стратегиями управления на каждом этапе. Алгоритм управления каналом представлен на рисунке (Рисунок 105), где T1, T2, T3 – таймаут ПАЗ соответствующего этапа, S1, S2, S3 – стратегия ПАЗ этапа, Cnt – количество повторов.

⚠	**ВНИМАНИЕ!** Стратегия этапа S3 - выполняет автоматическую инверсию текущего состояния канала (то есть 0→1, 1→0), без возможности установки значения состояния пользователем

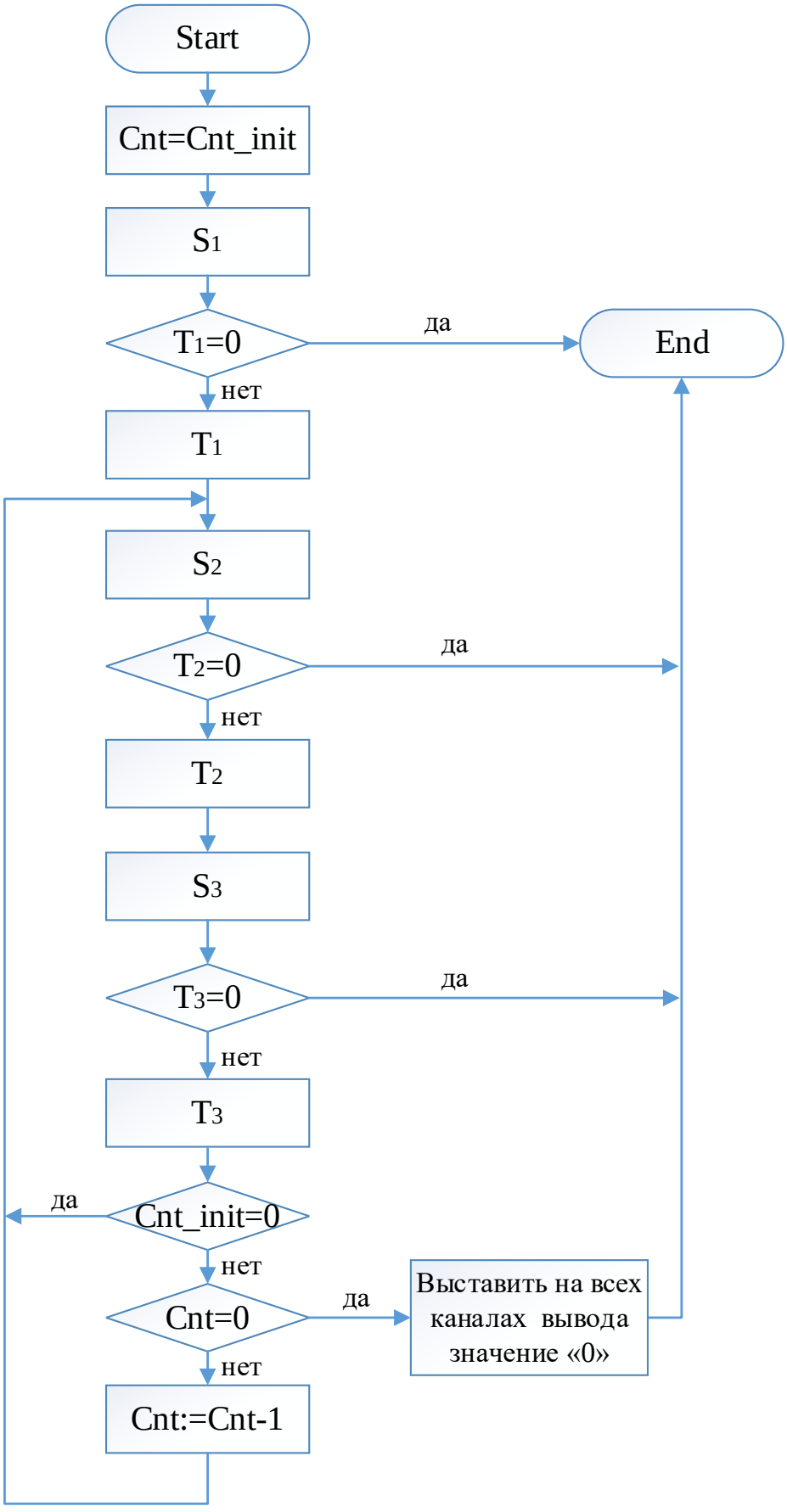


Рисунок 105 – Алгоритм управления каналом

Для реализации вышеописанного алгоритма каждому каналу нужно установить значения следующих параметров:

- **Таймаут ПАЗ на этапе 1 (мс) (на этапе 2, на этапе 3)** – отрезок времени этапа 1 (этапа 2, этапа 3), мс, в течение которого держится стратегия соответствующего этапа. Диапазон [0 – 65535], (0 – это бесконечность);
- **Стратегия ПАЗ этапа 1 (Стратегия ПАЗ этапа 2)**. Возможные варианты:
  - *установить 0;*
  - *не изменять*;
  - *установить 1*;
- **Стратегия ПАЗ этапа 3.** Автоматическая инверсия текущего состояния канала (то есть 0→1, 1→0);
- **Количество повторов этапов 2 и 3** – значение количества повторений этапа 2 и 3. Диапазон [0 – 65535] (0 – это бесконечность).

Пример 1. Вне зависимости от значений, установленных на модуле, в случае потери связи установить значение 1. Значения параметров: **Таймаут ПАЗ на этапе 1 (мс)** - *0*, **Стратегия ПАЗ этапа 1** – *установить 1*, остальные параметры могут иметь любые значения.

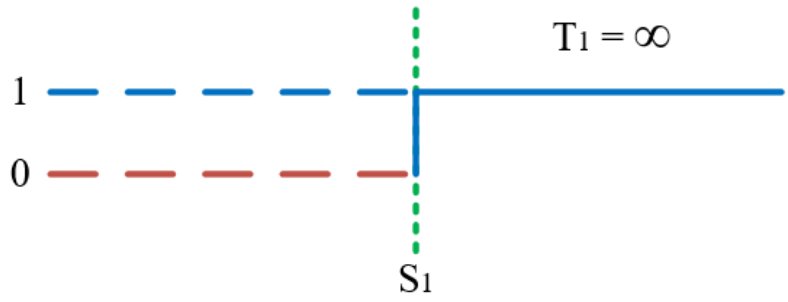


Рисунок 106 – Пример 1

Пример 2. Вне зависимости от значений, установленных на модуле, в случае потери связи оставить их неизменными. Параметры: **Таймаут ПАЗ на этапе 1 (мс)** - *0*, **Стратегия ПАЗ этапа 1** – *не изменять*, остальные параметры могут иметь любые значения.

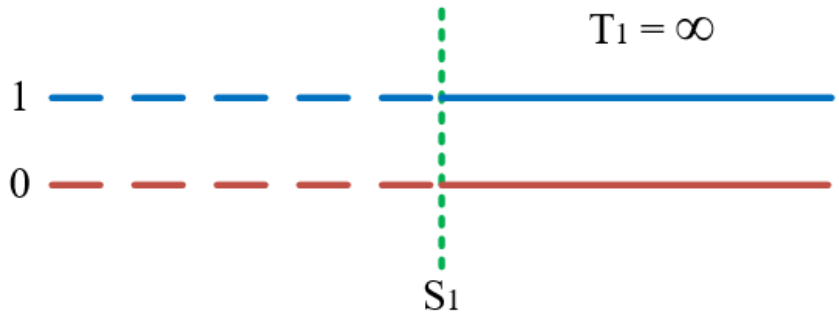


Рисунок 107 – Пример 2

Пример 3. Вне зависимости от значений, установленных на модуле, в случае потери связи оставить их неизменными на протяжении этапа 1, потом установить значение 1. Параметры: **Таймаут ПАЗ на этапе 1 (мс)** – *10000 мс*, **Стратегия ПАЗ этапа 1** – *не изменять*, **Таймаут ПАЗ на этапе 2 (мс)** – *0*, **Стратегия ПАЗ этапа 2** – *установить 1*, остальные параметры могут иметь любые значения.

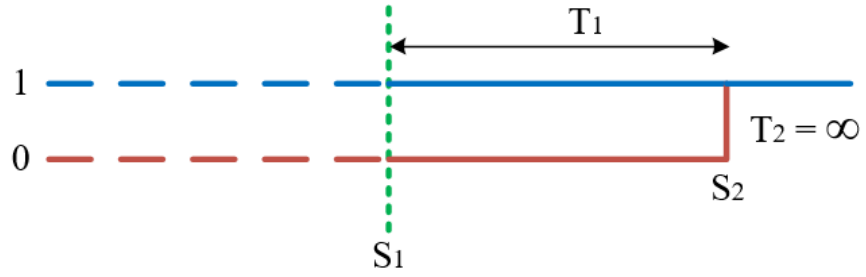


Рисунок 108 – Пример 3

Пример 4. Параметры: **Таймаут ПАЗ на этапе 1 (мс)** – *10000 мс*, **Стратегия ПАЗ этапа 1** – *установить 0*, **Таймаут ПАЗ на этапе 2 (мс)** – *2000 мс*, **Стратегия ПАЗ этапа 2** – *установить 1*, **Таймаут ПАЗ на этапе 3 (мс)** – *0*, остальные параметры могут иметь любые значения.

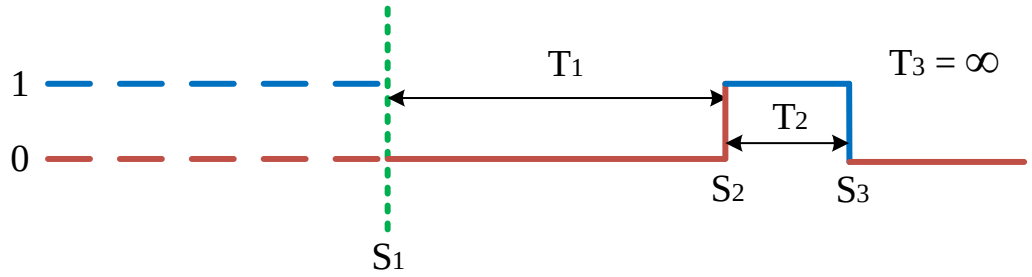


Рисунок 109 – Пример 4

Пример 5. Параметры: **Таймаут ПАЗ на этапе 1 (мс)** – *10000 мс*, **Стратегия ПАЗ этапа 1 (мс)** – *установить 0*, **Таймаут ПАЗ на этапе 2 (мс)** – *2000 мс*, **Стратегия ПАЗ этапа 2** – *установить 1*, **Таймаут ПАЗ на этапе 3 (мс)** – *5000 мс*, **Количество повторов** – *2*.

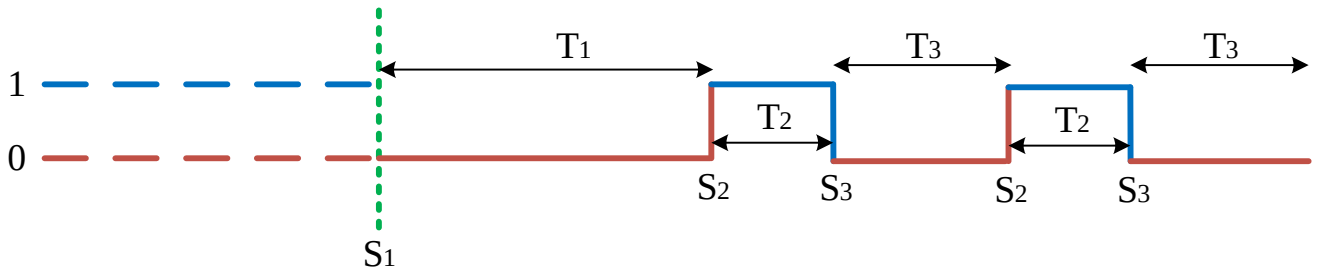


Рисунок 110 – Пример 5

Если в момент выполнения алгоритма ПАЗ (после потери связи с ЦП) связь с модулем ЦП будет восстановлена, то модуль дискретного вывода безударно вернется в состояние выполнения заложенного программой функционала.

i	**ИНФОРМАЦИЯ** На шине **RegulBus OS** доступна возможность активации ПАЗ в случае ИСКЛЮЧЕНИЯ и/или в режиме СТОП, но только для модулей с соответствующей версией СПО (см. раздел «Конфигурирование крейтов. Редактор шины»)

Задание параметров модулей дискретных комбинированных

Комбинированные дискретные модули сочетают в себе параметры каналов дискретных модулей ввода и вывода, которые описаны выше (Рисунок 111).

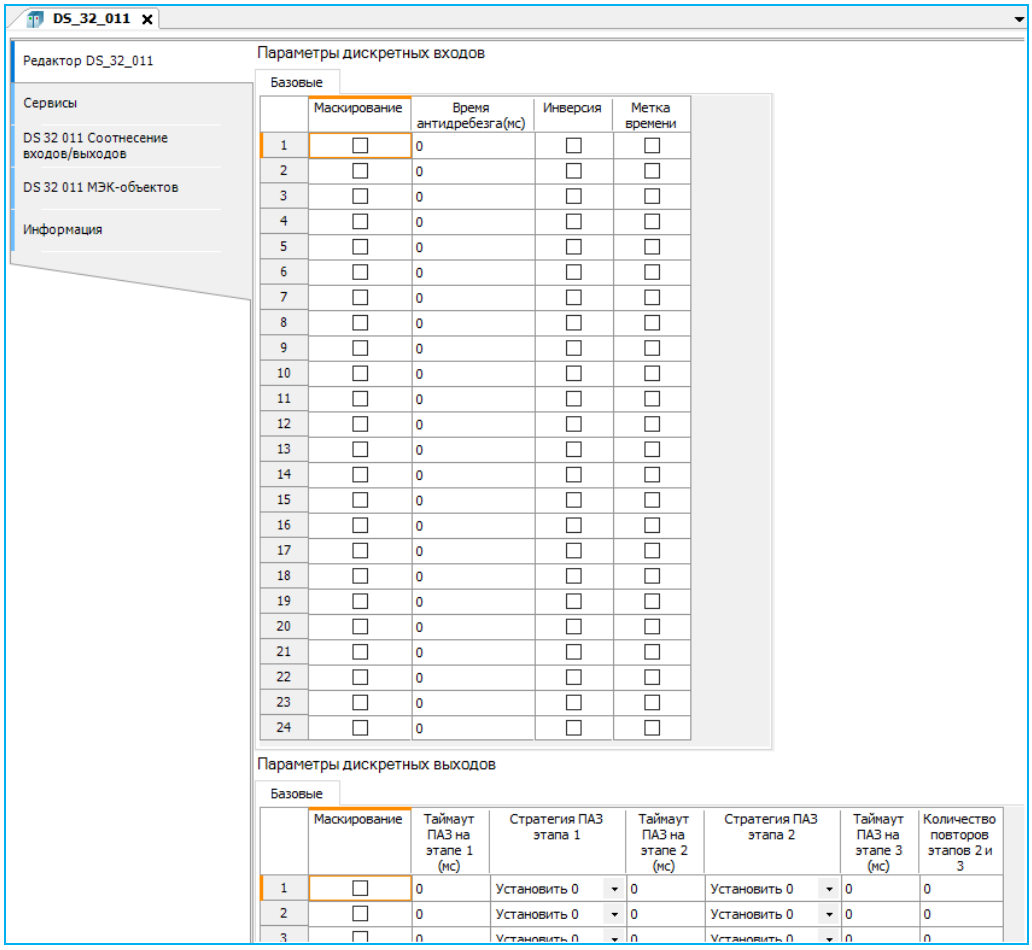


Рисунок 111 – Параметры дискретных каналов ввода и вывода на примере модуля R500 DS 32 011

Задание параметров модулей счета импульсов

Модули счета импульсов могут работать в одном из следующих режимов:

– частотомер до 10 кГц с подсчетом количества импульсов;

– частотомер до 500 кГц;

- обработка данных с энкодера;
- модуль системы измерения качества и количества нефти (LACT);
- автомат безопасности (R500 DA 03 021, R600 DA 03 021, R200 DA 01 011).

Настроечные параметры модуля (частотомер)

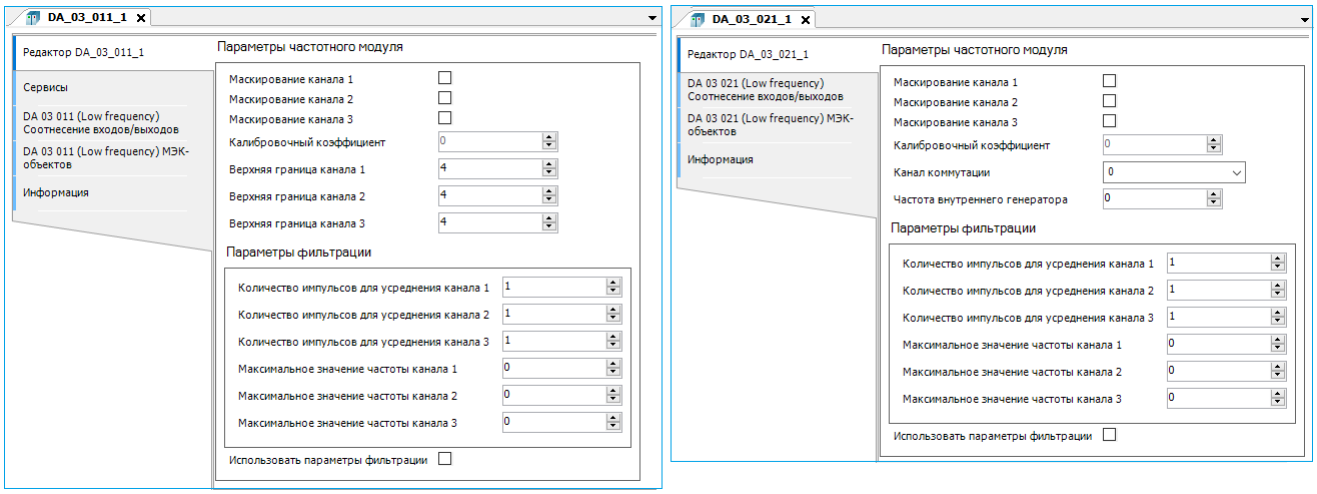


Рисунок 112 – Параметры каналов модулей на примере модуля R500 DA 03 011 и модуля R600 DA 03 021

Для настройки доступны следующие параметры (Рисунок 112):

- **Параметры частотного модуля:**
  - **Маскирование канала 1 (канала 2, канала 3)** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован;
  - **Калибровочный коэффициент** – используется для расчета частоты.
  - **Верхняя граница канала 1 (канала 2, канала 3)** – верхний порог срабатывания канала 1, канала 2, канала 3 [В]. Диапазон [4-18];
  - **Канал коммутации** – канал для коммутации внутреннего генератора частоты. По умолчанию установлено значение 0 – никакой. При указании значения от 1 до 3 происходит замыкание внутреннего генератора частоты на один из импульсных входов (это нужно для самотестирования алгоритма);
  - **Частота внутреннего генератора**. Диапазон [0 – 10 000];
- **Параметры фильтрации:**
  - **Количество импульсов для усреднения канала 1 (канала 2, канала 3)** – определение среднего значения периода. Используется для расчета измеряемой частоты. Диапазон [1:240];
  - **Максимальное значение частоты канала 1 (канала 2, канала 3)** – максимальное значение частоты, Гц;

- **Использовать параметры фильтрации** – при установлении флажка активируются Параметры фильтрации.

Настроечные параметры модуля (энкодер)

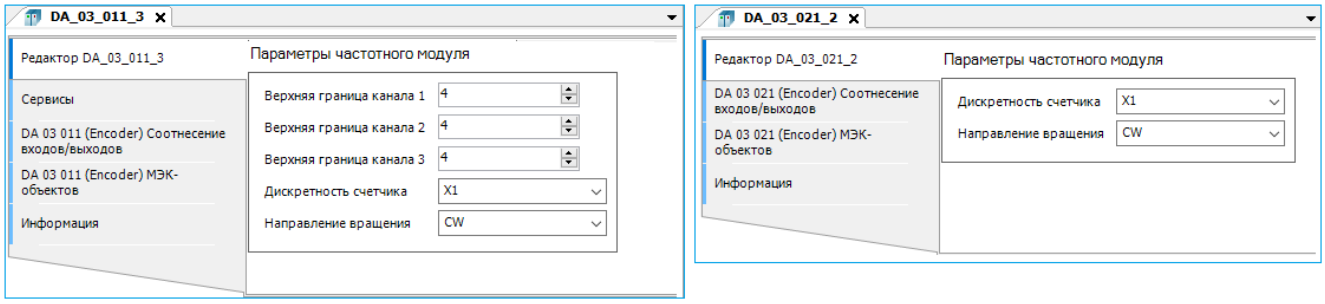


Рисунок 113 – Параметры каналов модулей на примере модуля R500 DA 03 011 и модуля R600 DA 03 021

Для настройки доступны следующие параметры (Рисунок 113):

– **Верхняя граница канала 1 (канала 2, канала 3)** – верхний порог срабатывания канала 1, канала 2, канала 3 [В]. Диапазон [4-18];

– **Дискретность счетчика**. Возможные значения: *X1* – только по передним фронтам линии А, *X2* - только по передним и задним фронтам линии А, *X3* – по передним и задним фронтам линии А и линии В;

– **Направление вращения**. Возможные значения: *CW* – по часовой стрелке, *CCW* – против часовой.

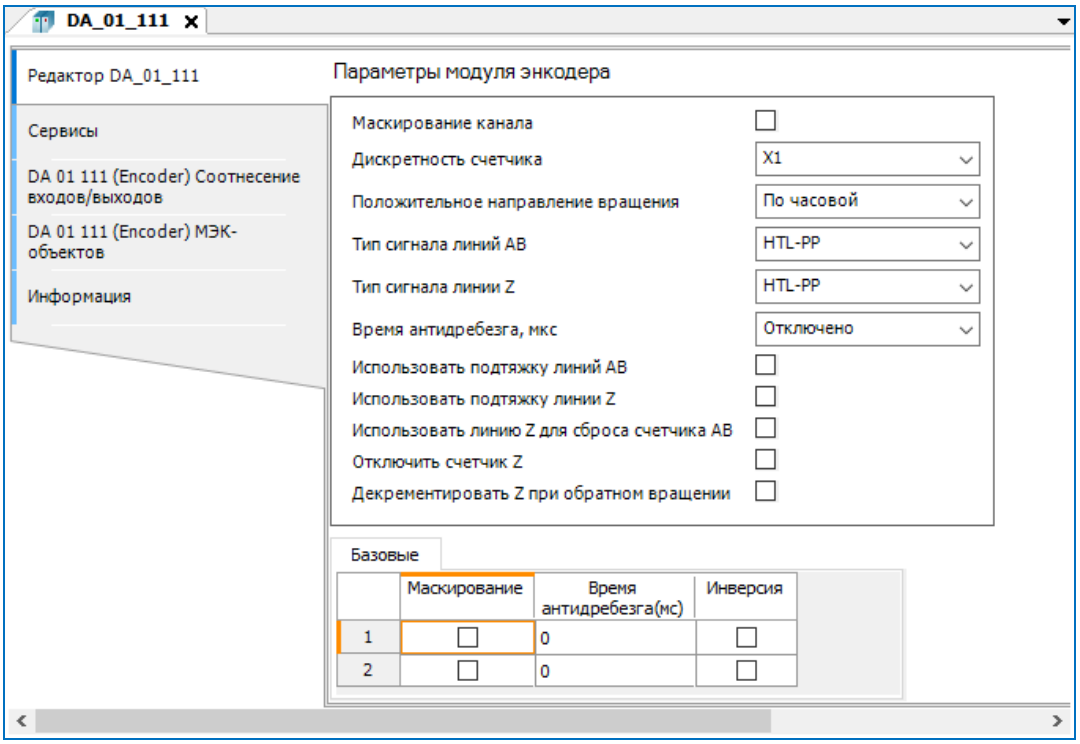


Рисунок 114 – Параметры каналов модуля счета импульсов R200 DA 01 111

Для настройки канала счета модуля R200 DA 01 111 доступны следующие параметры (Рисунок 114):

- **Маскирование канала 1** – установка флажка в этом поле задает маскирование канала счета, то есть канал не обрабатывается. По умолчанию установлено значение *0* (пустое поле) – канал не замаскирован;
- **Дискретность счетчика**. Возможные значения: *X1* – только по передним фронтам линии A, *X2* - только по передним и задним фронтам линии A, *X4* – по передним и задним фронтам линии A и линии B;
- **Положительное направление вращения**. Возможные значения: по часовой стрелке (CW), против часовой (CCW). Счет в прямом/обратном направлении. CW - значение счетчика увеличивается при перемещении по часовой стрелке. CCW - значение счетчика увеличивается при перемещении против часовой стрелки;
- **Тип сигнала линий A, B.** Возможные значения: HTL_PP, HTL_NPN, TTL_PP, TTL_NPN;
- **Тип сигнала линии Z.** Возможные значения: HTL-PP, HTL-NPN, TTL-PP, TTL-NPN;
- **Время антидребезга, мкс** (по линии A/B/Z) – минимальное время между сменами состояния $0 \leftrightarrow 1$, допустимое для регистрации смены состояния. Возможные значения: отключено (0); 0.025; 0.05; 0.1; 0.6; 0.8; 1.2; 1.6; 2.4; 3.2; 4.0; 4.8; 6.4; 8.0; 9.6; 12.8. Задается в мкс;
- **Использовать подтяжку линий A, B.** Возможные значения: *0* (пустое поле) – не использовать, флажок ☑ – использовать;
- **Использовать подтяжку линии Z.** Возможные значения: *0* (пустое поле) – не использовать, флажок ☑ – использовать;
- **Использовать линию Z для сброса счетчика AB.** Возможные значения: *0* (пустое поле) – не использовать, флажок ☑ – использовать (сброс счетчика A/B от каждого импульса по каналу Z);
- **Отключить счетчик Z** (оборотов). Возможные значения: *0* (пустое поле) – не использовать, флажок ☑ – использовать;
- **Декрементировать Z при обратном вращении.** При вращении в обратном направлении, уменьшать счет по линии Z. Возможные значения: *0* (пустое поле) – нет, флажок ☑ – да;

Для настройки дискретного канала (1/2) доступны следующие основные параметры:

- **Маскирование** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По умолчанию установлено значение 0 (пустое поле) – канал не замаскирован;

– **Время антидребезга (мс)** – минимальное время между сменами состояния $0 \leftrightarrow 1$, допустимое для регистрации смены состояния. Задается в мс;

– **Инверсия** - установка флажка в этом поле включает инверсию канала. По умолчанию установлено значение 0 (пустое поле) – инверсии нет.

Настроечные параметры модуля (системы измерения качества и количества нефти)

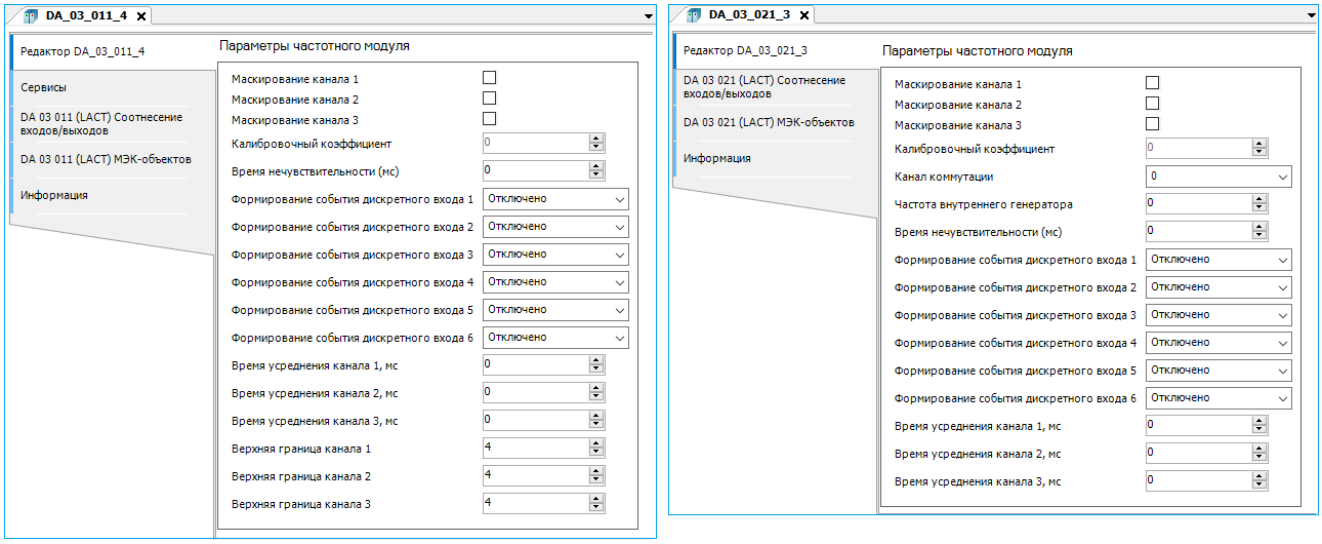


Рисунок 115 – Параметры каналов модулей на примере модуля R500 DA 03 011 и модуля R600 DA 03 021

Для настройки доступны следующие параметры (Рисунок 115):

– **Маскирование канала 1 (канала 2, канала 3)** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По умолчанию установлено значение *0* (пустое поле) – канал не замаскирован;

– **Калибровочный коэффициент** – используется для расчета частоты;

– **Верхняя граница канала 1 (канала 2, канала 3)** – верхний порог срабатывания канала 1 (канала 2, канала 3) [В]. Диапазон [4-18] (версия СПО 1.5.6.15 поддерживает расширенный диапазон [2-20]);

– **Канал коммутации** – канал для коммутации внутреннего генератора частоты. По умолчанию установлено значение *0* – никакой. При указании значения от 1 до 3 происходит замыкание внутреннего генератора частоты на один из импульсных входов (это нужно для самотестирования алгоритма);

– **Частота внутреннего генератора**. Диапазон [0 – 10 000];

– **Время нечувствительности (мс)** – время нечувствительности при формировании событий, мс;

– **Формирование события дискретного входа 1 (входа 2 ... входа 6)** – тип формирования события дискретного входа. Возможные значения: *Отключено*, *По фронту*, *По спаду*;

– **Время усреднения канала 1, мс (канала 2, канала 3)** – усреднение значений частоты. Количество импульсов для усреднения будет определяться каждый раз перед вызовом функции фильтра в зависимости от текущего значения измеренной частоты и значения времени реакции. При отсутствии импульсов частота равна 0, количество импульсов для усреднения равно 0.

Настроечные параметры модуля (автомат безопасности турбины)

Модули R600 DA 03 021, R500 DA 03 021, R200 DA 01 011 в качестве автомата безопасности предназначены для контроля скорости вращения турбины и экстренного отключения в случае разгона турбины.

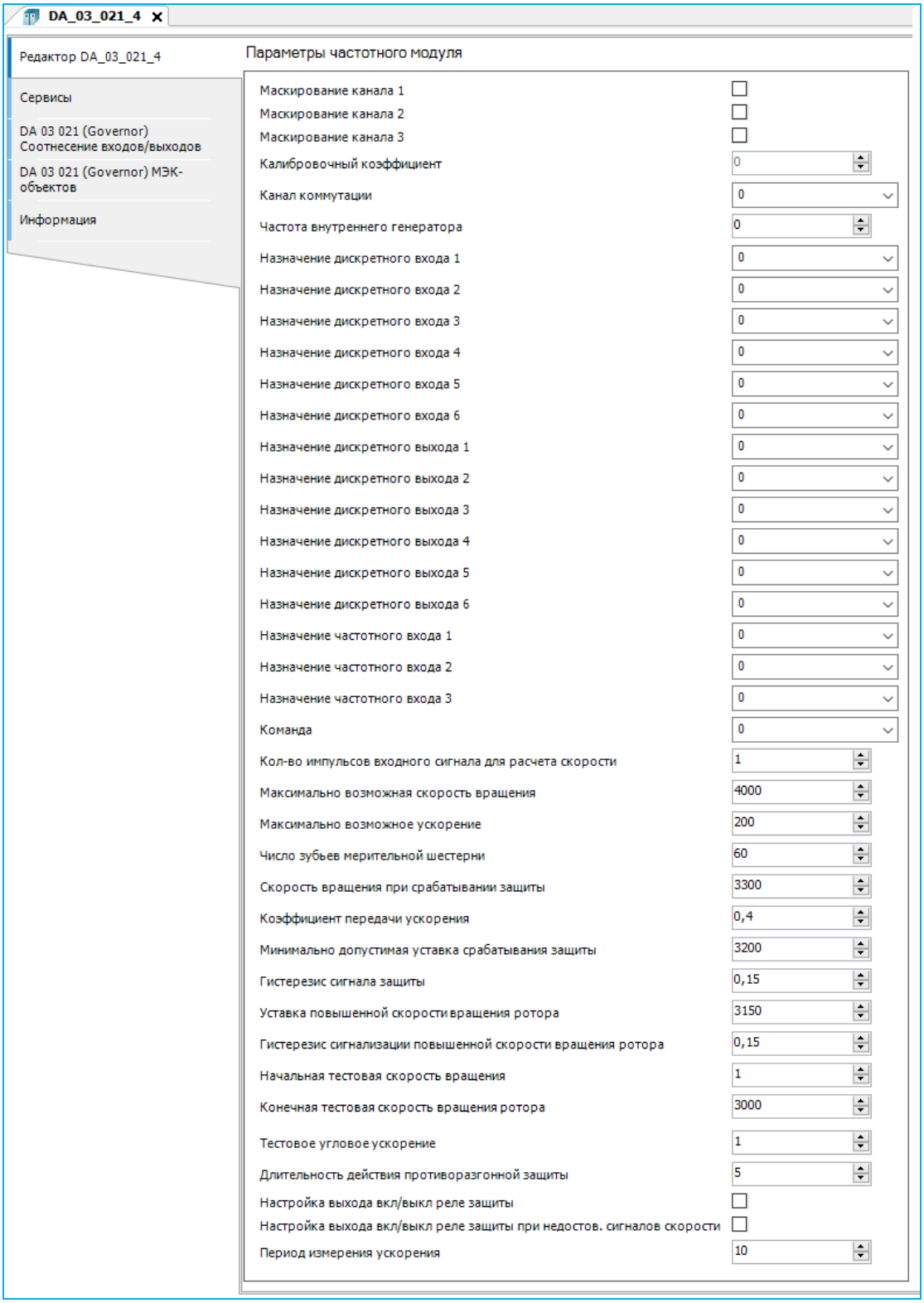


Рисунок 116 – Параметры модуля R500 DA 03 021 (автомат безопасности турбины)

Для настройки доступны следующие параметры (Рисунок 116):

– **Маскирование канала 1 (канала 2, канала 3)** – установка флажка в этом поле задает маскирование канала, то есть канал не обрабатывается. По умолчанию установлено значение *0* (пустое поле) – канал не замаскирован;

- **Канал коммутации** – канал для коммутации внутреннего генератора частоты. По умолчанию установлено значение *0* – никакой. При указании значения от 1 до 3 происходит замыкание внутреннего генератора частоты на один из импульсных входов (это нужно для самотестирования алгоритма);
- **Частота внутреннего генератора**. Диапазон [0 – 10 000]. В режиме работы модуля «Электронный автомат безопасности» данный параметр не используется. Для задания частоты генератора настраиваются поля **Начальная тестовая скорость вращения**, **Конечная тестовая скорость вращения ротора**, **Тестовое угловое ускорение**;
- **Назначение дискретного входа 1 (входа 2 ... входа 6)**. Возможные значения:
  - *0* – произвольный контроль. Вход данного типа предназначен для приема сигнала, состояние которого не анализируется в алгоритмах защиты и диагностики модуля, а только передается в технологическую программу ЦП,
  - *1* – обратный контроль включения реле защиты. Вход данного типа предназначен для приема сигнала контроля включения реле защиты, управляемого выходом «Включение реле защиты». Состояние сигнала используется в алгоритме диагностики реле защиты,
  - *2* – наличие питания цепей защиты. Вход данного типа предназначен для приема сигнала об отсутствии питания в цепях защиты. Отсутствие питания определяется уровнем сигнала 0 на входе модуля. Состояние сигнала используется в алгоритме диагностики защиты;
- **Назначение дискретного выхода 1 (выхода 2 ... выхода 6)**. Возможные значения:
  - *0* – произвольное управление. Состояние выхода данного типа не формируется в алгоритмах защиты и диагностики модуля, а задается в технологической программе ЦП,
  - *1* – срабатывание защиты. Выход данного типа сигнализирует выполнение условия для срабатывания защиты (логический уровень 1). Используется для внешней сигнализации (информационный сигнал на верхний уровень). Инициируется алгоритмом АБ,
  - *2* – включение/выключение реле защиты. Выход данного типа управляет реле защиты при срабатывании защиты (логический уровень 1 или 0, выбирается в конфигурации алгоритма защиты),
  - *3* – повышенная частота. Выход данного типа предназначен для сигнализации превышения значения частоты вращения ротора, используемого в алгоритмах защиты (задается в поле **Уставка повышенной скорости вращения ротора**), предупредительного порога (логический уровень 1),
  - *4* – неисправность (каналов измерения частоты либо цепей защиты). Выход данного типа сигнализирует недостоверность значения частоты, неисправность модуля или отказ любого реле защиты (логический уровень 1);
- **Назначение частотного входа 1 (входа 2, входа 3)**. Возможные значения:

- *0* – произвольное измерение. Вход данного типа предназначен для приема частотного сигнала, по которому рассчитываются значения скорости вращения и углового ускорения ротора, не используемые в алгоритмах защиты и диагностики,
- *1* – защитное измерение. Вход данного типа предназначен для приема частотного сигнала, по которому рассчитываются значения скорости вращения и углового ускорения ротора, используемые в алгоритмах защиты и диагностики;

– **Команда** – команда (числовая кодировка). Возможные значения:

- *0* – нет,
- *1* – включить режим ТЕСТ1,
- *2* – включить режим ТЕСТ2,
- *3* – сброс срабатывания защиты. Сбрасываются значения *Скорость вращения при срабатывании защиты* и *Угловое ускорение ротора при срабатывании защиты*,
- *4* – сброс ошибок диагностики;
- *5* – прервать запущенный ТЕСТ1 или ТЕСТ2;

– **Кол-во импульсов входного сигнала для расчета скорости** вращения ротора. Значение по умолчанию – *1*. Данное значение используется для расчета скорости по алгоритму скользящего среднего. Рекомендуется выставлять значение, кратное количеству зубьев мерительной шестерни;

– **Максимально возможная скорость вращения** ротора, об/мин. При превышении считается, что произошла ошибка измерения. Значение по умолчанию – *4 000*;

– **Максимально возможное ускорение** – максимально возможное изменение скорости за период, (об/мин)/с. При превышении считается, что произошла ошибка измерения. Значение по умолчанию – *200*;

– **Число зубьев мерительной шестерни**. Диапазон [1 – 120]. Значение по умолчанию – *60*;

– **Скорость вращения при срабатывании защиты** – уставка срабатывания защиты при нулевом угловом ускорении ротора, об/мин. При данной частоте срабатывание защиты произойдет при нулевом ускорении. Диапазон [0,00 – 8000,00]. Значение по умолчанию – *3 300*;

– **Коэффициент передачи ускорения**. Вес ускорения в формуле, по которой определяется условие защиты. Значение по умолчанию – *0,4*;

– **Минимально допустимая уставка срабатывания защиты** – частота вращения турбины, ниже которой срабатывание не произойдет ни при каком значении ускорения (допустимом), об/мин. Диапазон [0,00 – 8 000,00]. Значение по умолчанию – *3 200*;

– **Гистерезис сигнала защиты**, %. Диапазон [0 – 100]. Значение по умолчанию – 0,15;

- **Уставка повышенной скорости вращения ротора**, об/мин. Диапазон [0 – 8 000]. Значение по умолчанию – *3 150*;
- **Гистерезис сигнализации повышенной скорости вращения ротора**, %. Диапазон [0 – 100]. Значение по умолчанию – *0,15*;
- **Начальная тестовая скорость вращения ротора**, об/мин. Диапазон [0 – 8 000]. Значение по умолчанию – *0,15*;
- **Конечная тестовая скорость вращения ротора**, об/мин. Диапазон [0 – 8 000]. Значение по умолчанию – *3 000*;
- **Тестовое угловое ускорение ротора**, (об/мин)/с. Диапазон [0 – 654] Значение по умолчанию – *1*;
- **Длительность действия противоразгонной защиты**, с. Диапазон [0 – 65 535]. Значение по умолчанию – *5*;
- **Настройка выхода Вкл/выкл реле защиты**. Возможные значения:
  - *0* – при срабатывании защиты контакт DO размыкается,
  - *1* – при срабатывании защиты контакт DO замыкается;
- **Настройка выхода Вкл/выкл реле защиты при недостов. сигналов скорости.** Возможные значения:
  - *0* – только сигнал «Недостоверность»,
  - *1* – сигнал «Недостоверность» и срабатывание реле защиты;
- **Период измерения ускорения**, мс. Диапазон [1 – 250]. Значение по умолчанию – *10*.

Задание параметров модулей коммуникационного процессора

Модули коммуникационного процессора конфигурируются исходя из функционального назначения (более подробное описание приведено в руководствах пользователя по настройке обмена данными для соответствующего протокола). Модули предназначены для:

- организации каналов связи по интерфейсу RS-485 (реализуемые протоколы: ГОСТ Р МЭК 60870-5-101 (Master/Slave), Modbus RTU (Master/Slave) PROFIBUS DP, Foundation Fieldbus H1);
- организации каналов связи по интерфейсу Ethernet (реализуемые протоколы: ГОСТ Р МЭК 60870-5-104 (Master/Slave), Modbus TCP (Master/Slave));
- расширения внутренней шины данных.

Параметры модулей с интерфейсом RS-485 на примере R500 CP 04 011(Рисунок 117). Представлены системные параметры, с атрибутом *Только для чтения* и используются средой исполнения контроллера для идентификации самого модуля и его типа.

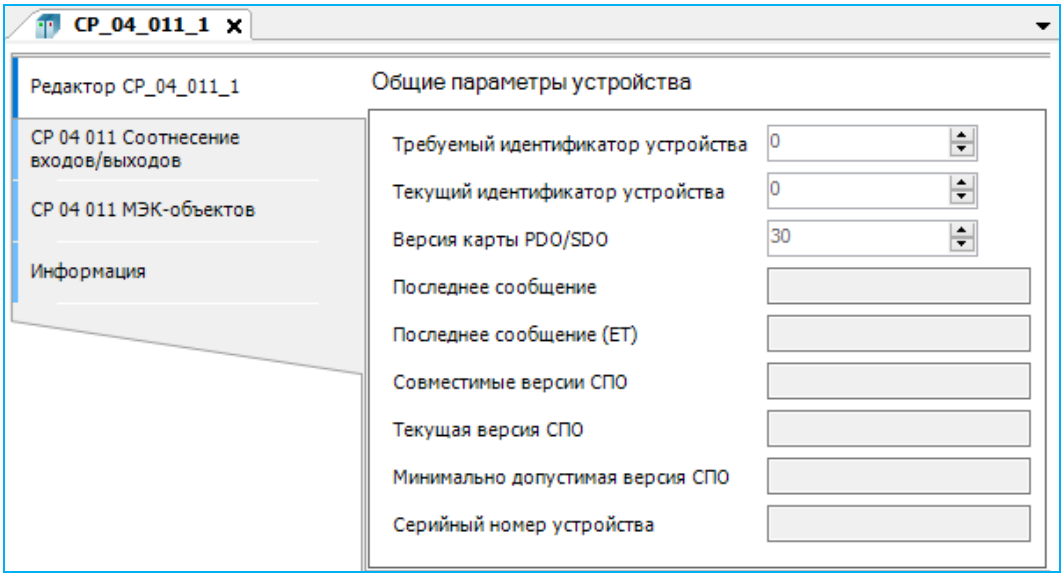


Рисунок 117 – Параметры коммуникационного модуля CP 04 011

Параметры модулей с интерфейсом Ethernet на примере R500 CP 02 021 и R200 CP 01 021 (Рисунок 118)

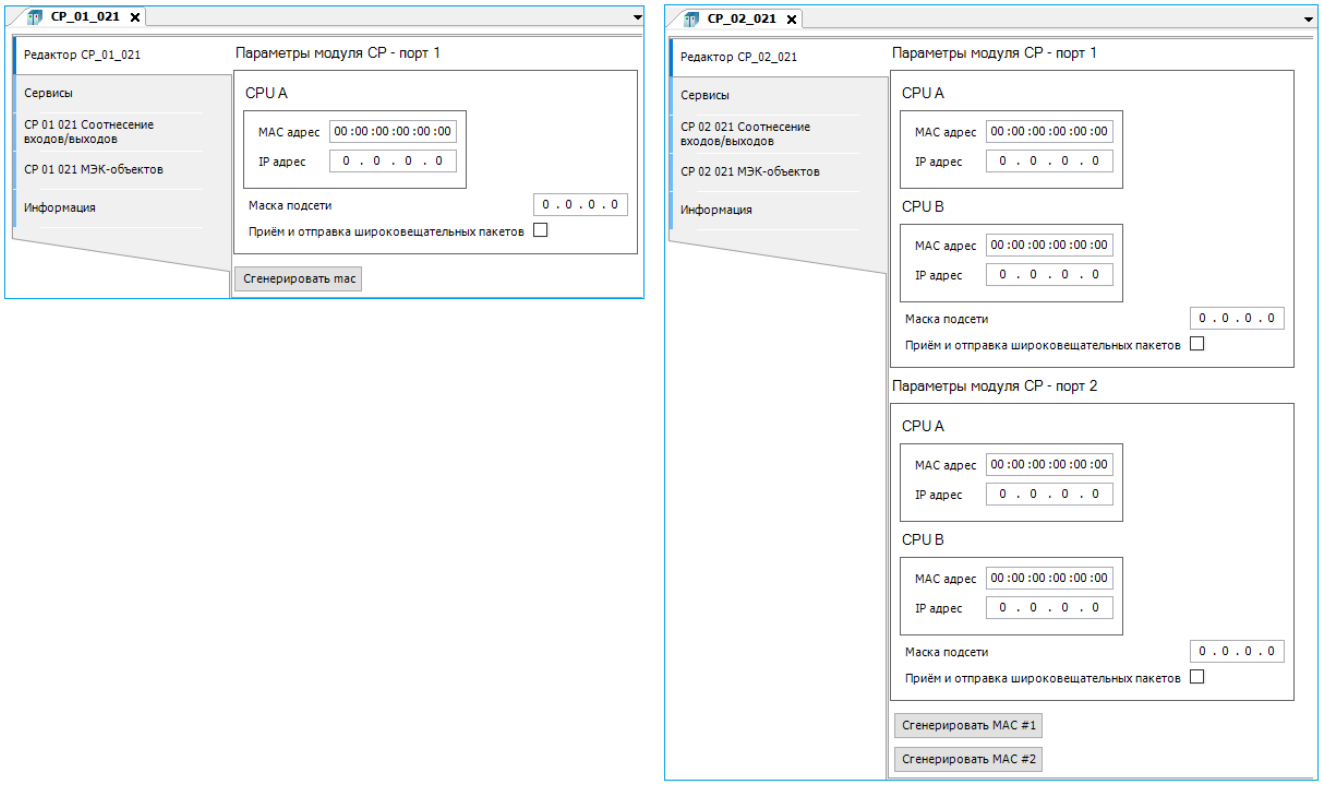


Рисунок 118 – Параметры коммуникационного модуля R200 CP 01 021 и R500 CP 02 021

Для настройки каждого канала связи (порт 1/2, секции CPU A/B) доступны следующие параметры:

– **MAC-адрес** –уникальный идентификатор устройства в сети.

– *MAC-адрес* можно назначить автоматически, для этого нажмите на кнопку **Сгенерировать MAC (#1/2)**;

- **IP-адрес** – адрес устройства;
- **Маска подсети** – битовая маска, определяющая, какая часть IP-адреса узла сети относится к адресу сети, а какая — к адресу самого узла в этой сети;
- **Прием и отправка широковещательных пакетов** – при установке галочки производится выключение функции фильтрации широковещательного трафика, как входящего, так и исходящего.

i	**ИНФОРМАЦИЯ** CPU A работает по первой шине RegulBus, CPU B – по второй шине RegulBus. Подробнее см. документ «Конфигурирование резервированной системы на контроллерах серии REGUL RX00»

Параметры модулей расширения внутренней шины данных на примере R500 CP 06 111 (Рисунок 119).

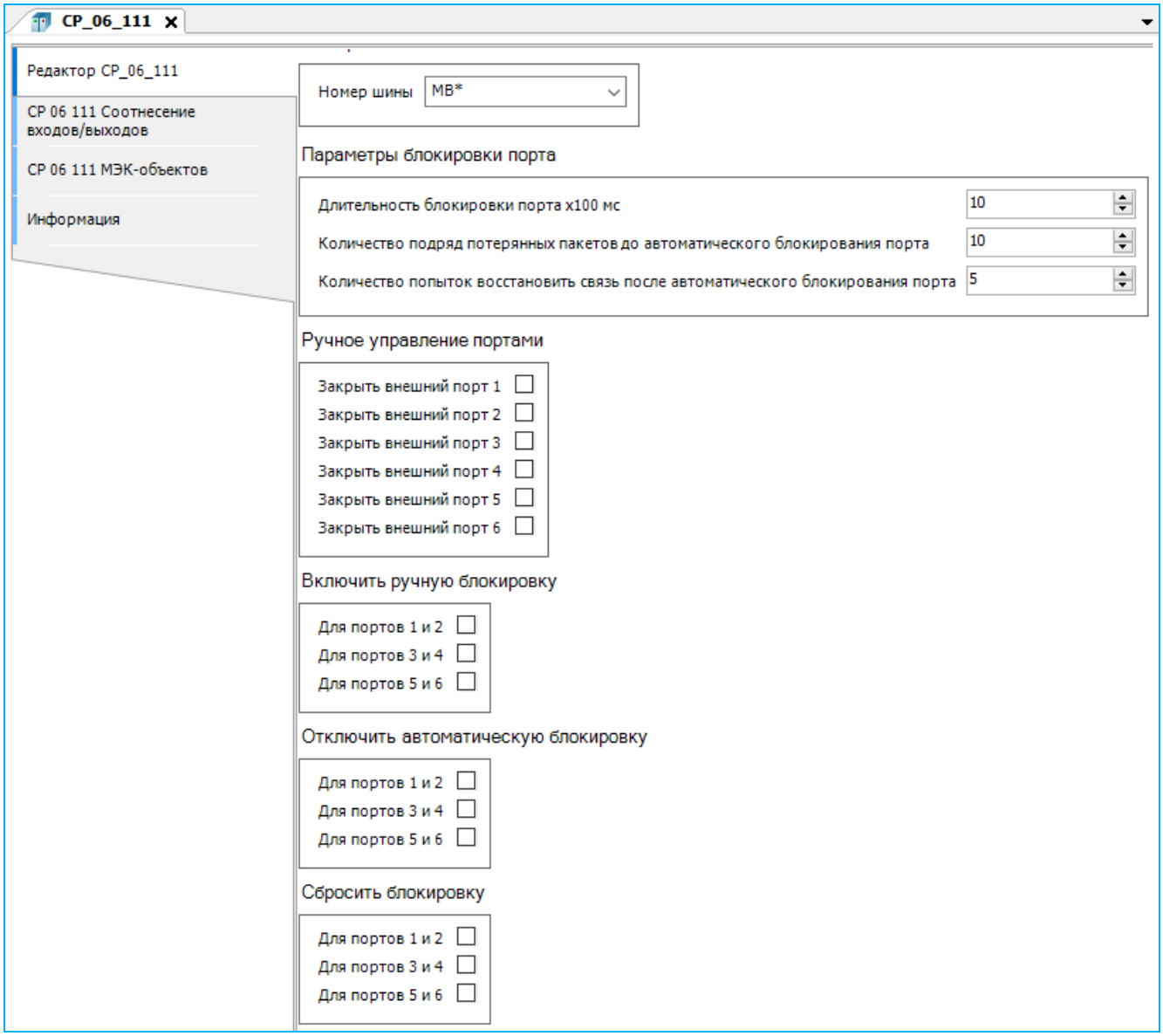


Рисунок 119 – Параметры коммуникационного модуля R500 CP 06 111 (аналогично для Regul R000 CP 06 1X1)

Для настройки каналов расширения внутренней шины данных доступны следующие параметры:

- **Номер шины** – модуль работает только по одной из внутренних шин данных: MB1 – первая внутренняя шина данных, соответственно MB2 – вторая (по умолчанию установлена: MB* – любая);
- **Параметры блокировки порта:**
  - **Длительность блокировки порта x100 мс** – указывается временной интервал блокировки порта, значение кратно 100 мс (по умолчанию 1000 мс);
  - **Количество подряд потерянных пакетов до автоматического блокирования порта** – указывается ограничение по числу потерянных пакетов данных внутренней шины, необходимых для автоматической блокировки порта (по умолчанию 10 пакетов);
  - **Количество попыток восстановить связь после автоматического блокирования порта** – указывается ограничение по числу повторных попыток, по окончании которых, порт блокируется до момента разблокировки пользователем (по умолчанию 5).

В области **Ручное управление портами** можно принудительно закрыть/открыть любой внешний порт. Для этого необходимо установить/снять галочку напротив выбранного порта.

Исходя из задаваемых значений параметров, расположенных в области **Параметры блокировки порта**, производится настройка процедуры автоматического блокирования сегментов сети. Сегменты сети (далее блоки 1/2/3, сгруппированы, соответственно, по портам: 1 и 2/ 3 и 4/ 5 и 6) блокируются временно или окончательно. Временная блокировка происходит, когда превышено количество подряд потерянных пакетов. Окончательная блокировка происходит, когда превышено количество попыток восстановить связь.

В области **Включить ручную блокировку** можно принудительно включить/отключить блокировку любого блока 1/2/3. Для этого необходимо установить/снять флажок напротив выбранного блока. По аналогии, в области **Отключить автоматическую блокировку** можно принудительно разрешить/запретить автоматическую блокировку выбранного блока.

В области **Сбросить блокировку** можно принудительно сбросить алгоритм блокировки сегмента сети, при условии, что сняты галочки в области **Включить ручную блокировку.** Для сброса необходимо установить флажок напротив выбранного блока.

Параметры модуля с интерфейсом RS-485 для обмена данными по протоколу PROFIBUS DP, на примере модуля R500 CP 01 031 (Рисунок 120). Поддержка модуля реализована только на версии внутренней шины данных RegulBus OS.

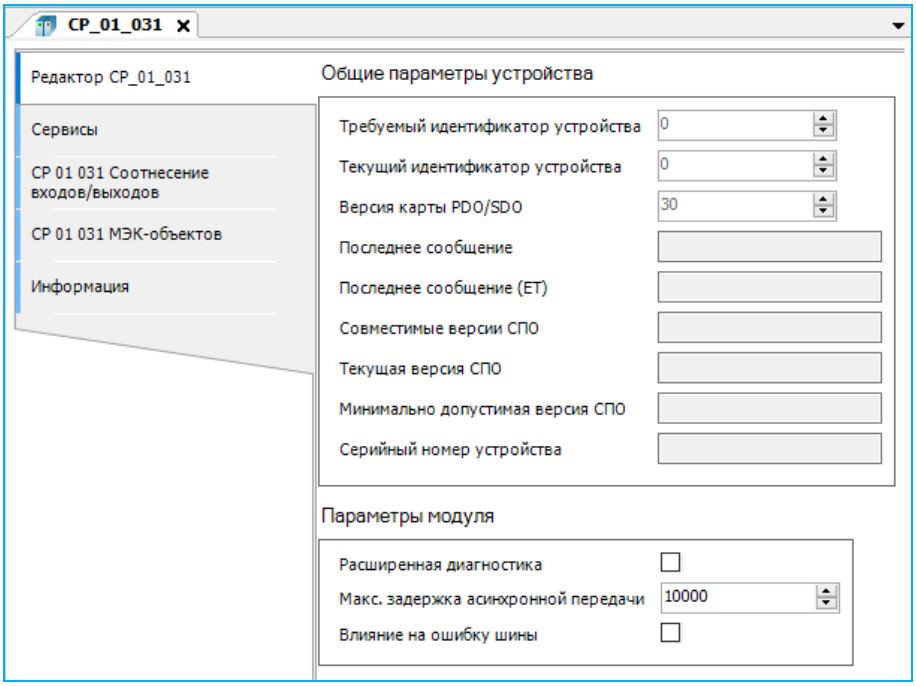


Рисунок 120 – Параметры коммуникационного модуля R500 CP 01 031

Для настройки модуля доступны следующие параметры:

– **Расширенная диагностика** - установка флажка позволяет включить режим обновления диагностической информации сконфигурированных ведомых устройств. Расширенная диагностика предоставляет информацию о текущем состоянии ведомого устройства согласно спецификации PROFIBUS DP (IEC 61158-6);

– **Макс.задержка асинхронной передачи** - максимально допустимое время, в течение которого ведомые устройства должны сформировать ответ на асинхронный запрос, по истечении данного времени осуществляется повтор запроса, количество повторов - 5, по исчерпании количества повторов выставляется ошибка асинхронного запроса;

– **Влияние на ошибку шины («Impact on bus HwError»)** – можно принудительно включить/отключить формирование ошибки (HwError) для модуля CP, в случае выхода из строя одного из опрашиваемых ведомых устройств модулем. По умолчанию (флажок отсутствует) неисправное ведомое устройство не влияет на формирование ошибки модуля CP и передачи ее по шине данных.

Параметры модуля с интерфейсом RS-485 для обмена данными по протоколу Foundation Fieldbus H1 на примере R500 CP 04 041 (Рисунок 121). Поддержка модуля реализована только на версии внутренней шины данных RegulBus OS и только модулями ЦП I-го, III-го типа.

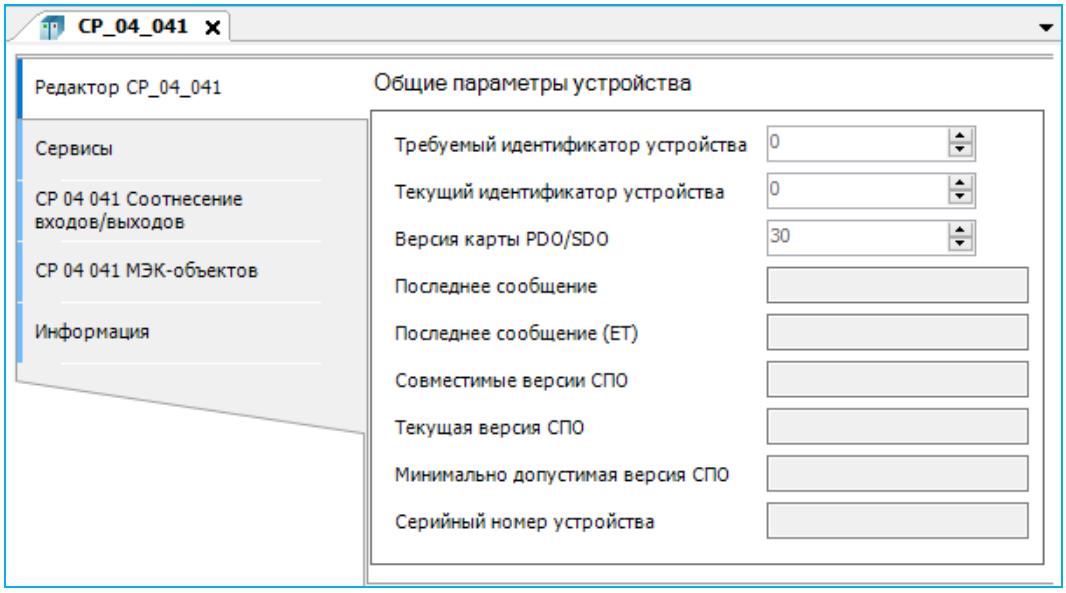


Рисунок 121 – Параметры коммуникационного модуля R500 CP 04 041

Представлены системные параметры, с атрибутом *Только для чтения* и используются средой исполнения контроллера для идентификации самого модуля и его типа.

Задание параметров модулей центрального процессора

На вкладке редактора модуля ЦП присутствуют системные параметры, с атрибутом *Только для чтения* (Рисунок 122). В случае с резервированным контроллером (подробнее см. документ «Конфигурирование резервированной системы на контроллерах серии REGUL RX00. Руководство пользователя»), на вкладке необходимо указать номер шины для каждого ЦП-партнера:

– **Номер шины** – модуль ЦП работает только по одной из внутренних шин данных: **MB1** – первая внутренняя шина данных (**CPU A**), соответственно **MB2** – вторая (**CPU B**), по умолчанию установлена: MB* – любая.

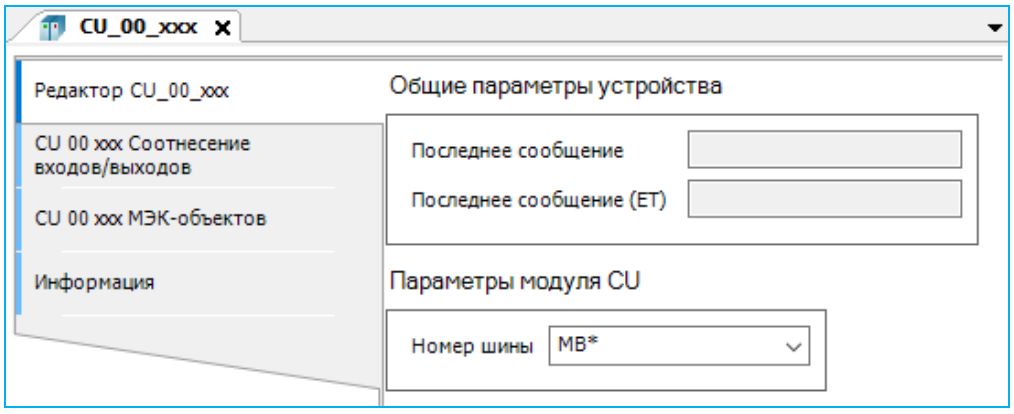


Рисунок 122 – Параметры модуля центрального процессора R500

Задание параметров блоков расширения EU для модулей ЦП III-го типа

На вкладке редактора блока расширение присутствует следующий параметр (Рисунок 123):

– **Влияние на ошибку шины** – можно принудительно включить/отключить влияние выхода из строя блока расширения на работу контроллера. По умолчанию (флажок отсутствует) неисправный блок расширения не влияет на формирование ошибки модуля ЦП по шине данных.

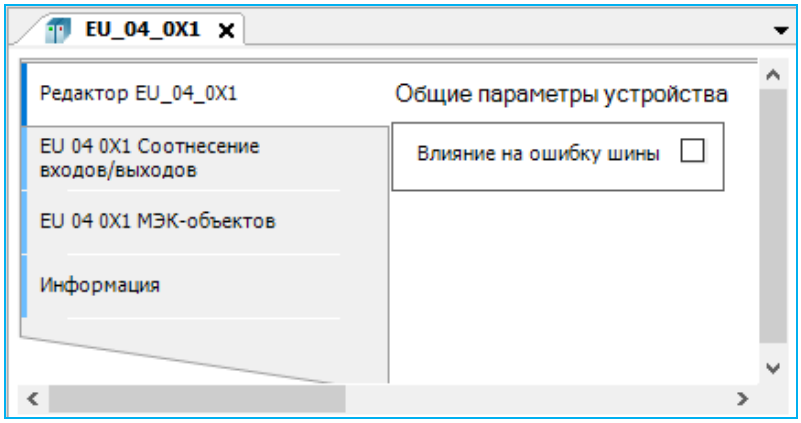


Рисунок 123 – Параметры блока расширения EU модуля ЦП R500 III-го типа

Привязка каналов к переменным программы

Общие сведения

Модули контроллера имеют определенное количество логических каналов ввода/вывода, к которым можно привязать переменные прикладной программы. Некоторые из этих логических входов/выходов соответствуют тем или иным «физическим» входам/выходам модуля, а некоторые привязаны к внутренним регистрам модуля. Как и в случае с параметрами модулей, логические входы/выходы также доступны для конфигурирования пользователем в среде разработки Astra.IDE. Для этого в редакторе модуля перейдите на вкладку **Соотнесение входов/выходов** (Рисунок 124).

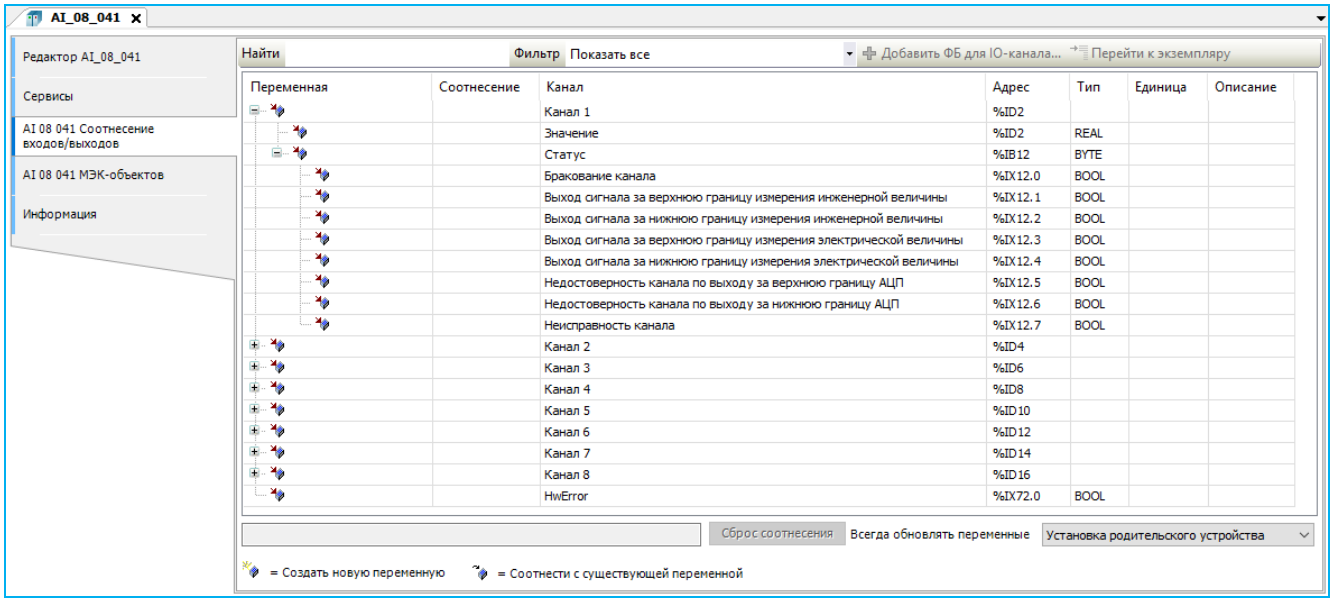


Рисунок 124 – Соотнесение входов/выходов

При соотнесении переменных и входов/выходов необходимо учитывать следующее:

– к переменным, соотносимым со входом, нельзя обращаться посредством записи;

– существующая переменная может быть соотнесена только с одним входом;

– тип канала обозначается в столбце **Переменная** иконкой для входа и для выхода.

В зависимости от устройства может быть показана следующая информация:

– **Канал:** символьное имя входа или выхода канала устройства;

– **Адрес:** адрес канала, например, %QX3.0. Значение адреса можно изменить вручную. Это может потребоваться для изменения адресации в соответствии с данной аппаратной конфигурацией или сохранения значения адреса даже при изменении порядка модулей (по умолчанию это приводит к автоматическому изменению значений адреса). Учтите, что независимо от описания устройства вы можете изменить адрес только всего входа или выхода целиком, а не отдельных его подэлементов (бит-каналов). Таким образом, если вход или выход представлен в таблице соотнесений поддеревом, адрес можно изменить только у самого верхнего объекта. Чтобы зафиксировать значение адреса выделите его и нажмите клавишу ***Пробел***. При этом откроется поле ввода. Затем измените значение или оставьте как есть и закройте поле клавишей ***Enter***;

ВНИМАНИЕ!

Изменив адрес вручную и скомпилировав проект, в окне сообщений появится предупреждающая информация следующего вида:

«обнаружено ручное присвоение адреса для параметра…»

В случае конфликта адресов (при генерации кода), для быстрого поиска ошибки достаточно будет выполнить следующее:

- дважды щелкните левой кнопкой мыши по сообщению для открытия вкладки модуля с присвоенным вручную адресом;
- перейдите на вкладку …**Соотнесение входов/выходов**;
- дважды щелкните левой кнопкой мыши по адресу, помеченный значком Ⓜ и удалите значение. Нажмите клавишу **Enter** и произойдет автоматическая установка адреса

- **Тип:** тип данных входного или выходного канала, например, *BOOL*;
- **Единица:** единица значения параметра, например, «ms» для миллисекунд;
- **Описание:** краткое описание параметра;
- **Текущее значение параметра:** текущее значение параметра, показывается в онлайн-режиме.

Соотнесение переменных и входов/выходов

Для того чтобы привязать переменную ко входу или выходу модуля, на вкладке **Соотнесение входов/выходов** (Рисунок 125) дважды щелкните левой кнопкой мыши в строке нужного входа/выхода. Появится курсор (можно вручную ввести имя переменной) и кнопка …, открывающая окно **Ассистент ввода**.

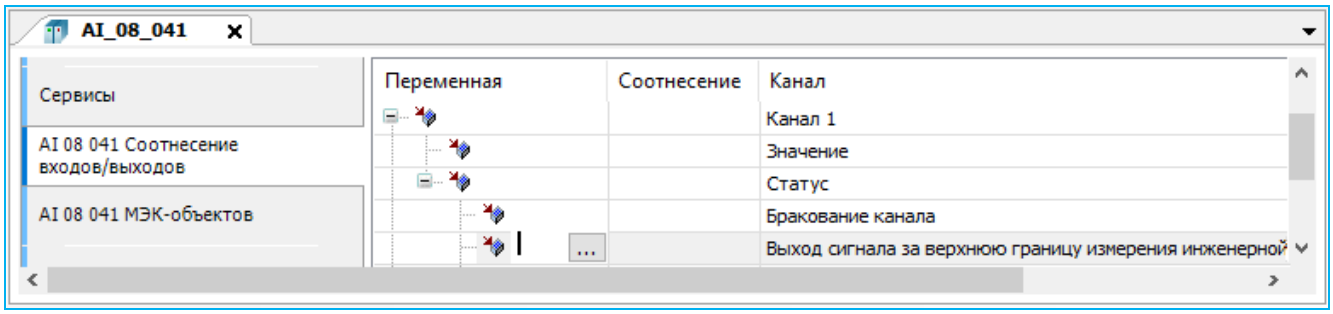


Рисунок 125 – Вызов Ассистента ввода

В окне **Ассистент ввода** (Рисунок 126) найдите нужную переменную. Если установлен флажок в поле **Структурированный вид**, то раскрывайте списки с помощью кнопки ⊞. Если флажок снят и переменные представлены одним большим списком, для удобства поиска воспользуйтесь фильтром. После выбора переменной нажмите кнопку ***ОК***, закроется окно **Ассистент ввода**, а переменная появится на вкладке **Соотнесение входов/выходов**.

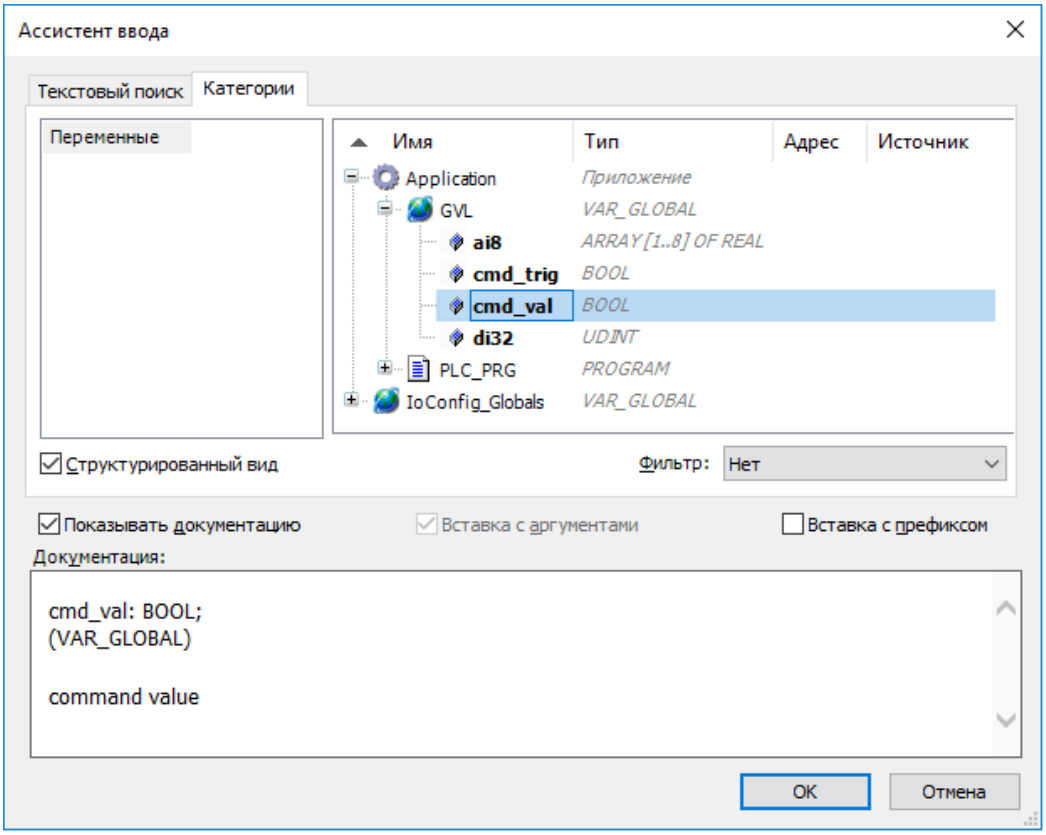


Рисунок 126 – Диалоговое окно «Ассистент ввода»

Также на вкладке **Соотнесение входов/выходов** присутствует параметр **HwError**, информирующий о статусе состояния самого модуля. Значение параметра отображается в онлайн-режиме. Параметр принимает значение *TRUE* (истина) или *FALSE* (ложь), т.е. при наличии ошибки отобразится: *TRUE*, а при отсутствии *FALSE*.

На вкладке **Соотнесение входов/выходов** двойной щелчок левой кнопкой мыши по ячейке в столбце **Описание** в строке какого-либо входа/выхода позволяет задать описание этого входа/выхода.

Чтобы сбросить все привязки переменных к входам/выходам модуля нужно нажать кнопку ***Сброс соотнесения***.

Если опция *Всегда обновлять переменные* активирована, то переменные будут обновляться в каждом цикле задачи цикла шины, вне зависимости от наличия привязки к входам/выходам. Режим постоянного обновления переменных нужен и полезен во время отладки проекта. При эксплуатации постоянное обновление большого количества переменных приводит к повышенной нагрузке на центральный процессор контроллера, поэтому режим постоянного обновления переменных не рекомендуется использовать в процессе эксплуатации.

Для модуля аналогового ввода есть описание входа, представленное несколькими каналами (в терминологии среды разработки). Это: значение, считываемое с аналогового входа, и байт

статуса, который показывает достоверность значения и набор флагов событий, связанных с выходом за границы измерений (127).

i	**ИНФОРМАЦИЯ** В прикладной программе необходимо отслеживать статусы канала и статус модуля, так как при браковании канала происходит «замораживание» величины параметра на последнем достоверном значении

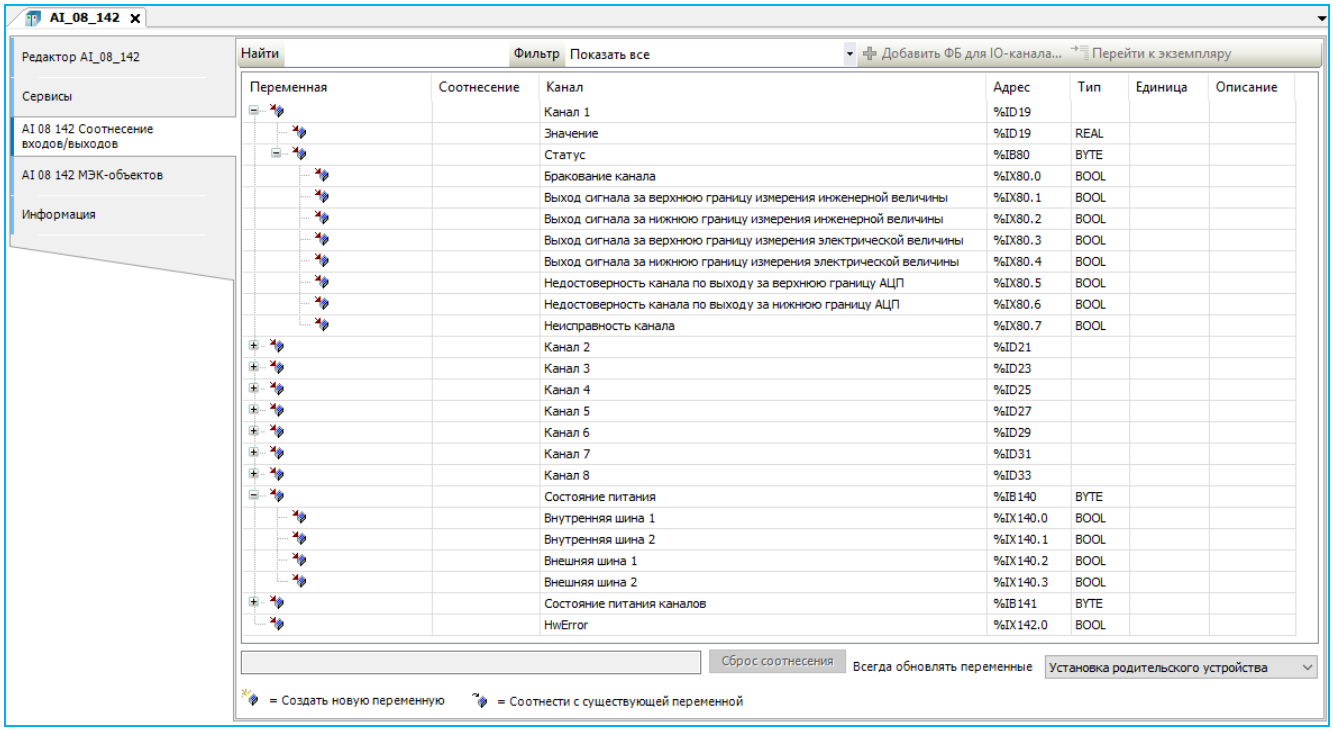


Рисунок 127 – Проверка статуса канала модуля аналогового ввода

Дополнительно, в зависимости от модуля и версии СПО, могут присутствовать:

– байт состояния питания, который может содержать флаги:
  - состояния внутренних шин питания 1 или 2;
  - состояния внешних шин питания 1 или 2;
– байт состояния каналов питания, который содержит флаги состояния каналов питания внешних цепей для каждого канала ввода (например: R500 AI 08 142, см. рисунок 127).

Для модуля аналогового вывода отображаются значения, установленные по каждому из выходов, и набор флагов состояния, которые показывают наличие/отсутствие внешнего питания и сигнализируют о возможном обрыве по каждому из каналов (Рисунок 128).

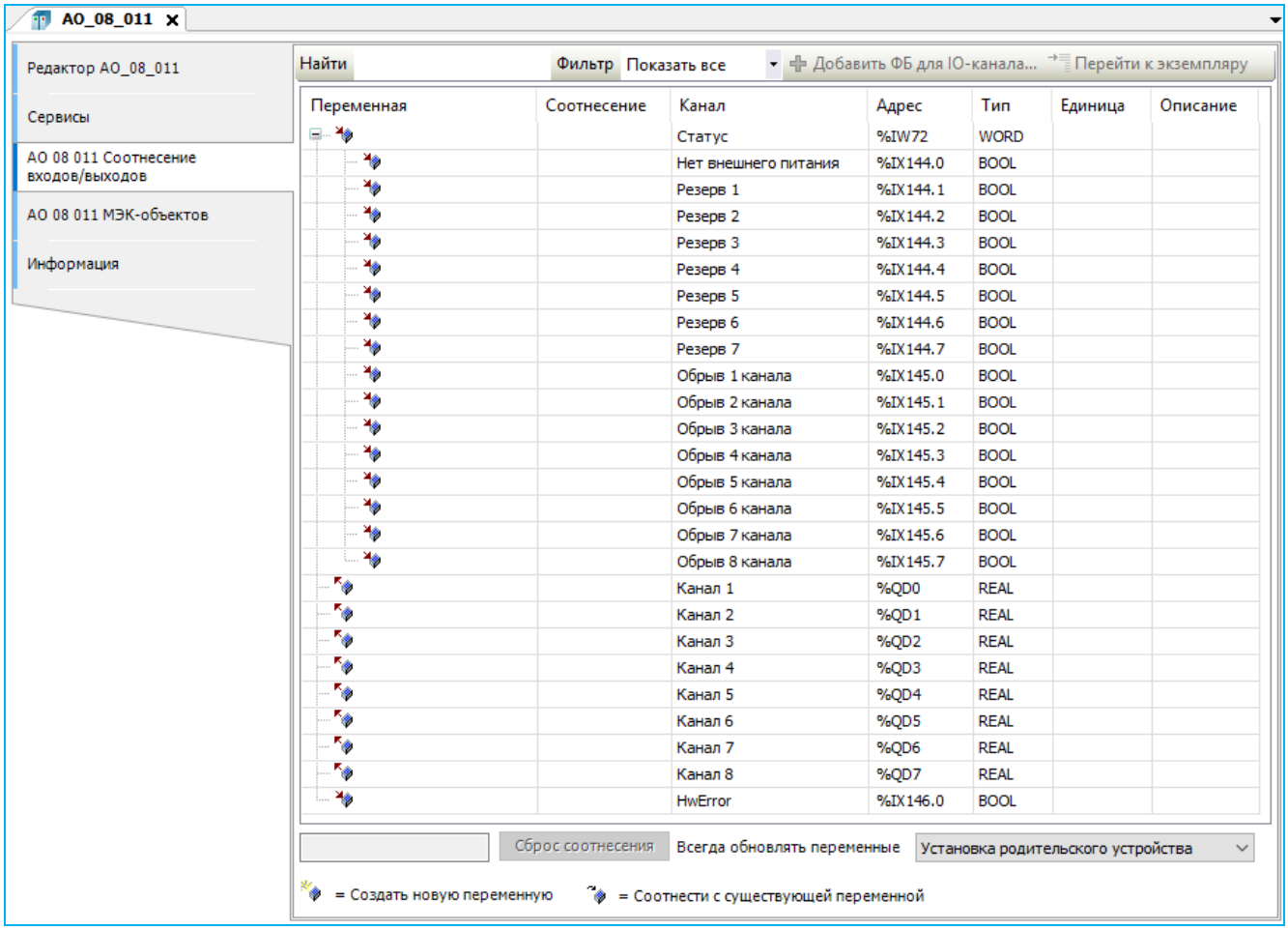


Рисунок 128 – Проверка статуса модуля аналогового вывода

Для модуля источника питания R500 PP 00 051 отображаются:

- входное напряжение;
- напряжения на внутренних шинах питания;
- ток потребления по внутренним шинам питания;
- температура внутри модуля;
- статус положения ключа переключения шины питания:
  - 0 – все шины питания отключены,
  - 1 – включена первая шина питания,
  - 2 – включена вторая шина питания,
  - 3 – подключены обе шины питания;
- байт статуса срабатывания предохранителя (при наличии дополнительного ИП на шине):
  - 0 - сработал предохранитель шины 1,
  - 1 - сработал предохранитель шины 2,
  - 2…7 - резерв.

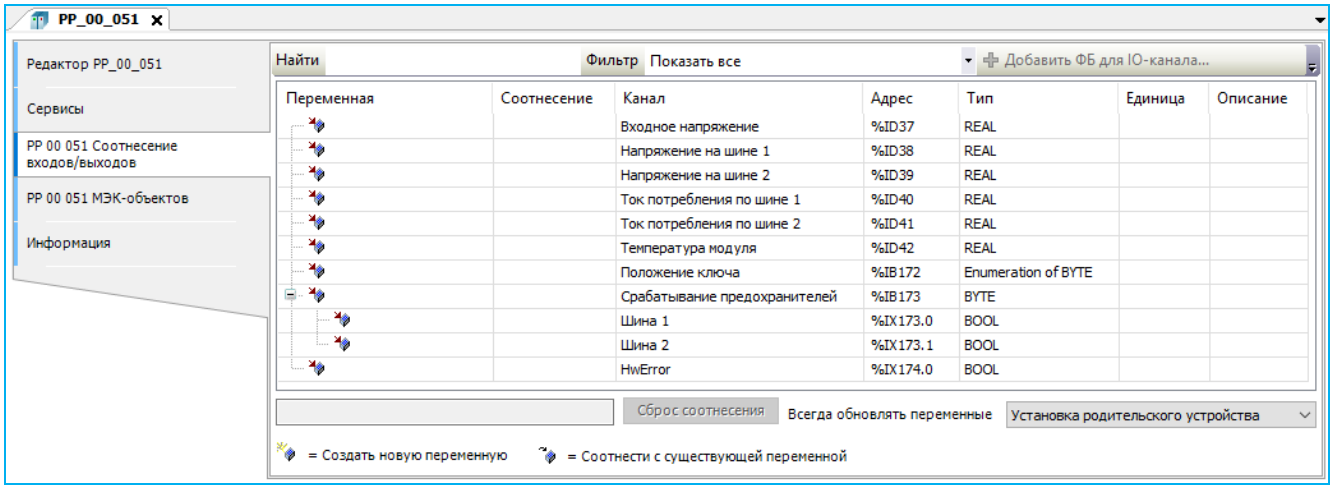


Рисунок 129 - Модуль источника питания R500 PP 00 051

Для модуля источника внешнего питания R500 PO 08 041 отображается набор статусов состояния: (Рисунок 130):

- внутреннего питания:
  - 0 – питание с внутренней шины 1,
  - 1 – питание с внутренней шины 2;
- внешнего питания:
  - 0 – питание с внешней шины 1 в допуске (ОК),
  - 1 – низкое напряжение на внешней шине 1,
  - 2 – высокое напряжение на внешней шине 1,
  - 3 – питание с внешней шины 2 в допуске (ОК),
  - 4 – низкое напряжение на внешней шине 2,
  - 5 – высокое напряжение на внешней шине 2;
- по каждому из каналов:
  - 0 – ошибка,
  - 1 – низкое напряжение,
  - 2 – высокое напряжение.

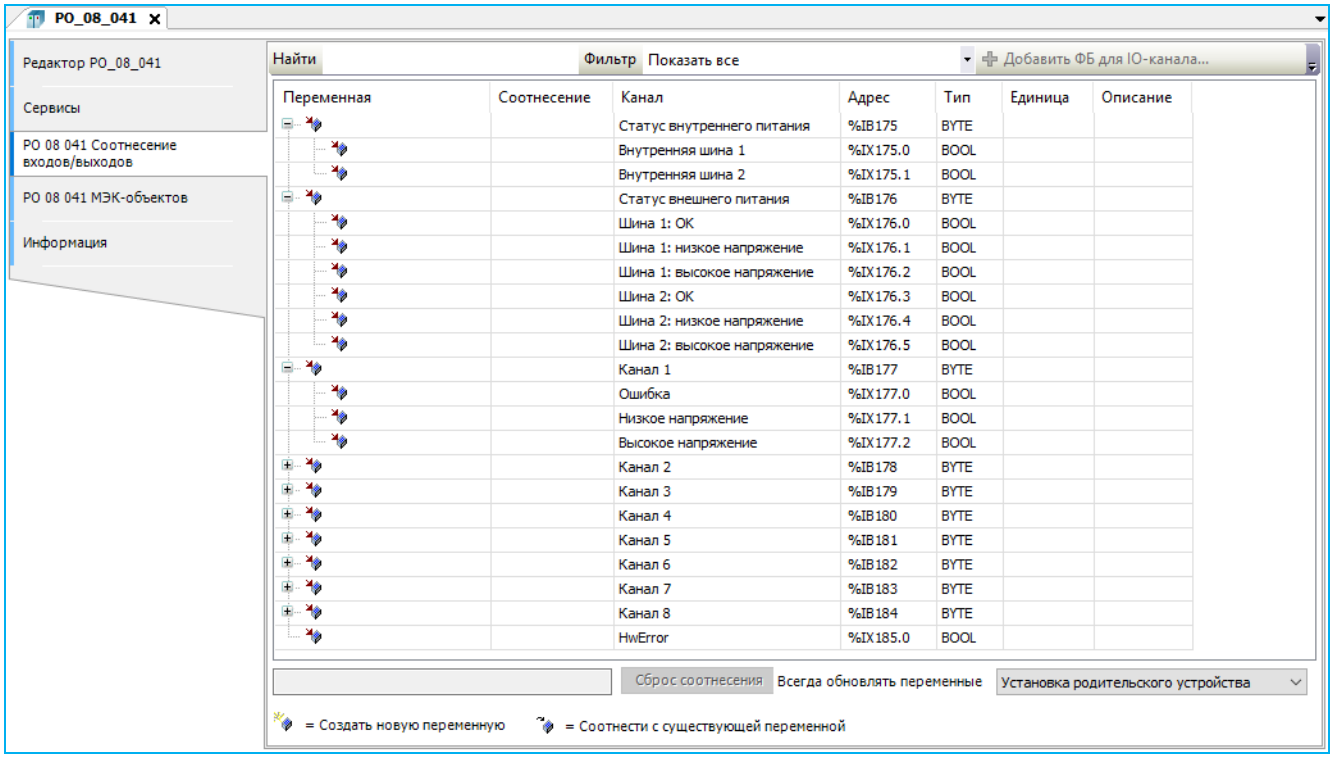


Рисунок 130 - Модуль источника внешнего питания R500 PO 08 041

Для модулей дискретного ввода отображаются биты состояния каждого канала. Дополнительно, в зависимости от модуля и версии СПО, может присутствовать байт состояния питания, который содержит флаги состояния внутренних шин питания 1 или 2.

Для модулей дискретного ввода Namur отображаются биты состояния обрыва и короткого замыкания по каждому каналу (Рисунок 131).

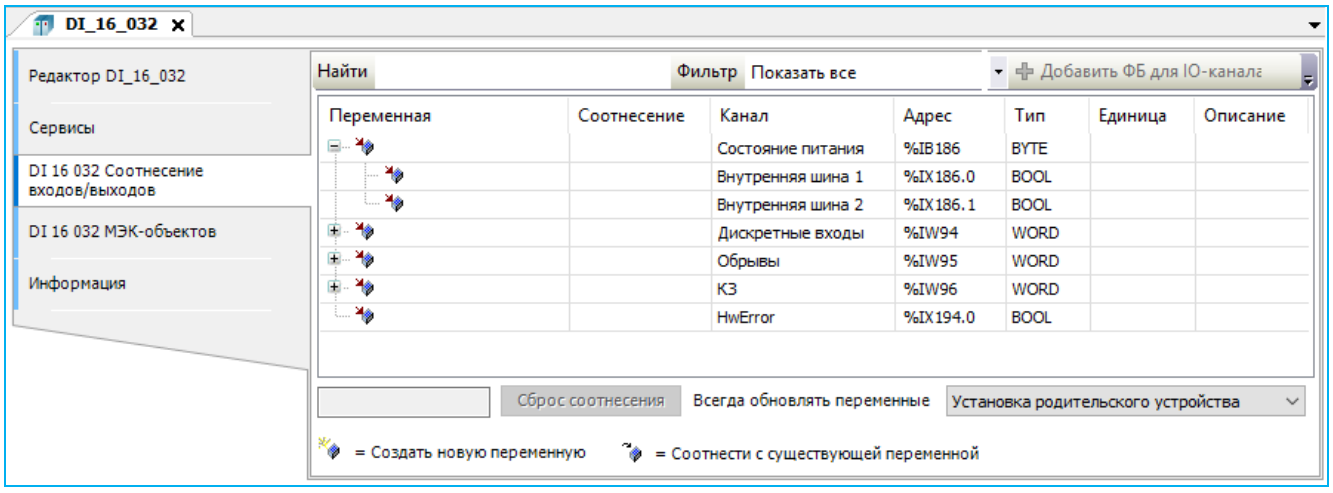


Рисунок 131 - Модуль дискретного ввода R500 DI 16 032

Для модуля коммуникационного процессора R500 CP 06 111 можно отслеживать состояние соединений, блоков и статус состояния модуля HwError (Рисунок 132).

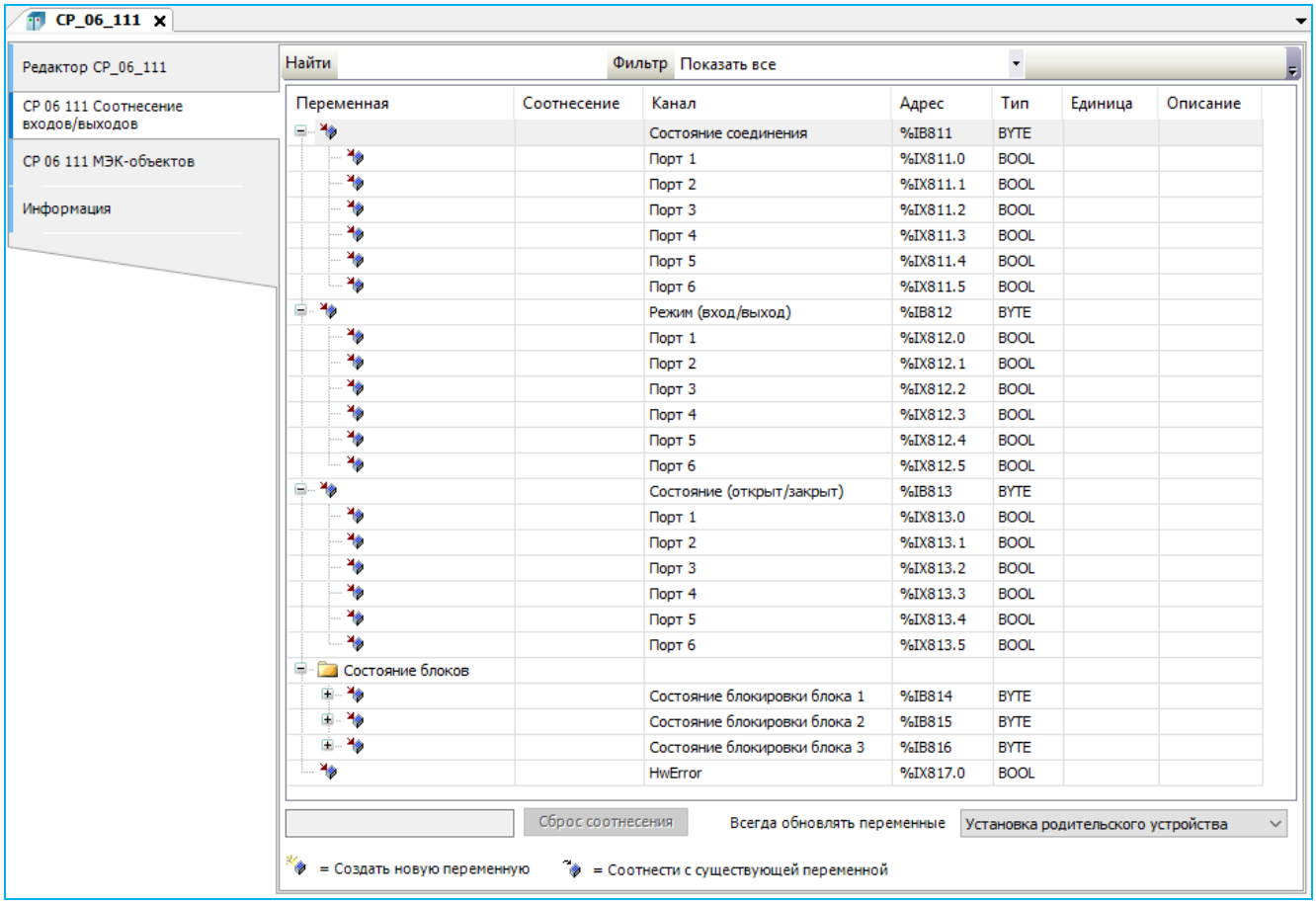


Рисунок 132 – Вкладка соотнесение входов/выходов на примере модуля CP 06 111

Дополнительно к параметрам, представленным на рисунке 132, для модулей R000 CP 06 1X1 присутствует параметр, информирующий о наличии внешнего питания на одном из каналов (1/2).

Для модуля коммуникационного процессора R500 CP 01 031 можно отслеживать следующие статусы состояния (Рисунок 133):

- состояние питания (внутреннего):
  - Бит 0 – питание внутренней шины 1 в допуске,
  - Бит 1 – питание внутренней шины 2 в допуске;
- Reliability – флаг, определяющий достоверность входных данных всех подчиненных устройств;
- статус состояния модуля HwError.

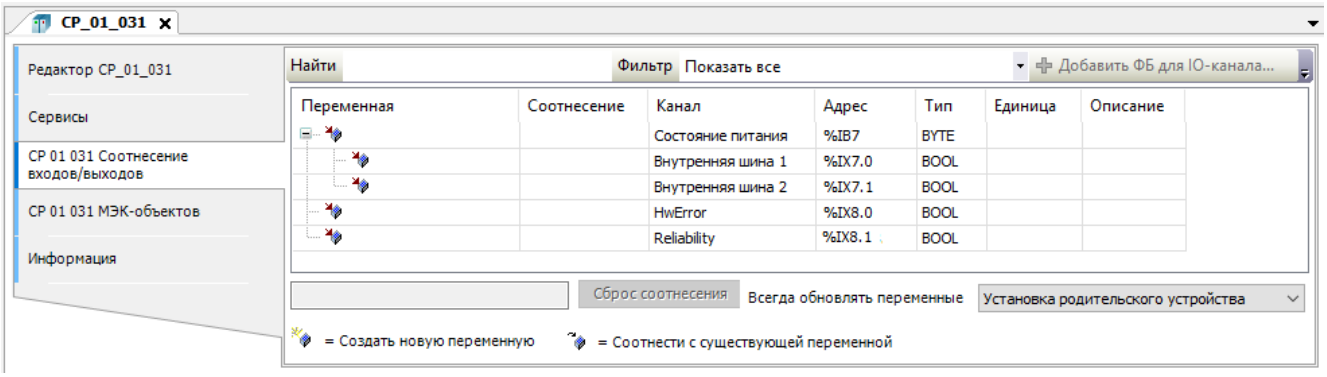


Рисунок 133 – Вкладка соотнесение входов/выходов на примере модуля CP 01 031

Для модуля коммуникационного процессора R500 CP 04 041 можно отслеживать следующие статусы состояния (Рисунок 134):

- состояние питания (внутреннего):
  - Бит 0 – питание внутренней шины 1 в допуске,
  - Бит 1 – питание внутренней шины 2 в допуске;
- статус каналов:
  - 0 бит - питание канала 1,
  - 1 бит - питание канала 2,
  - 2 бит - питание канала 3,
  - 3 бит - питание канала 4,
  - 4 бит - неисправность канала 1,
  - 5 бит - неисправность канала 2,
  - 6 бит - неисправность канала 3,
  - 7 бит - неисправность канала 4;
- статус состояния модуля HwError.

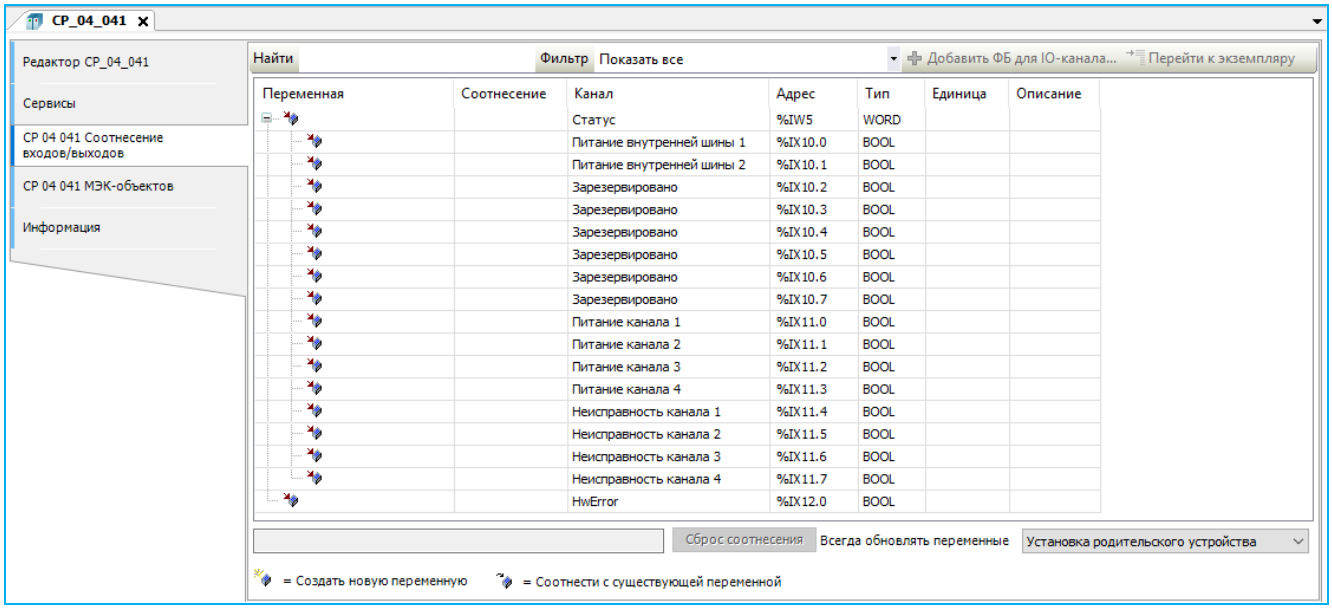


Рисунок 134 – Вкладка соотнесение входов/выходов на примере модуля CP 04 041

Для портов с интерфейсом Ethernet в модулях ЦП III-го типа (шина RegulBus OS), блоках расширения EU и коммуникационных модулях (CP 0x 021) отображаются биты состояния соединения (Link Status), на рисунке 135 приведен пример.

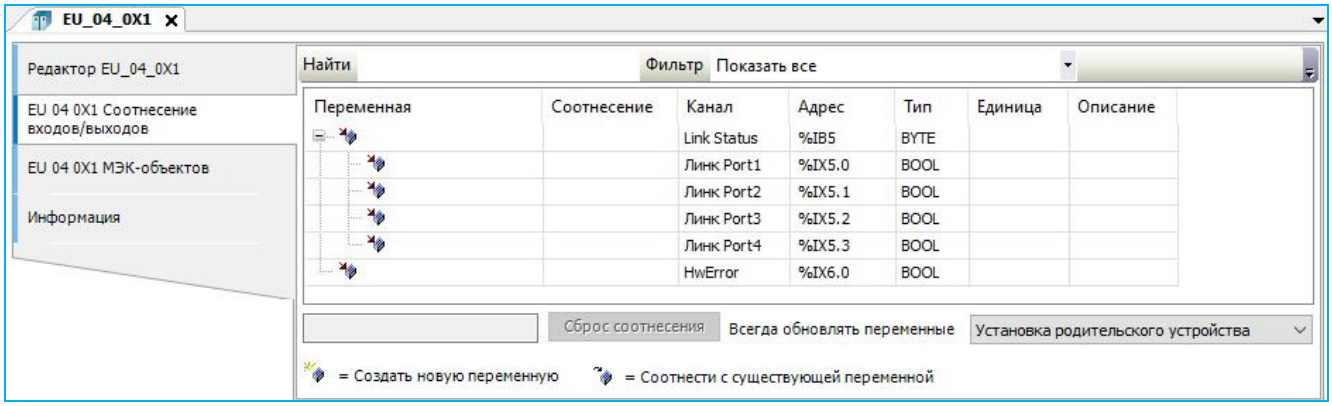


Рисунок 135 – Пример вкладки соотнесение входов/выходов блоков расширения EU

Для модулей ввода/вывода на вкладке **IEC Objects** (МЭК объекты) представлены объекты, позволяющие выполнить доступ к устройству из МЭК-приложения (Рисунок 136). С возможностью добавлять, редактировать и удалять экземпляры.

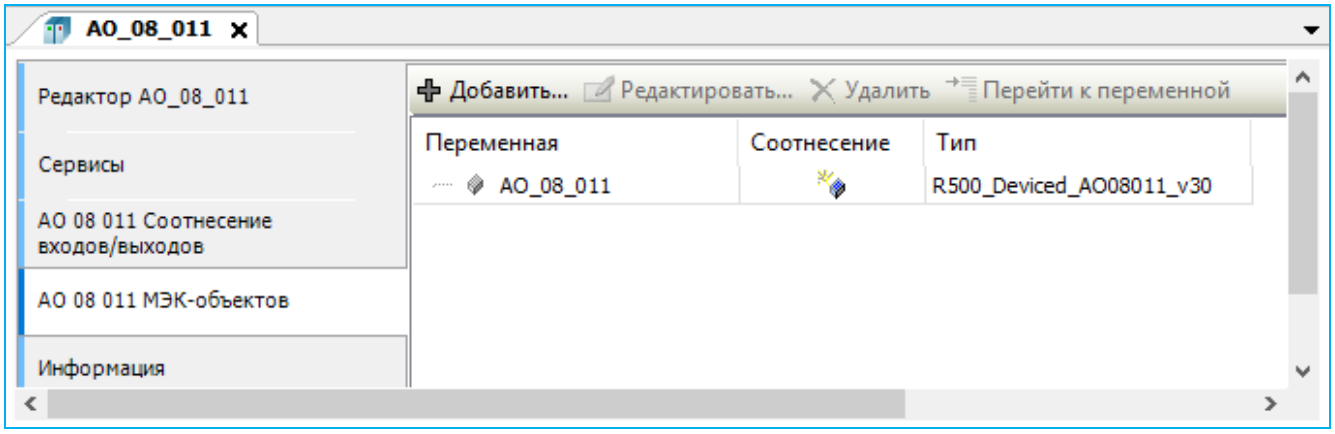


Рисунок 136 – Вкладка МЭК объекты на примере модуля AO 08 011

ПОДКЛЮЧЕНИЕ КОНТРОЛЛЕРА К СЕТИ

Сканер сети. Настройка IP-адресов

Чтобы обеспечить взаимодействие контроллера с компьютером и другими устройствами необходимо настроить его сетевые параметры. На первом шаге настройки используется специальный компонент Astra.IDE **Сканер сети**, который делает контроллер доступным для встроенных механизмов подключения.

Выберите в основном меню **Инструменты** ⇨ **Сканер сети** (Рисунок 137).

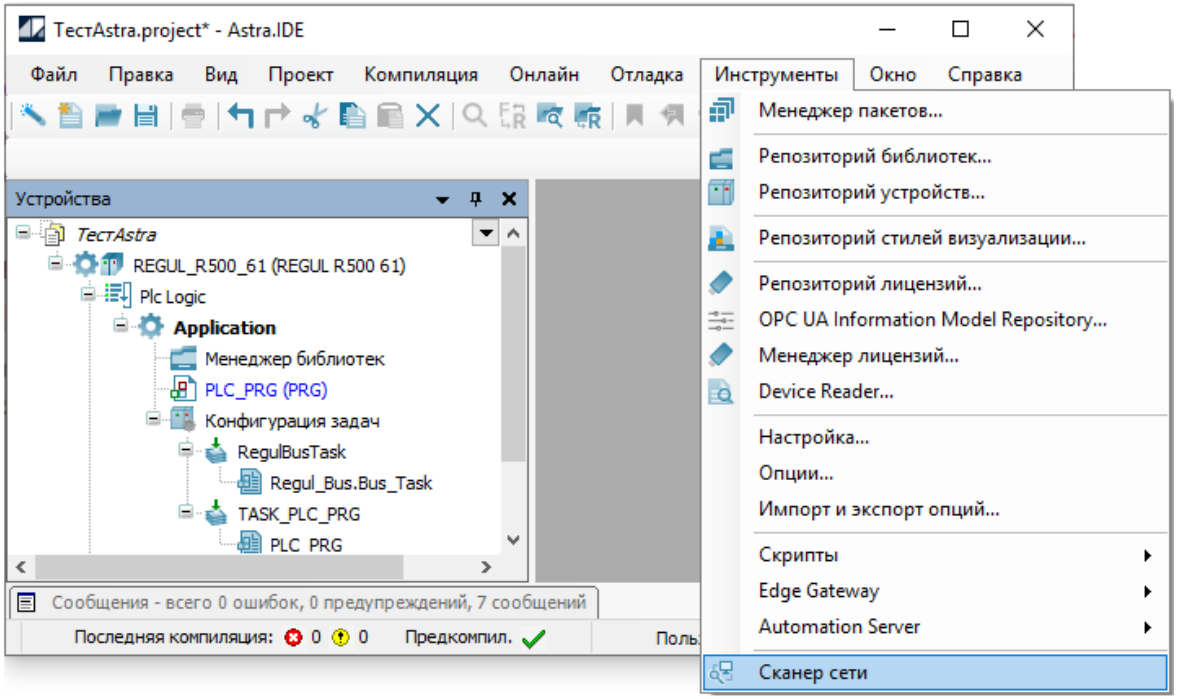


Рисунок 137 – Начало работы по определению сетевых параметров контроллера

Откроется окно **Сканер сети**. В поле **Выберите сетевое подключение:** выберите значение из раскрывающегося списка, нажмите кнопку ***Сканировать***. В результате сканирования сети в поле **Список ПЛК:** появится список всех контроллеров, подключенных к выбранной сети (Рисунок 138). Зеленым цветом выделены новые контроллеры, то есть те, которых не было в списке после предыдущего сканирования.

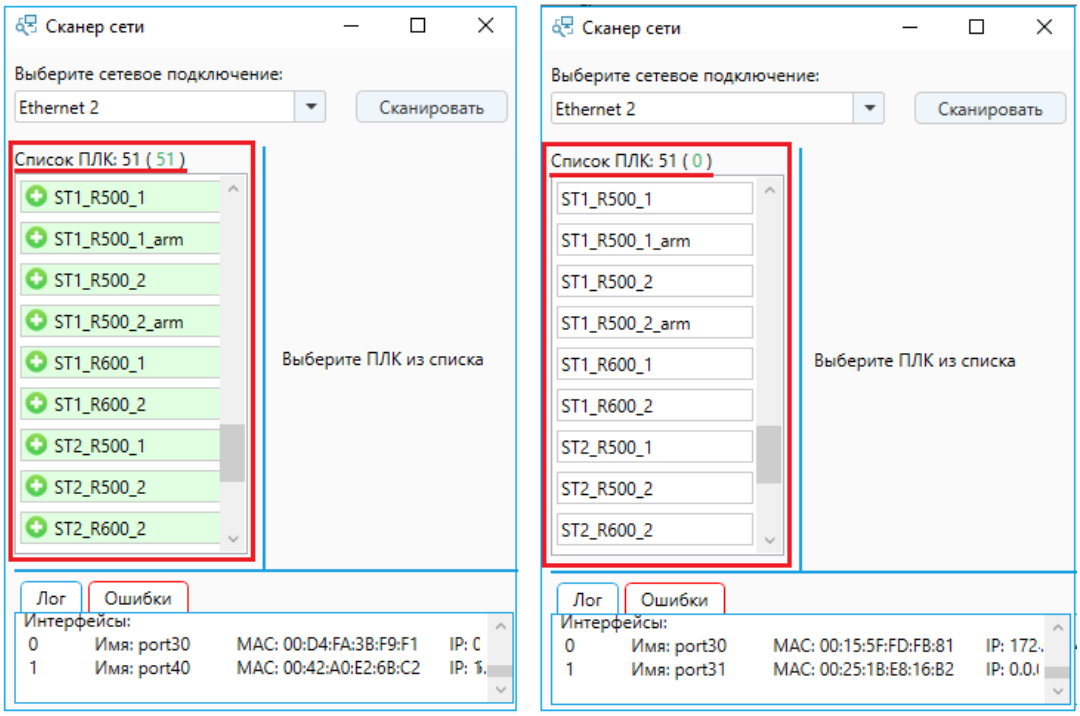


Рисунок 138 – Список контроллеров в сети при первичном сканировании и повторном

В поле **Список ПЛК:** выберите контроллер, для которого требуется установить параметры соединения (Рисунок 139). Чтобы убедиться, что выбран контроллер, который соответствует нужному, нажмите кнопку ***Звуковой сигнал.*** Контроллер издаст кратковременный звуковой сигнал, сопровождающийся одновременным миганием индикаторов, расположенных на передней панели модуля ЦП:

- серия R200: индикаторы RUN и LDx;
- серия R400: функциональный индикатор (на лицевой панели в верхнем левом углу);
- серия R500: индикаторы RUN, RDD, MBx и LDx;
- серия R600: индикаторы RUN, RDD и LEDx.

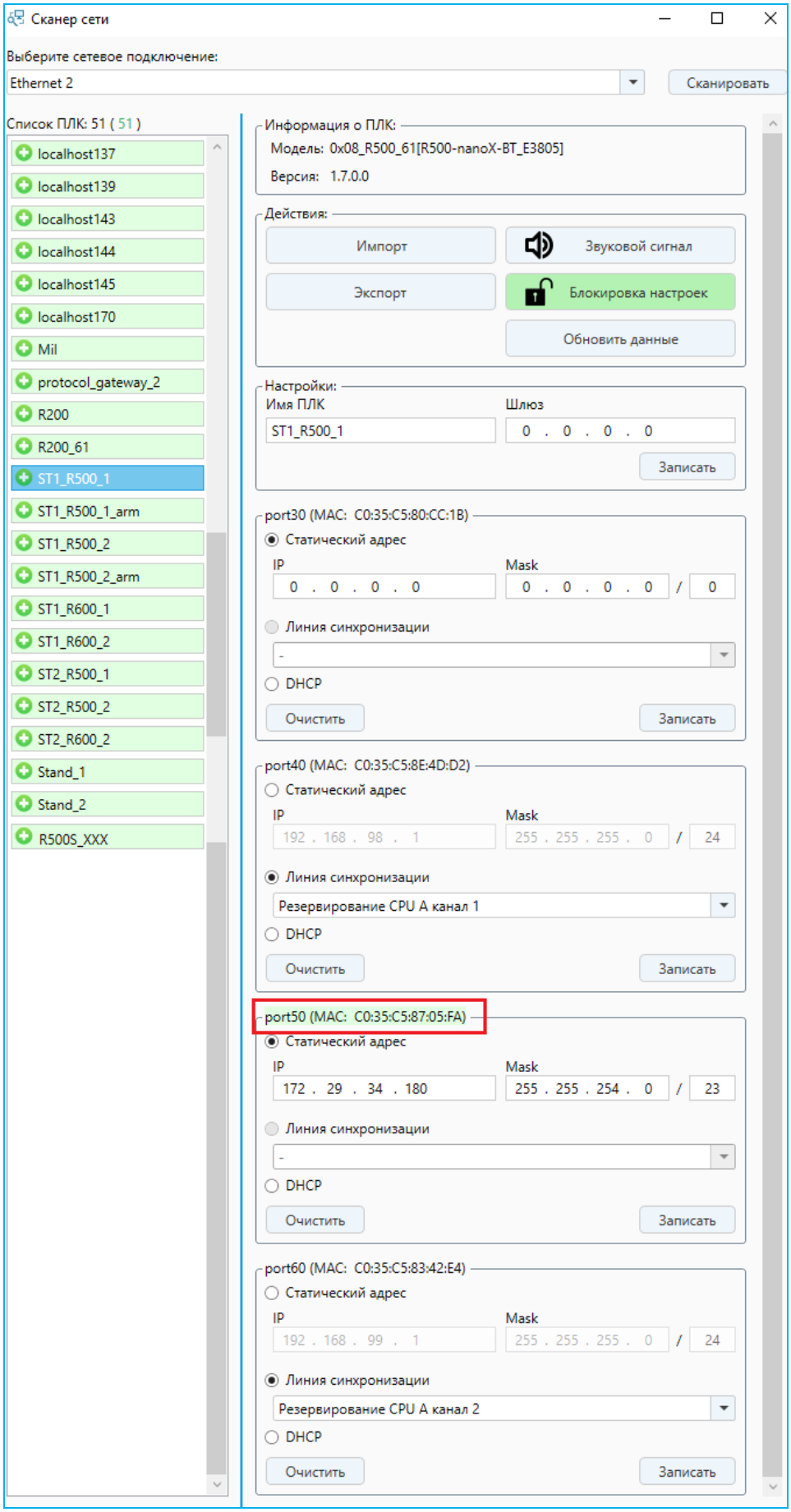


Рисунок 139 – Информация о контроллере

В области **Информация о ПЛК:** будут отображены сведения о модели и версии контроллера.

В области **Действия**: отображены кнопки команд *Импорт*, *Экспорт*, *Звуковой сигнал*, *Блокировка настроек* и *Обновить данные*.

В области **Настройки**: представлено символьное сетевое имя контроллера и адрес Шлюза.

В областях **port** представлены номер порта, его MAC-адрес и сетевые интерфейсы выбранного контроллера (порты: **port30**…**port60** относятся к модулю ЦП, а **port70**…**port130** – к блокам расширения EU). Зеленым цветом выделен активный порт, то есть с которого контроллер отвечает на сетевые запросы.

При наведении курсора мыши на название активного порта появится всплывающая подсказка о скорости сетевого подключения (Рисунок 140).

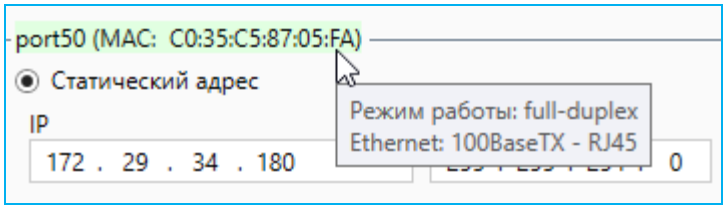


Рисунок 140 – Информация о скорости сетевого подключения

Если требуется изменить символьное сетевое имя контроллера, то в поле **Имя ПЛК** сотрите существующее название, введите новое, нажмите кнопку ***Записать***. Имя может содержать только латинские буквы, цифры и знак подчеркивания «_».

Если есть необходимость назначить шлюз по умолчанию, то укажите его адрес в поле **Шлюз**, нажмите кнопку ***Записать***. Этот адрес шлюза будет действителен для всех сетевых интерфейсов.

Статический адрес позволяет редактировать IP-адрес и маску подсети. Для этого в полях **IP** и **Mask** введите IP-адрес и маску подсети.

	**ВНИМАНИЕ!** Важно, чтобы на портах ПЛК были заданы IP-адреса из разных подсетей

Нажмите кнопку ***Записать***. Для удаления данных в полях **IP** и **Mask** нажмите кнопку ***Очистить***, поля будут очищены, нажмите кнопку ***Записать***.

Есть возможность автоматического конфигурирования IP-адресов на контроллерах, поддерживающих протокол DHCP, чтобы исключить ошибки ручного конфигурирования IP-адресов, приводящих к конфликту в сети.

Для назначения IP-адреса автоматически, установите точку в поле **Port** с названием строки **DHCP** (Рисунок 141).

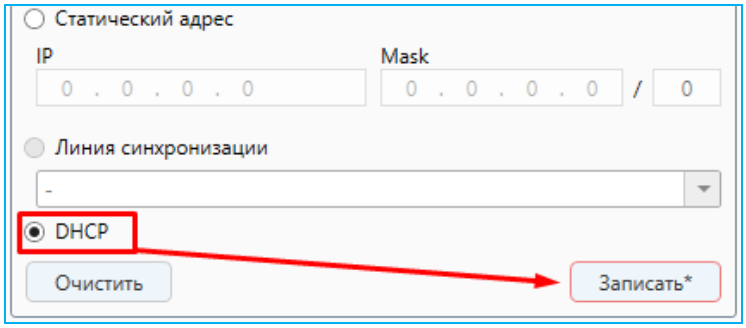


Рисунок 141 – Включение автоматического конфигурирования IP-адресов

Нажмите кнопку ***Записать***, а затем (примерно через 30 сек.) кнопку ***Обновить данные*** (область **Действия:**). Автоматически будут заполнены поля **IP** и **Mask**. Значение полей будет не доступно для редактирования вручную (Рисунок 142).

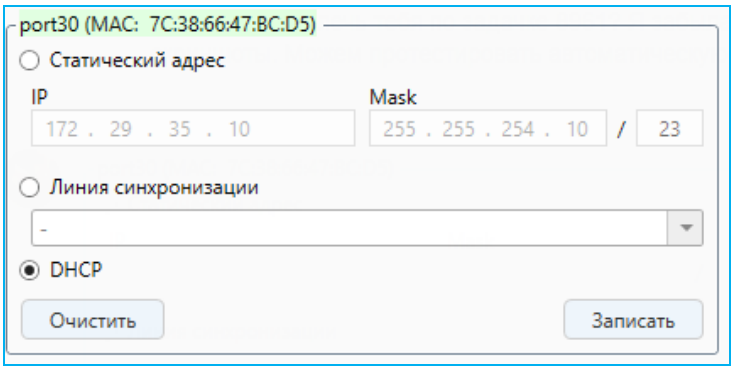


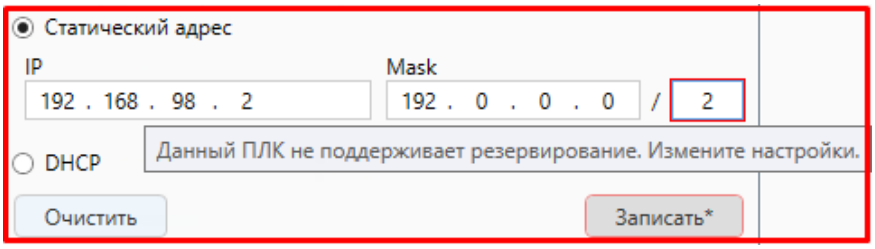
Рисунок 142 – Автоматическое заполнение полей

i	**ИНФОРМАЦИЯ** Для сохранения внесенных изменений на ПЛК необходимо каждый раз нажимать кнопку ***Записать*** (при этом, будет появляться напоминание - «*» и контур кнопки будет окрашен в красный цвет). В резервированных системах, после нажатия кнопки ***Записать***, потребуется перезагрузить ЦП для вступления в силу сетевых настроек

Установка точки «**Линия синхронизации**» резервирует порт для синхронизации данных в резервированном контроллере (подробнее см. документ «Конфигурирование резервированной системы на контроллерах серии REGUL RX00. Руководство пользователя»). При этом следует выбрать за каким из двух каналов синхронизации резервируется порт (Рисунок 143).

ИНФОРМАЦИЯ

Начиная с версии 1.7.1.0 отсутствует параметр «**Линия синхронизации**» у ПЛК, не поддерживающих резервирование. Также, при попытке назначить ПЛК один из IP-адресов, предназначенных для резервированных контроллеров (192.168.98.1/24, 192.168.98.2/24, 192.168.99.1/24, 192.168.99.2/24), появится красная подсветка, а при наведении на нее курсора мыши - всплывающая подсказка.



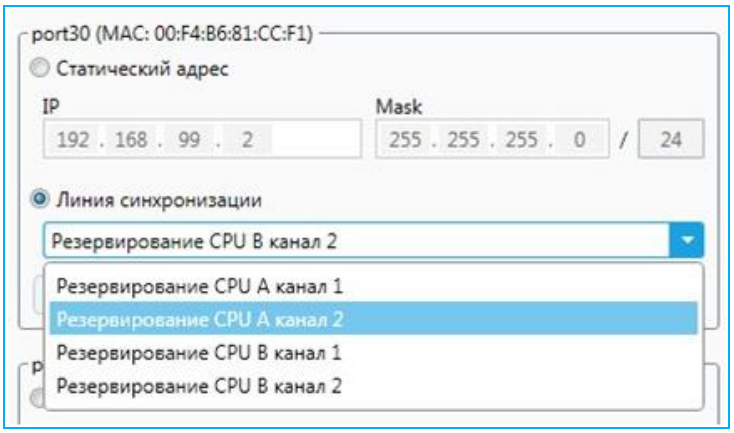


Рисунок 143 – Выбор канала резервирования CPU A/B резервированного контроллера

ВНИМАНИЕ!

При настройке линии синхронизации на активный порт (отмечен зеленым цветом) будет потеряна связь с контроллером (Сканер сети не будет видеть данный порт)

При переключении настроек с **Линия синхронизации** на **DHCP**, после нажатия кнопки ***Записать***, в окне сообщений появится запись следующего вида: «!Для вступления в силу изменений потребуется перезагрузка» (Рисунок 144). Перезагрузите контроллер.

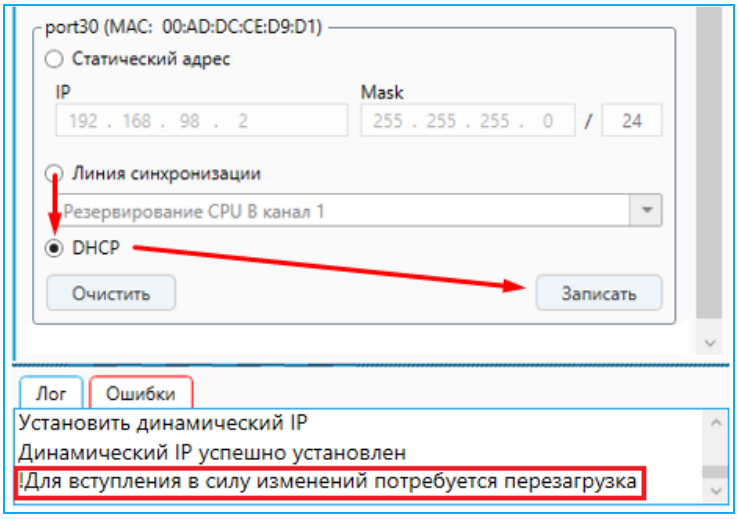


Рисунок 144 – Изменение настроек

В сканере сети предусмотрены команды **Экспорт** и **Импорт**, избавляющие от необходимости вручную вносить все IP-адреса.

Для сохранения IP-адресов в файл нажмите кнопку ***Экспорт***, откроется окно **Экспорт сетевых интерфейсов (Save net interfaces settings)**. Определите папку, в которой будет храниться этот файл, задайте ему имя, нажмите кнопку Сохранить.

Для восстановления IP-адресов после обновления системного ПО (или другой процедуры, повлекшей аннулирование настроек контроллера) обратитесь к сканеру сети, найдите нужный контроллер. Далее нажмите кнопку ***Импорт***, выберите файл с настройками, нажмите кнопку ***Открыть***. Появится диалоговое окно, где будет указан контроллер и перечень настроек (IP-адресов), применяемых к нему. Если эти настройки подходят, нажмите кнопку ***Да***. Окно закроется, а значения IP-адресов, масок подсети, шлюз будут автоматически внесены в соответствующие поля. Если настройки не подходят, нажмите кнопку ***Нет***. Окно закроется, выберите другой файл.

С помощью кнопки ***Блокировка настроек*** можно установить запрет на изменение сетевых настроек, если данный функционал поддерживает ПЛК (Рисунок 145).

Рисунок 145 – Кнопка Блокировка настроек в окне сканера сети

При нажатии кнопки всплывет окно подтверждения действия (Рисунок 146).

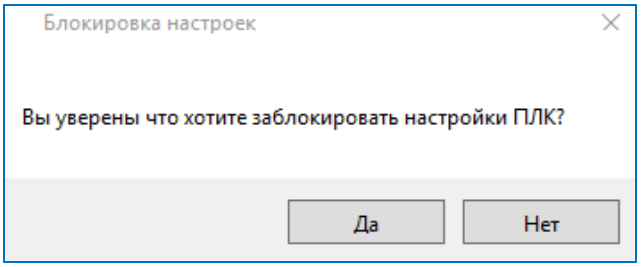


Рисунок 146 – Окно подтверждения блокировки настроек

Нажмите кнопку «**Да**» и все элементы управления станут тусклыми и не доступными для изменения, кроме кнопки ***Звуковой сигнал***.

Кнопка ***Звуковой сигнал*** предназначена для получения звукового отклика от соответствующего ПЛК, сопровождающегося одновременным миганием соответствующих индикаторов.

Если необходимо снять блокировку настроек, выполните следующее:

– установите соединение с контроллером (смотри раздел «Установка соединения с контроллером»);

– пройдите на вкладку **Сервис ПЛК ⇨ Системные параметры** и нажмите кнопку (***Обновить***). Выберите вкладку: **Простой режим** или **Экспертный режим**;

– на вкладке **Простой режим** в поле *global* с названием строки *Запрет на изменение сетевых настроек* снимите флажок и нажмите кнопку **Сохранить** (Рисунок 147);

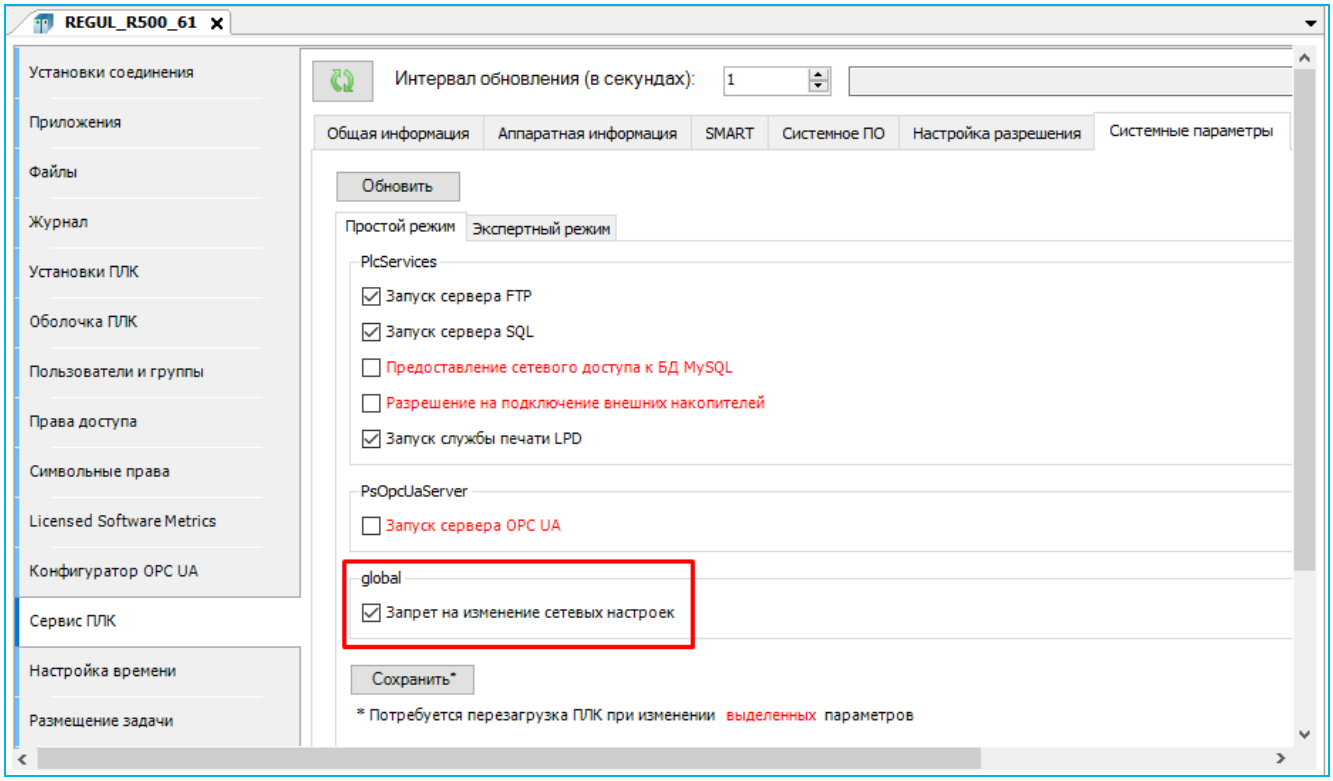


Рисунок 147 – Снятие запрета на изменение сетевых настроек, вкладка простого режима

– либо на вкладке **Экспертный режим** выберите название каталога конфигурационного файла **etc/network.cfg** (см. «Приложение В»). Удалите строку, выделенную красным цветом и нажмите кнопку **Сохранить** (Рисунок 148).

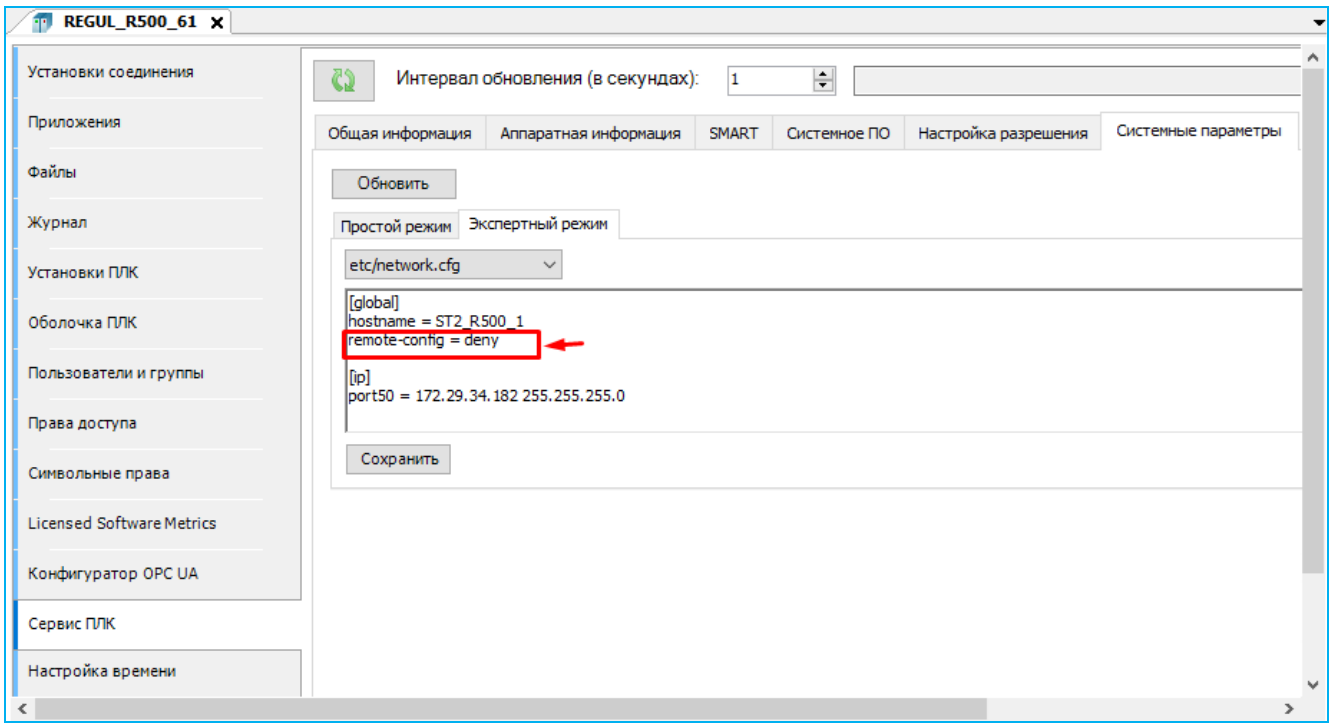


Рисунок 148 – Снятие запрета на изменение сетевых настроек, вкладка экспертного режима

При внесении изменений в одном из двух режимов, для актуализации изменений в другом режиме, нажмите кнопку **Обновить**.

Для продолжения работы по подключению контроллера к сети закройте окно **Сканер сети**, нажав кнопку ☒ в правом верхнем углу.

Установка MAC-адресов на сетевые интерфейсы модулей ЦП

Если требуется установить MAC-адрес на сетевой интерфейс модуля ЦП, выполните следующие действия (Рисунок 149):

– установите соединение с контроллером (смотри раздел «Установка соединения с контроллером»);

– пройдите на вкладку **Сервис ПЛК ⇨Системные параметры** и нажмите кнопку (***Обновить***). Выберите вкладку **Экспертный режим**;

– выберите название каталога конфигурационного файла **etc/network.cfg** (см. «Приложение В»). Добавьте секцию [mac] с необходимым значением;

– нажмите кнопку **Сохранить**.

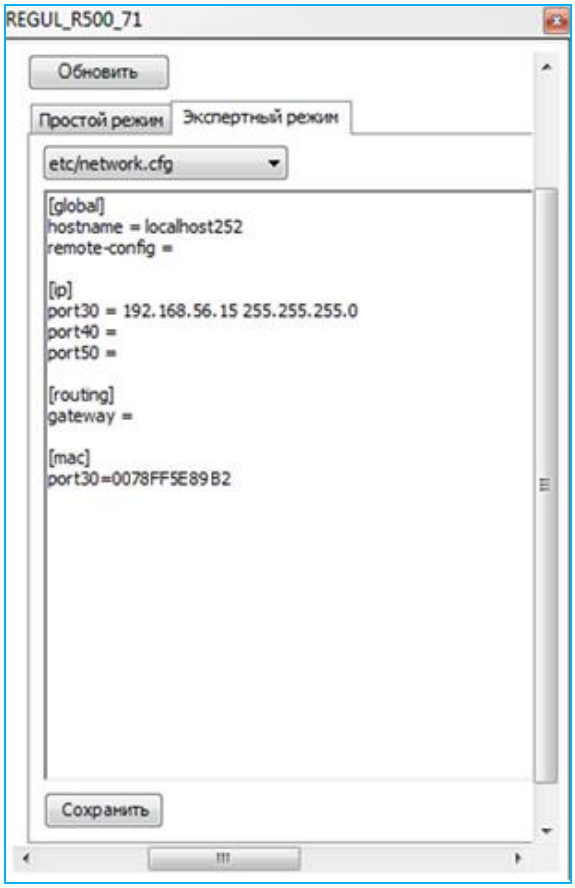


Рисунок 149 – Добавление MAC-адреса в конфигурационном файле network.cfg.

i	**ИНФОРМАЦИЯ** В случае, если для порта не задан или задан некорректный адрес, адрес устанавливается из файла random.mac

Установка дополнительных IP-адресов

Установить дополнительные IP-адреса и маску сетевого интерфейса можно путем назначения псевдонима (alias). Функция IP alias позволяет использовать псевдонимы IP на любом активном интерфейсе для разделения сети на различные подсети.

Для установки дополнительных IP-адресов, выполните следующие действия (Рисунок 150):

– установите соединение с контроллером (смотри раздел «Установка соединения с контроллером»);

– пройдите на вкладку **Сервис ПЛК ⇨Системные параметры** и нажмите кнопку (***Обновить***). Выберите вкладку **Экспертный режим**;

– выберите название каталога конфигурационного файла **etc/network.cfg** (см. «Приложение В»). Добавьте в секцию [ip] необходимое количество строк, согласно описанию:

```
[ip]
portXX.N=<ip-адрес> <маска>
```

– где N – номер дополнительно IP-адреса, нумерация начинается с «0».

– нажмите кнопку **Сохранить**.

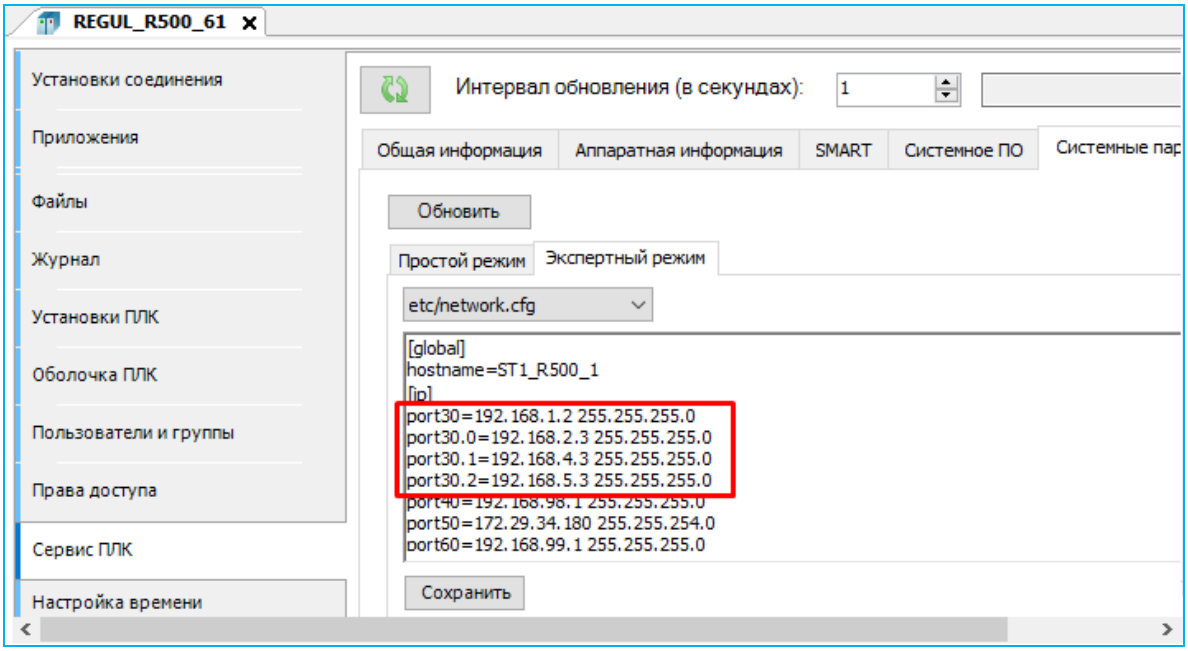


Рисунок 150 – Установка дополнительных IP-адресов

Установка соединения с контроллером

В среде разработки Astra.IDE используется понятие **шлюз (gateway)**. Только через него среда разработки Astra.IDE может установить соединение с контроллером. По сути, шлюз (gateway) – это устройство для сопряжения компьютерных сетей, использующих разные протоколы. Таким шлюзом может быть локальный ПК разработчика, может быть другой ПК в сети, имеющий прямое соединение с контроллером. На таком gateway-устройстве должна быть запущена коммуникационная служба Gateway Server.

Откройте проект. В окне **Устройства** в дереве устройств выберите настраиваемый контроллер, дважды щелкните по нему. В правой части окна появится главная вкладка параметров устройства, где по умолчанию открыта внутренняя вкладка **Установки соединения** (Рисунок 151).

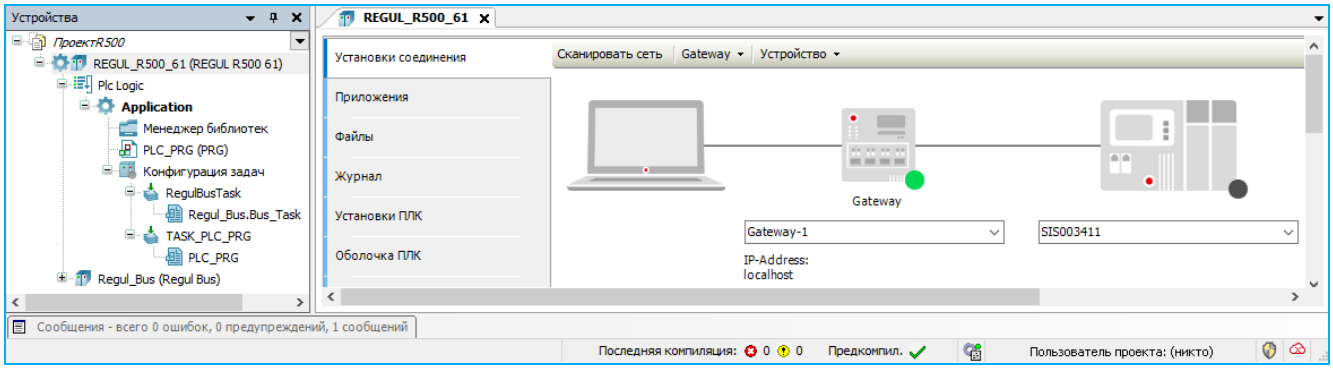


Рисунок 151 – Вкладка установок соединения

Выберите шлюз (gateway) из раскрывающегося списка. Локальный шлюз (gateway) доступен сразу, он автоматически запускается при старте сервиса Gateway.

При нормальной (штатной) работе шлюза (gateway) маркер рядом с его схематическим изображением окрашен в зеленый цвет, в противном случае – в красный. Черный маркер означает, что соединение еще не определено.

При отсутствии в списке нужного шлюза (gateway) его можно создать. Для этого выберите **Gateway** ▾ ⇨ **Add new gateway…(Добавить новый gateway…)** Откроется окно **Gateway** (Рисунок 152).

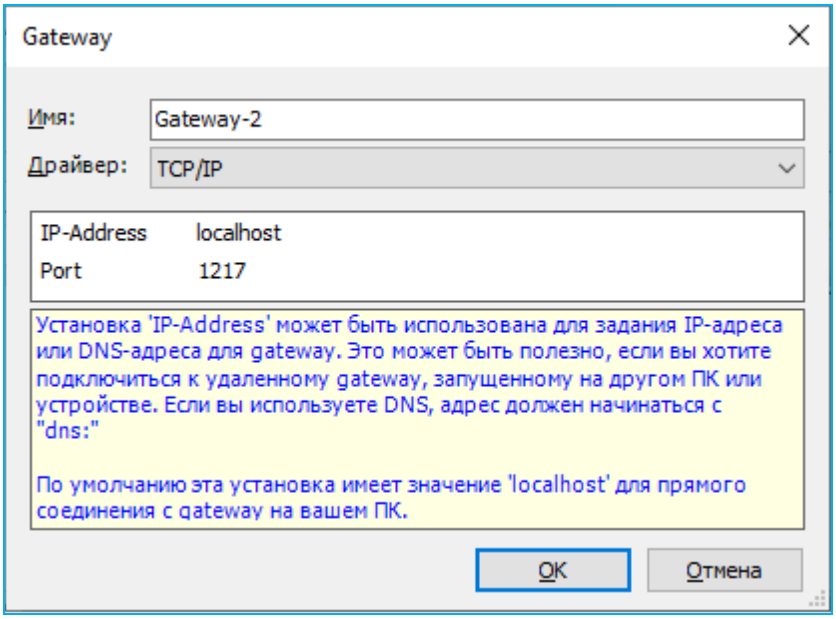


Рисунок 152 – Добавление нового gateway

Нужно задать имя шлюза (gateway) и его IP-адрес (двойной щелчок в поле **IP-Address** позволяет отредактировать значение в этом поле). Для локальной сети – это localhost или адрес локальной сети, для удаленной сети - соответствующий адрес.

Установку **Port** в локальной сети следует оставить без изменений (подлежит изменению только в случае, когда стандартный порт заблокирован). Для удаленной сети используются порты:

– UDP 1740-1743 для broadcast запросов;

– TCP 1217 для gateway;

– TCP 11740-11743 для соединения с ПЛК.

Для добавления или удаления шлюза (gateway) используйте **Gateway** ▾ ⇨ **Manage gateway…(Управление gateway…)** Откроется окно, где существующие шлюзы (gateway) можно расставить в нужном порядке с помощью кнопок ***Переместить вниз*** и ***Переместить вверх***. При этом объект, расположенный самым первым, будет являться шлюзом (gateway) по умолчанию для новых проектов и устройств. Для добавления gateway

воспользуйтесь кнопкой ***Добавить***, откроется окно **Gateway** (описано выше, Рисунок 152). Кнопка ***Удалить*** позволяет удалить шлюз (gateway).

	ВНИМАНИЕ! Программа не запрашивает подтверждения на удаление

После того, как определен шлюз (gateway), через сканирование сети необходимо определить канал к целевому устройству (контроллеру), который следует подключить через заданный шлюз (gateway).

Сканирование сети

Нажмите кнопку ***Scan network…(Сканировать сеть)*** Откроется окно **Выбор устройства** (Рисунок 153).

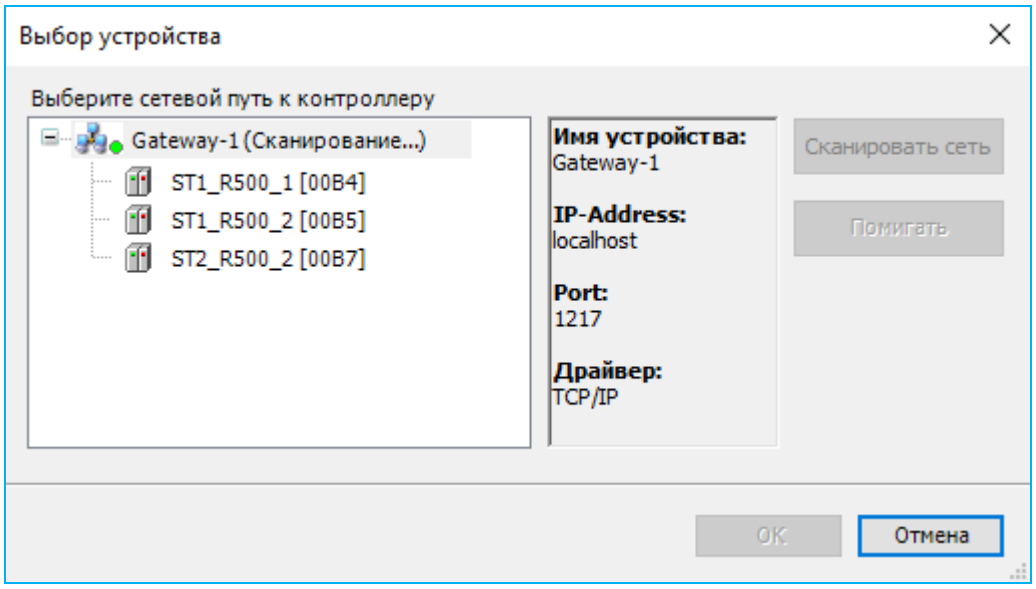


Рисунок 153 – Сканирование локальной сети

Подождите несколько секунд, пока кнопка ***Сканировать сеть*** станет активной. Нажмите ее чтобы получить актуальный список подключенных устройств. Выполнить команду можно также двойным щелчком мыши по элементу gateway.

Пока шлюз (gateway) работает нормально, он помечен зеленым маркером, в противном случае – красным. Серый маркер указывает на то, что шлюз (gateway) еще не подключен (некоторые коммуникационные протоколы не позволяют опрашивать состояние шлюза, поэтому его статус не может быть отображен).

В результате сканирования будут показаны все устройства, доступные в сети (локальной или удаленной). Если нужного устройства нет в списке, то закройте окно **Выбор устройства** и выполните одно или несколько следующих действий:

– проверьте физическое подключение контроллера к сети;

– проверьте, правильно ли указан IP-адрес контроллера;

– на вкладке **Установки соединения** выберите **Device** ▾ **(Устройство)** ⇨ **Опции** ▸ и снимите флажок в пункте **Filter network scans by target ID (Скан сети по ID Таргета)** (или убедитесь, что он не установлен).

Еще раз просканируйте сеть. Если устройство в списке обозначено бледно-серым значком, при этом недоступно для выбора, это означает, что оно не соответствует типу выбранного контроллера. Выберите нужный контроллер, в правой части окна появится информация о нем (Рисунок 154).

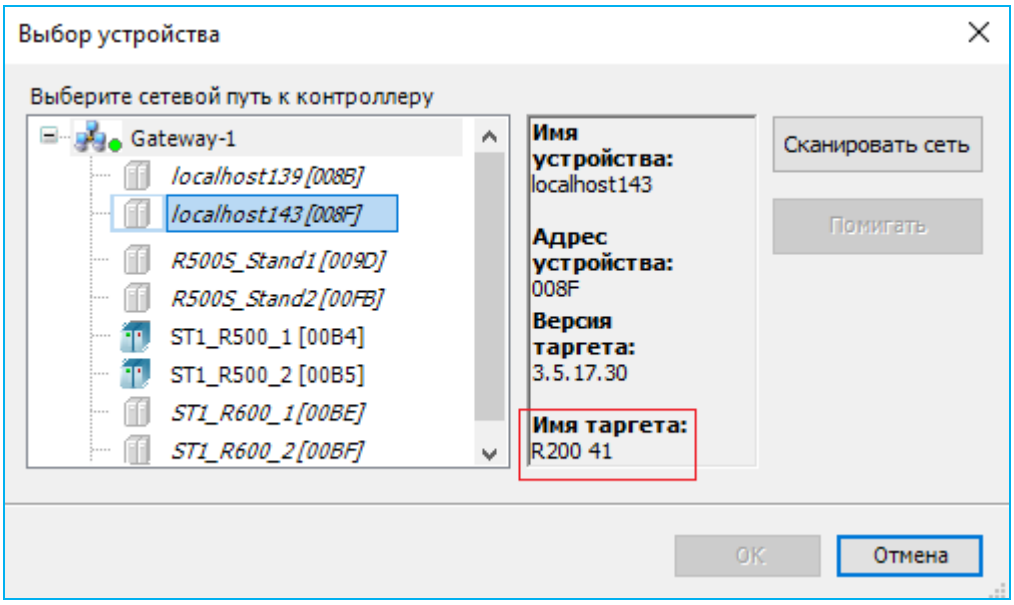


Рисунок 154 – Контроллер недоступен для выбора

В окне **Устройства** в дереве устройств обновите тип контроллера (в контекстном меню пункт **Обновить устройство…**), выбрав для него ту же модель, что указана в поле **Имя таргета:** (Рисунок 154).

Повторите сканирование сети. Просмотрите список всех устройств в сети и выберите нужный контроллер. Нажмите кнопку ***ОК***.

Авторизация при подключении к ПЛК

До версии 1.7.0.0

После выбора нужного контроллера, будет открываться окно **Вход в систему** (Рисунок 155). Для установки соединения с ПЛК необходимо будет ввести имя пользователя (учетную запись) – *Administrator*, с предварительно заданным по умолчанию паролем *Administrator* (заводская настройка). Данная учетная запись обладает максимальными правами.

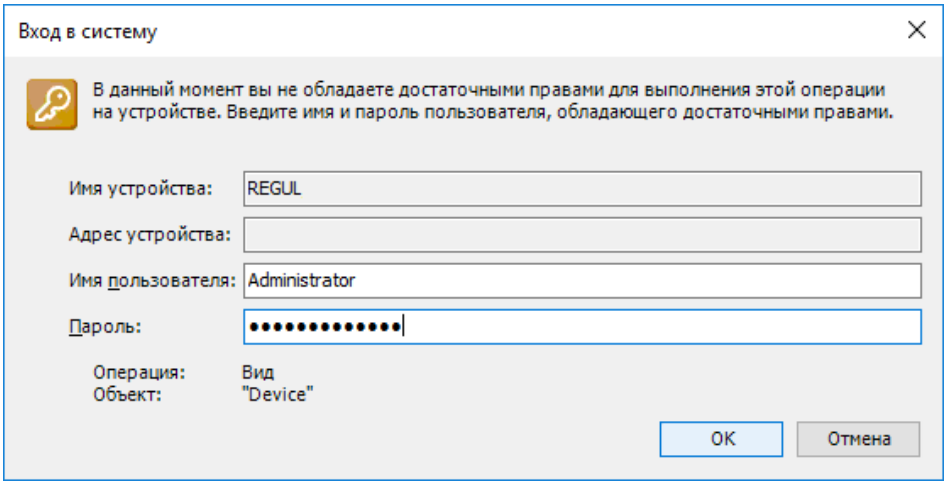


Рисунок 155 – Окно входа в систему

После успешной авторизации откроется окно с требованием сменить пароль учетной записи *Administrator*. В дальнейшем, при повторных подключениях к ПЛК, будет использоваться новый пароль (Рисунок 156).

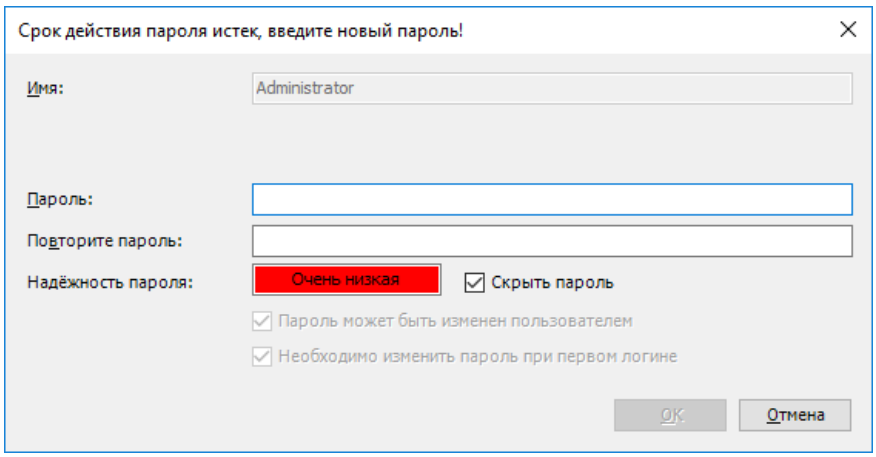


Рисунок 156 – Окно ввода нового пароля

После подтверждения окно закроется, произойдет переход обратно в главное окно программы, где на вкладке параметров устройства возле схематического изображения устройства должен стоять зеленый маркер (устройство работает нормально), показан адрес устройства и его параметры (Рисунок 157).

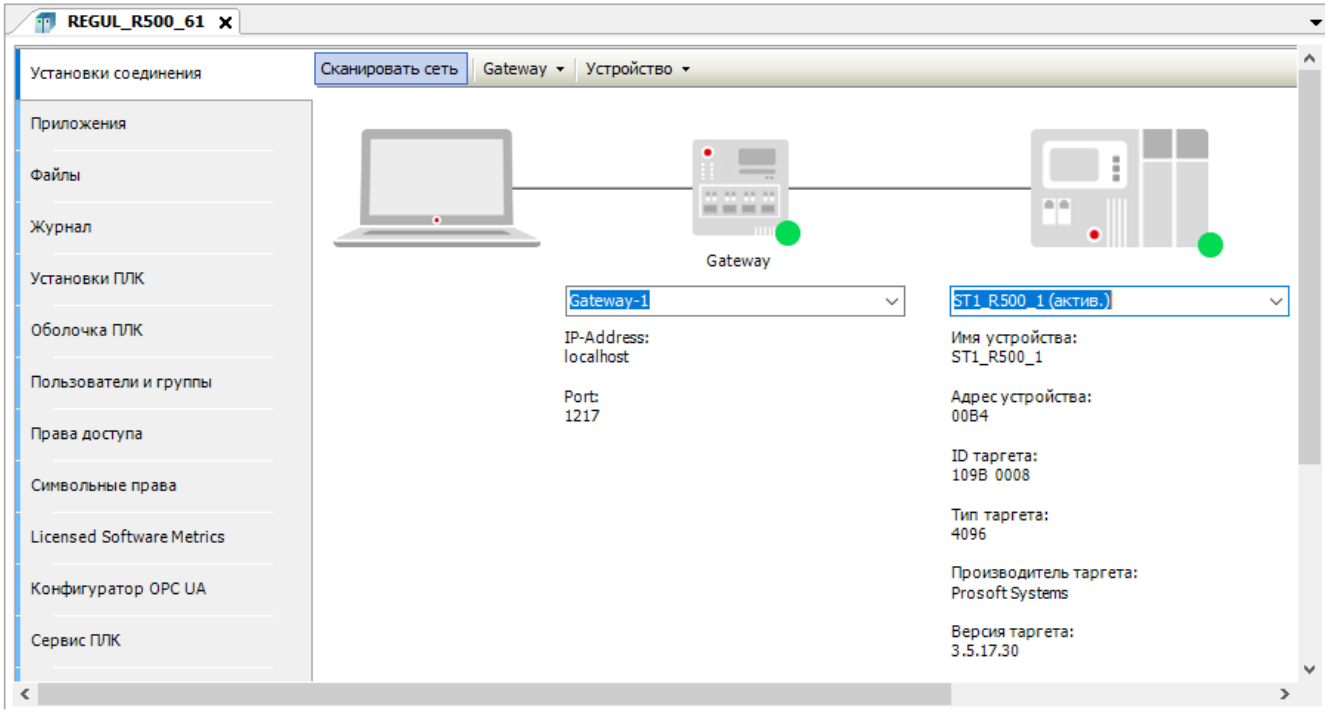


Рисунок 157 – Контроллер найден в локальной сети и выбран в качестве активного

Сбросить пароль учетной записи до заводского можно двумя способами: через ПО Astra.IDE или сервисный режим контроллера (см. подраздел «Сервисный режим контроллера» пункт «Алгоритм сброса к заводским настройкам»).

Сервисный режим используют, когда нет возможности подключения к контроллеру через Astra.IDE.

Сброс через ПО Astra.IDE производится следующим образом:

– в окне дерева устройств поместите курсор на название контроллера **REGUL…**, нажмите правую кнопку мыши (Рисунок 158);

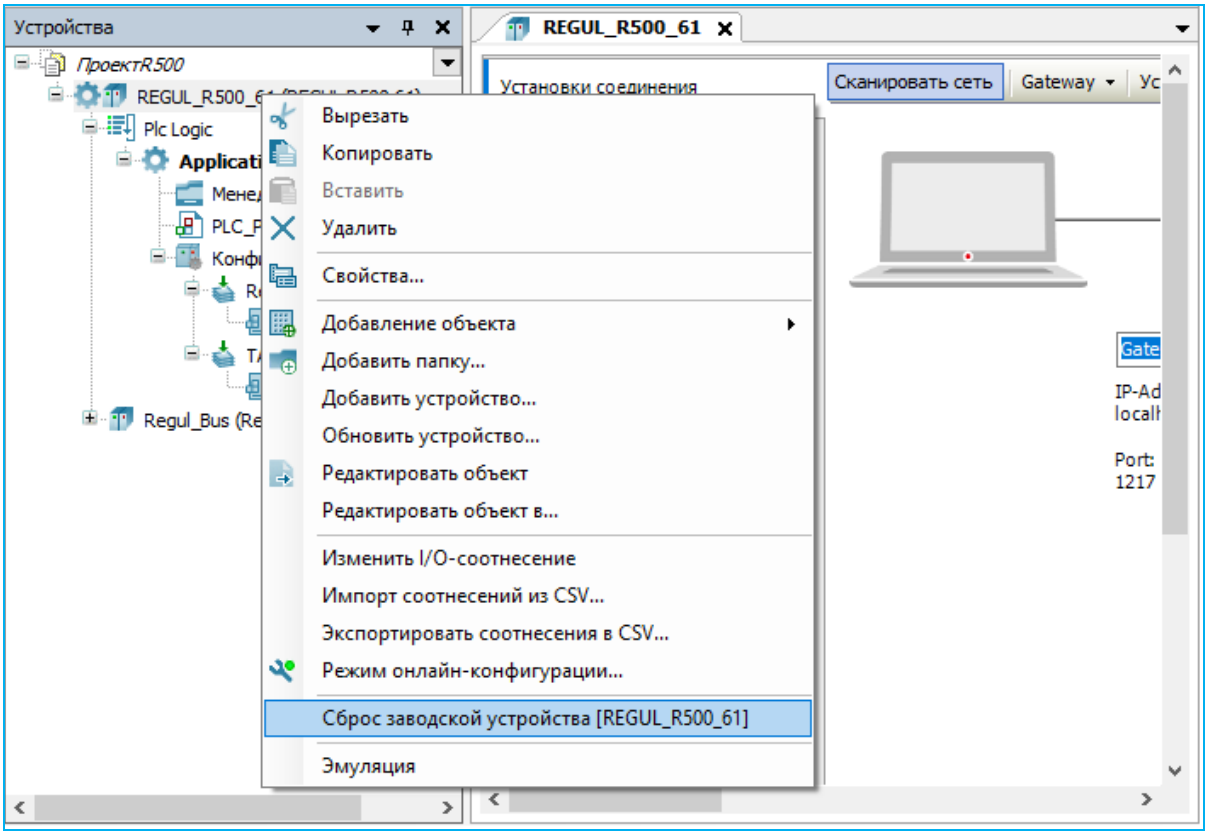


Рисунок 158 – Контекстное меню устройства

– в появившемся контекстном меню выберите пункт **Сброс заводской устройства [REGUL…].** Откроется окно подтверждения операции (Рисунок 159);

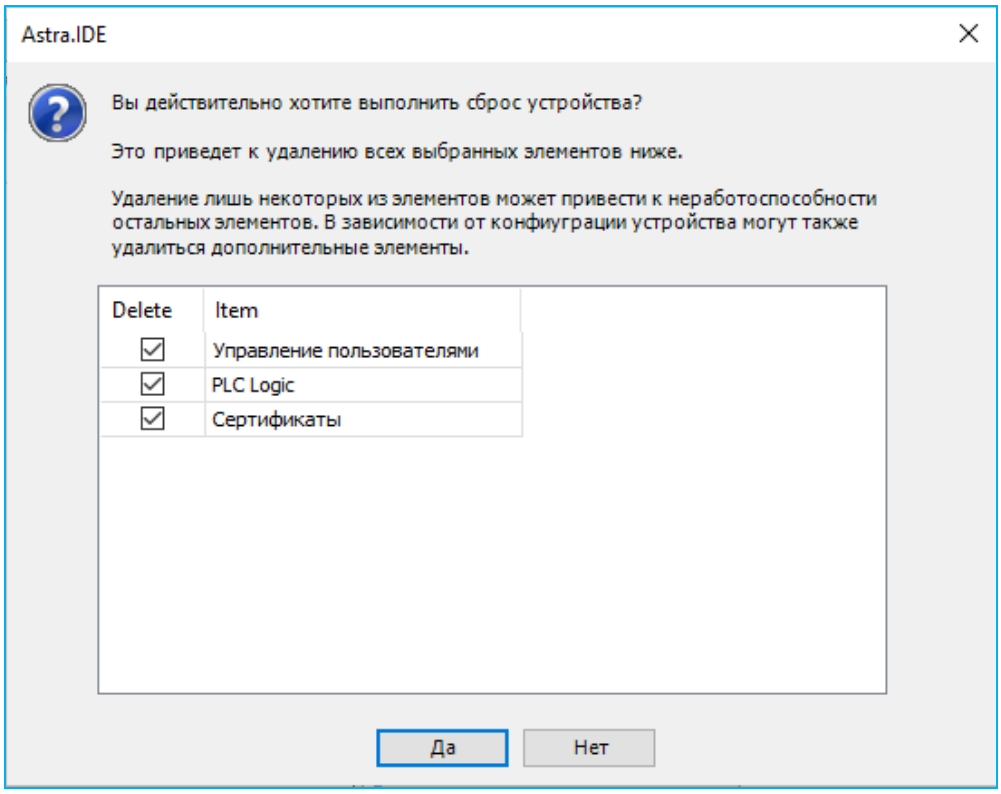


Рисунок 159 – Окно подтверждения операции

– определите необходимый объем удаления дополнительных элементов, установив/сняв флажок в поле напротив соответствующего элемента;

– нажмите кнопку **ДА**. Теперь при повторном подключении к ПЛК откроется окно **Вход в систему** с заводскими настройками, описанными выше. Контроллер сбросится до заводских настроек. Все приложения и переменные будут удалены с контроллера.

Начиная с версии 1.7.0.0

После выбора нужного контроллера, будет открываться информационное окно **Astra.IDE** (Рисунок 160). Для первоначального подключения необходимо активировать управление пользователями и задать параметры. Ознакомтесь с информацией и подтвердите активацию управления пользователями, нажав на кнопку **Yes**.

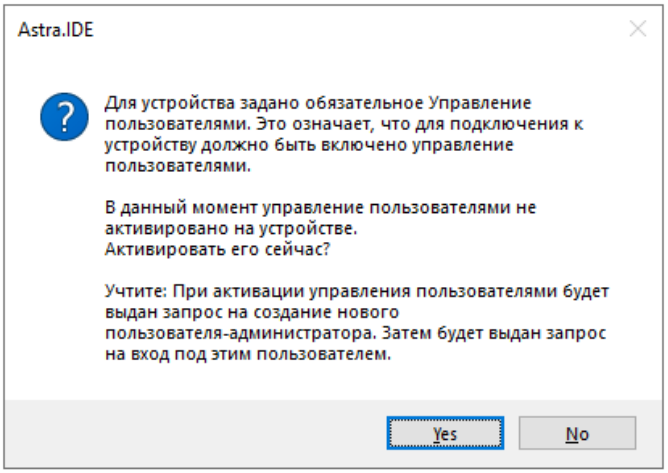


Рисунок 160 – Информационное окно по активации управления пользователями

Появится окно **Добавить пользователя устройства** (Рисунок 161).

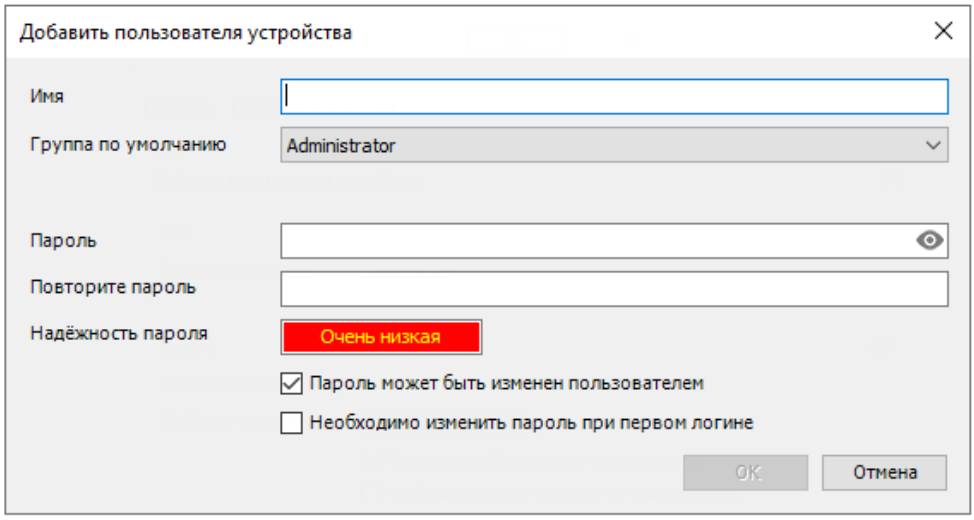


Рисунок 161 – Добавление пользователя устройства

Введите имя и задайте пароль для учетной записи (Рисунок 162), которая будет обладать максимальными правами (администратор). Нажмите на кнопку **ОК.**

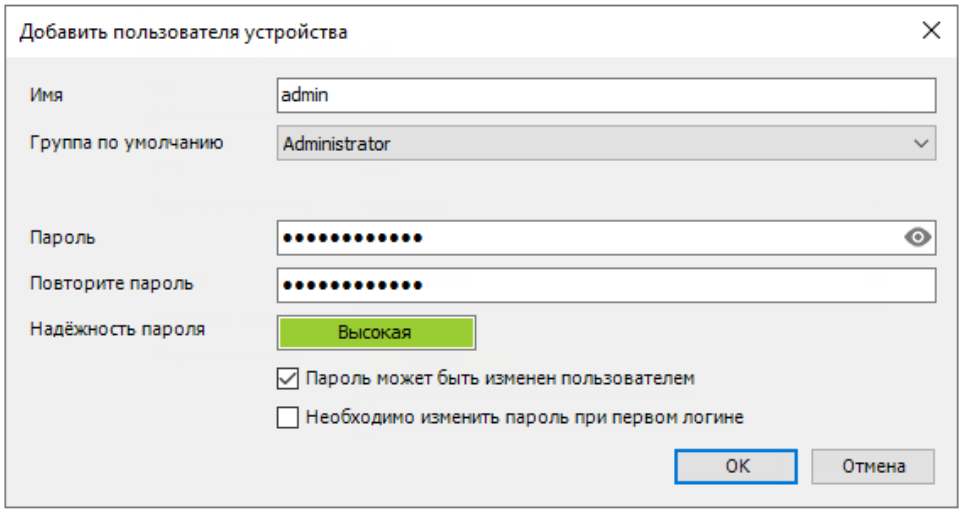


Рисунок 162 – Ввод данных учетной записи

Откроется окно **Вход в систему** (Рисунок 163**)** Для установки соединения с ПЛК введите ранее заданное имя пользователя (учетную запись) и пароль.

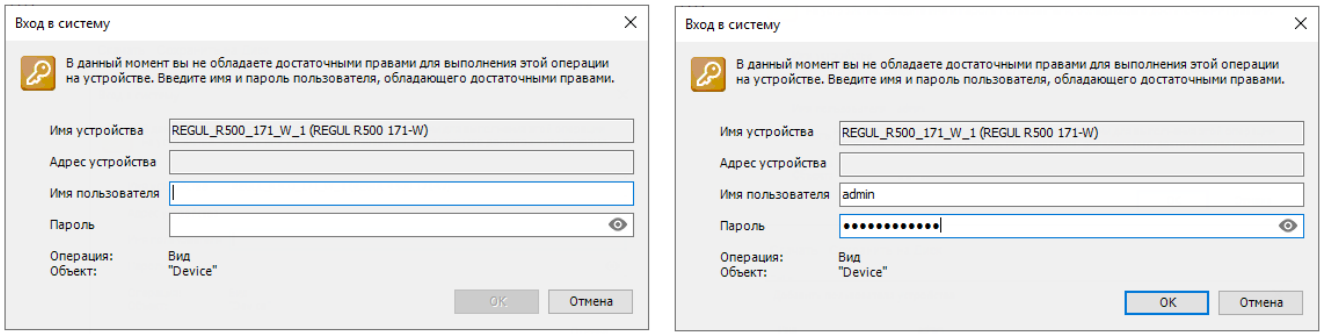


Рисунок 163 – Окно входа в систему

Если ранее был установлен флажок в поле **Необходимо изменить пароль при первом логине**, то, после успешной авторизации, откроется окно с требованием сменить пароль учетной записи (Рисунок 164). В дальнейшем, при повторных подключениях к ПЛК, будет использоваться новый пароль.

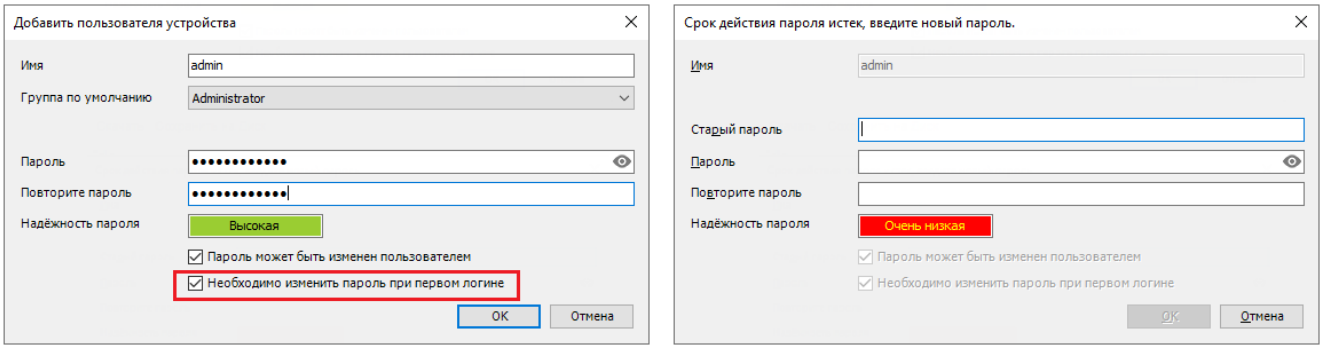


Рисунок 164 – Смена пароля при первом логине. Окно ввода нового пароля

В дальнейшем изменить пароль учетной записи можно двумя способами: через ПО Astra.IDE или сбросив контроллер до заводского состояния из сервисного режима контроллера (см. описание выше).

Конфигурирование авторизации

Для включения/выключения процедуры авторизации необходимо выполнить следующие действия:

– установите соединение с ПЛК (см. описание выше);

– дважды щелкните левой кнопкой мыши по названию контроллера в окне дерева устройств и в правой части окна появится вкладка **Установки соединения**.

– выберите **Устройство ▾ (Device) ⇨ Change communication policy…(Изменить политику соединения…).** Откроется окно следующего вида (Рисунок 165):

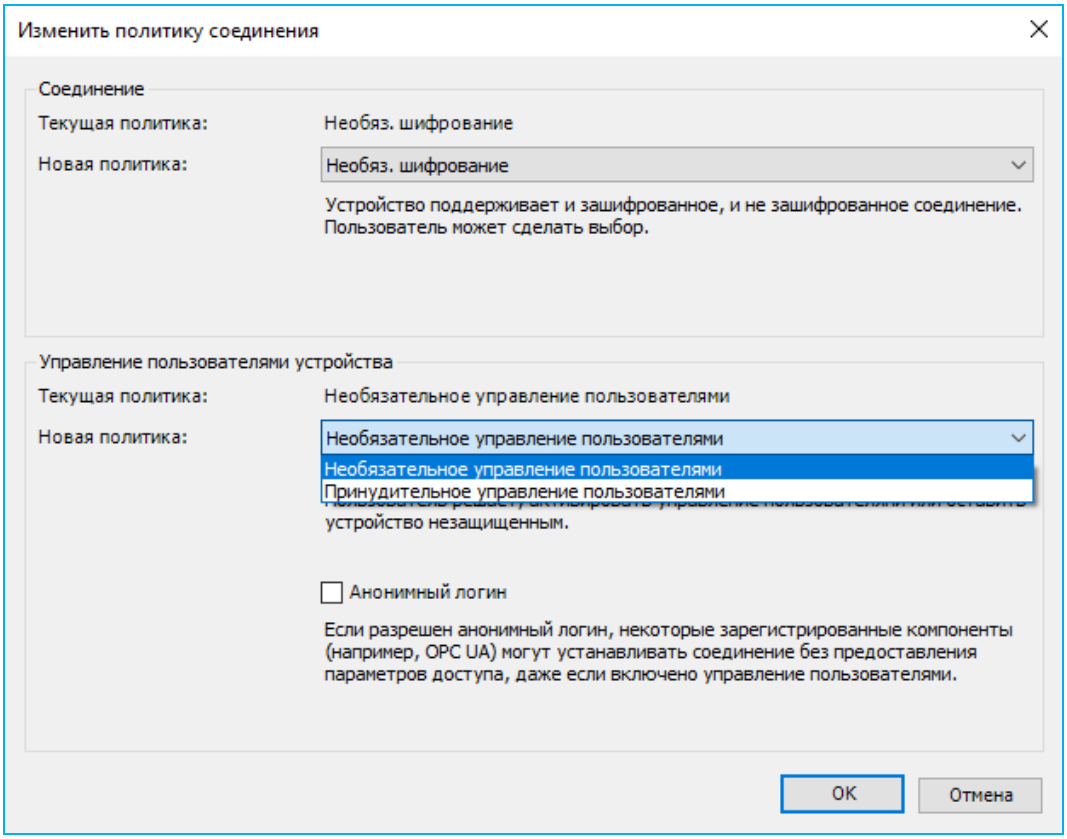


Рисунок 165 – Окно изменения политики соединения

– в области **Управление пользователями устройства (Device User Management)** в поле **Текущая политика (Current policy):** отображается текущее состояние политики управления пользователями (по умолчанию **Принудительное управление пользователями (Enforced user management)).** Для изменения политики выберите в выпадающем списке в поле **Новая политика (New policy)** необходимый вариант:

- **Необязательное управление пользователями (Optional user management)** – авторизация выключена (устройство не защищено).
- **Принудительное управление пользователями (Enforced user management)** – авторизация включена.

Установив флажок в поле **Анонимный логин**, пользователь разрешает определенным зарегистрированным компонентам (например, OPC UA) подключаться к контроллеру без предоставления каких-либо учетных данных. Даже если анонимный доступ к OPC UA разрешен, созданное управление пользователями устройства для контроллера остается активным.

Для выключения процедуры авторизации выберите пункт **Необязательное управление пользователями (Optional user management)** и нажмите кнопку **ОК**. Далее произведите сброс до заводского состояния (поместите курсор на название контроллера **REGUL…**⇨ **Сброс заводской устройства [REGUL…]**⇨ нажмите кнопку **ДА**). Теперь при повторном подключении к ПЛК окно **Вход в систему** будет отсутствовать.

Для включения обратно процедуры авторизации выберите пункт **Принудительное управление пользователями (Enforced user management)** и нажмите кнопку **ОК**. Теперь при повторном подключении к ПЛК откроется окно **Вход в систему** с заводскими настройками (имя пользователя – Administrator, пароль – Administrator).

Поиск устройства по IP-адресу

В программе предусмотрена возможность подключения к контроллеру непосредственно по IP-адресу, то есть нет необходимости сканировать сеть, выбирать устройство из списка найденных устройств.

Первый вариант: в поле, расположенном под изображением контроллера, введите «имя устройства» либо адрес устройства, либо его IP-адрес. Нажмите клавишу ***Enter***. Если адрес/имя указан верно, контроллер будет найден, будут отображены его адрес и параметры.

Второй вариант: настроить список «любимых устройств» - список устройств, настройки подключения к которым будут сохранены, и подключаться к ним можно без сканирования сети. Выберите **Device** ▾ **(Устройство)** ⇨ **Опции** ▸ ⇨ **Управлять избранными устройствами…** Откроется окно следующего вида (Рисунок 166):

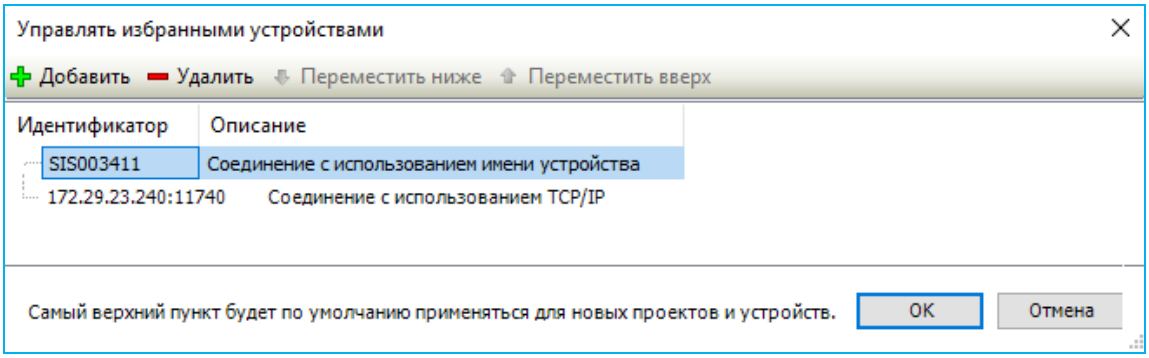


Рисунок 166 – Окно «Управлять любимыми устройствами»

Нажмите кнопку ***Добавить***. Откроется диалоговое окно, где нужно ввести имя устройства, адрес устройства или IP-адрес. Пример адреса устройства: 0104.02F4. Пример

IP-адреса: 192.168.101.15. Нажмите кнопку ***ОК***. Программа автоматически выполнит попытку определить механизм поиска или добавления устройства в список.

Фильтрация по типу устройства (Target ID)

Предусмотрена возможность искать устройства только определенного типа. Для этого выберите **Device (Устройство) ⇨ Опции ⇨ Filter network scans by target ID (Скан сети по ID Таргета),** т.е. фильтровать по идентификатору устройства (Рисунок 167).

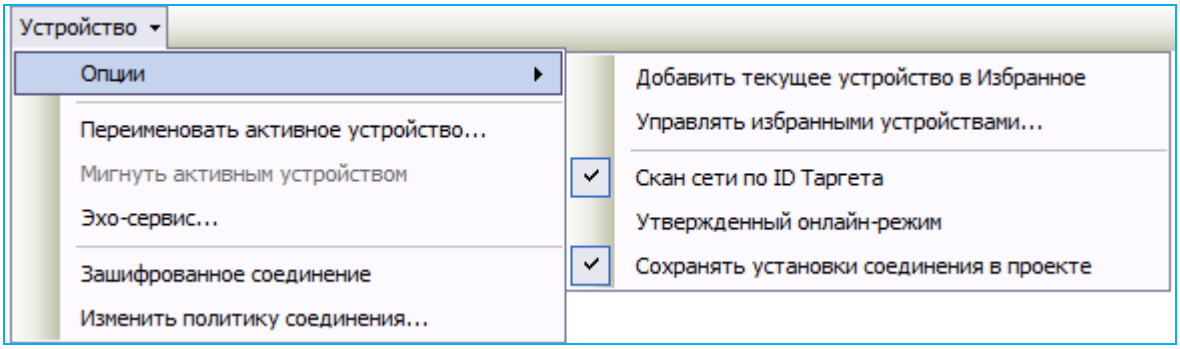


Рисунок 167 – Опция фильтрации активирована

Если установлена галочка, то при сканировании сети будут показываться только найденные устройства, идентификаторы которых совпадают с идентификатором текущего редактируемого устройства.

Защита от широковещательного шторма

Для предотвращения ухудшения или полного прекращения работы сетевой подсистемы ПЛК предусмотрена защита от широковещательного шторма. Защита от штормов осуществляется за счет установленного порогового значения на количество принятых широковещательных пакетов.

Пороговые значения для модулей ЦП следующие:

– для I/III типа – при получении более 50 широковещательных пакетов за 5 мс, на 10 с (таймаут) отключается получение и обработка широковещательных пакетов. По истечении таймаута получение и обработка широковещательных пакетов возвращается в нормальный режим работы;

– для II типа – установлен лимит в 5 широковещательных пакетов за 5 мс, при его превышении все оставшиеся широковещательные пакеты будут отброшены.

Журналирование функции защиты от шторма производится в файл: sloginfo.log, с формированием следующих записей, например:

«…port60 Broadcast storm detected: Ignore broadcast»

«…port60 Broadcast storm timeout: Accept broadcast packets»

Изменение политики соединения с ПЛК

Предусмотрена возможность зашифрованного соединения с контроллером. Для установки зашифрованного соединения будет использоваться сформированный на контроллере сертификат. Установленный сертификат контроллера действителен всего 30 дней и при попытке подключения к ПЛК, каждый раз будет открываться информационное окно об использовании сертификата, срок действия которого скоро истечет (Рисунок 168).

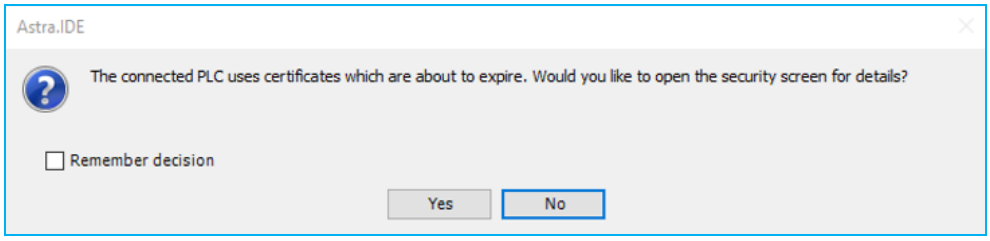


Рисунок 168 – Информационное окно об истечении срока действия сертификата

Необходимо создать сертификат с более длительным сроком действия (365 дней, либо значение определяет пользователь) одним из следующих способов:

– нажмите на кнопку ***NO*** и создайте самоподписанный сертификат через вкладку **Оболочка ПЛК** (срок действия сертификата 365 дней). В командной строке введите команду *cert-getapplist*. Появится список всех применяемых сертификатов. Для компонента **CmpSecureChannel** введите в командной строке команду **cert-genselfsigned *** (* - укажите порядковый номер компонента в списке). Спустя несколько секунд снова введите команду *cert-getapplist* и убедитесь в создании нового сертификата (Рисунок 169);

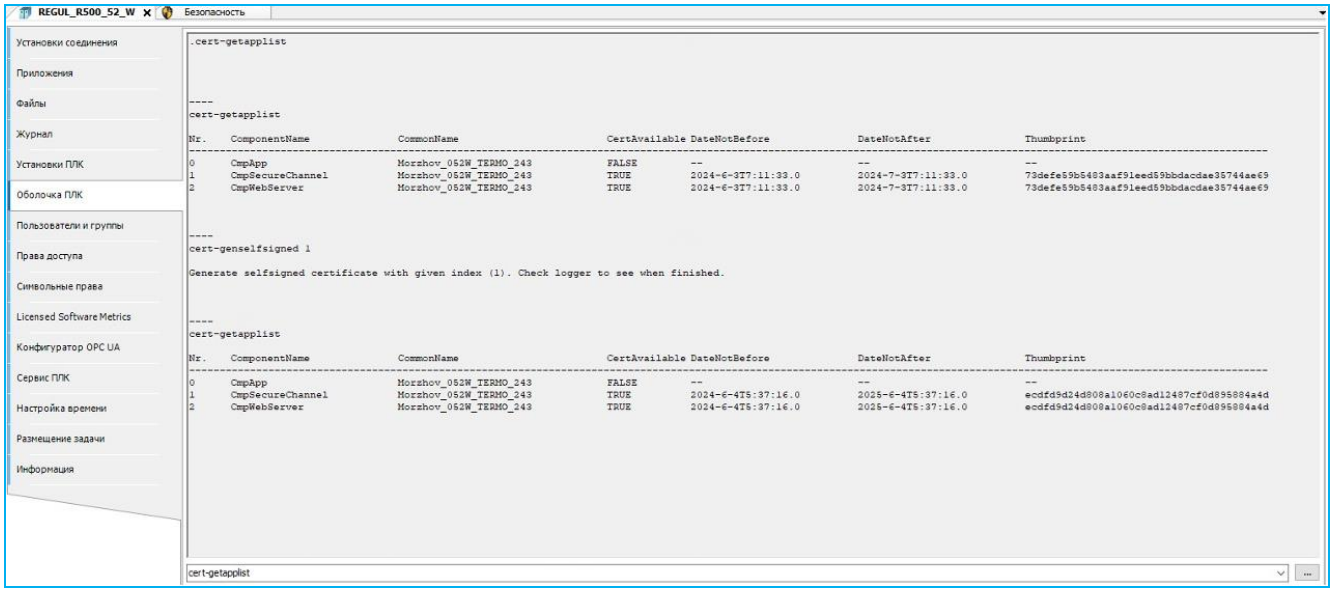


Рисунок 169 – Создание самоподписанного сертификата через **Оболочка ПЛК**

– нажмите на кнопку ***Yes*** (Рисунок 168) и откроется окно **Безопасность**. Также в раздел **Безопасность** можно попасть через основное меню **Вид** ⇨ **Безопасность**. На вкладке

Devices присутствуют сертификаты с истекшим сроком действия (подсвечены красным цветом), либо с истекающим сроком действия (Рисунок 170, подсвечены желтым цветом).

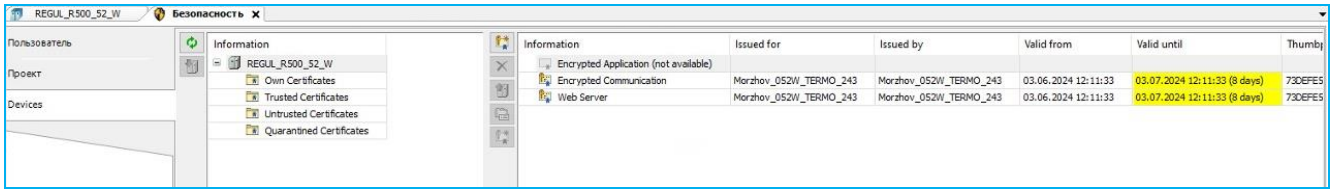


Рисунок 170 – Сертификаты с истекающим сроком действия

Выберите необходимый сертификат для продления и нажмите на кнопу с символом  (сертификаты с закрытым ключом). Откроется окно (Рисунок 171).

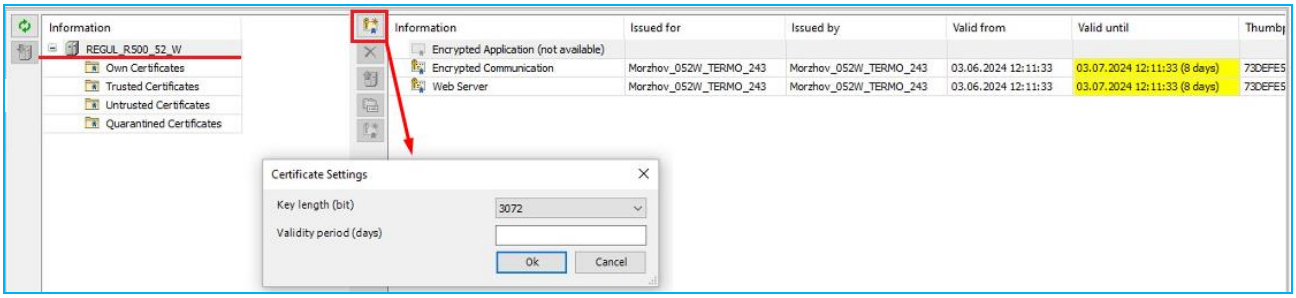


Рисунок 171 - Настройки сертификата

Задайте необходимые настройки:

- в поле **key length (bit) (длина ключа сертификата)**: отображается количество используемых при шифровании символов, измеряющееся в битах (по умолчанию - 3072). Для изменения значения выберите из выпадающего списка необходимый вариант: 2048; 3072; 4096 (Рисунок 172);

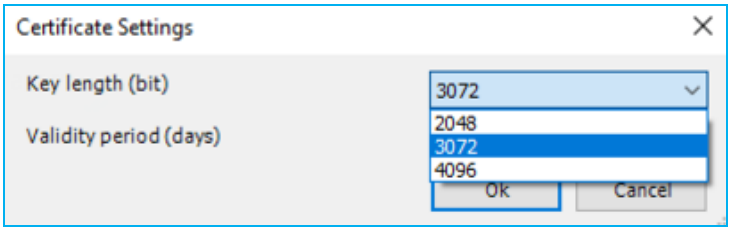


Рисунок 172 – Параметр. Длина ключа сертификата

- в поле **validity perid (days) (срок действия)**: указывается срок действия сертификата, измеряющегося в днях (максимальное значение - 3650 дней).

Нажмите на кнопку ***Ок***. По завершению процесса на вкладке **Devices** появятся обновленные сертификаты с заданным ранее сроком действия (Рисунок 173).

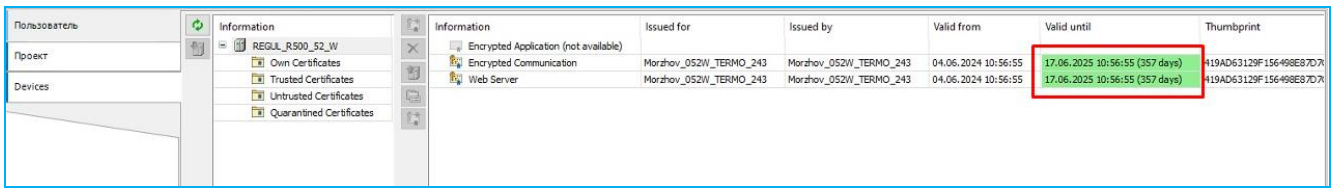


Рисунок 173 – Обновленные сертификаты

Необходимо повторять процедуру создания сертификата с более длительным сроком при каждом обновлении СПО.

Для изменения текущей настройки безопасности соединения зайдите на вкладку **Установки соединения,** выберите **Device ▾ (Устройство) ⇨ Change communication policy… (Изменить политику соединения…).** Откроется окно следующего вида (Рисунок 174):

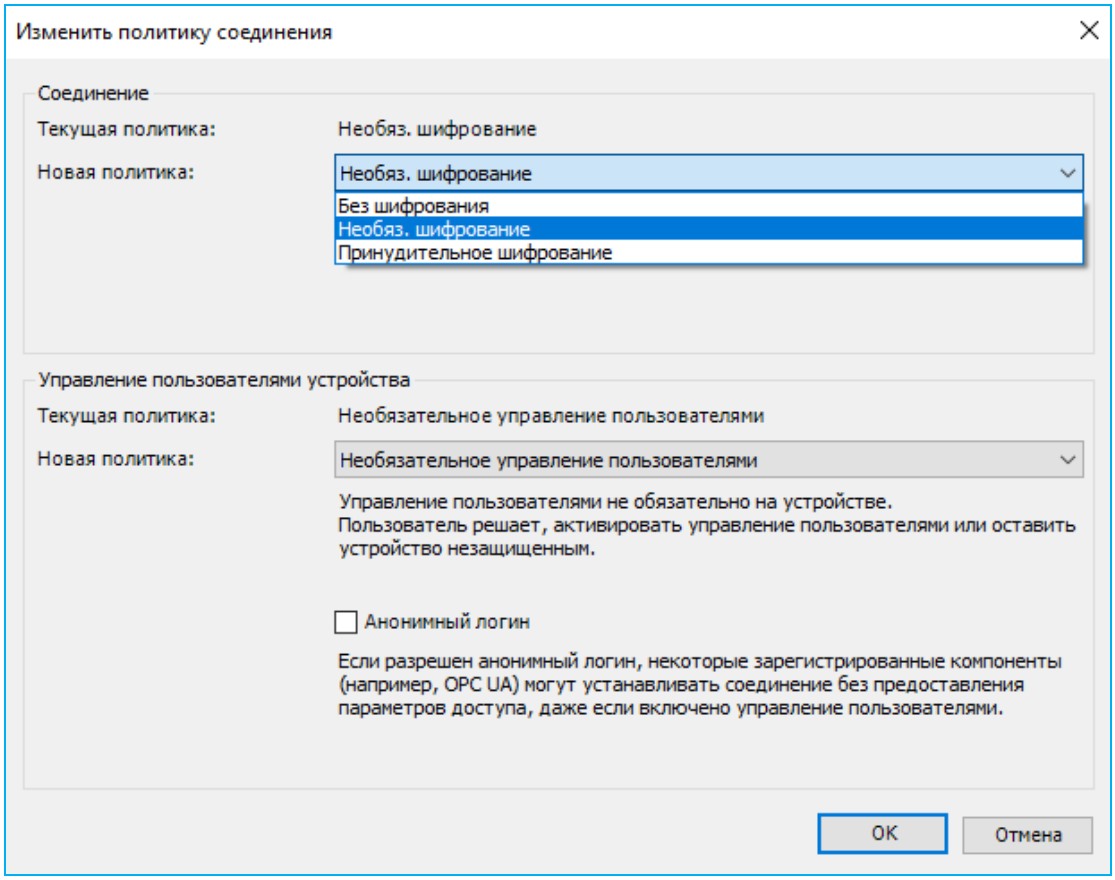


Рисунок 174 – Окно изменения политики соединения

В области **Изменить политику соединения (Communication)** в поле **Текущая политика (Current policy):** отображается текущее состояние политики соединения. Для изменения политики выберите в выпадающем списке в поле **Новая политика (New policy):** необходимый вариант:

- **Без шифрования (No encryption)** – без шифрования соединения;
- **Необязательное шифрование (Optional encryption)** – опциональное шифрование соединения (включение/выключение). Для того, чтобы включить шифрование, достаточно будет установить галочку в строке **Зашифрованное соединение.** Линия связи в области **Установки соединения** окрасится в желтый цвет (Рисунок 175);

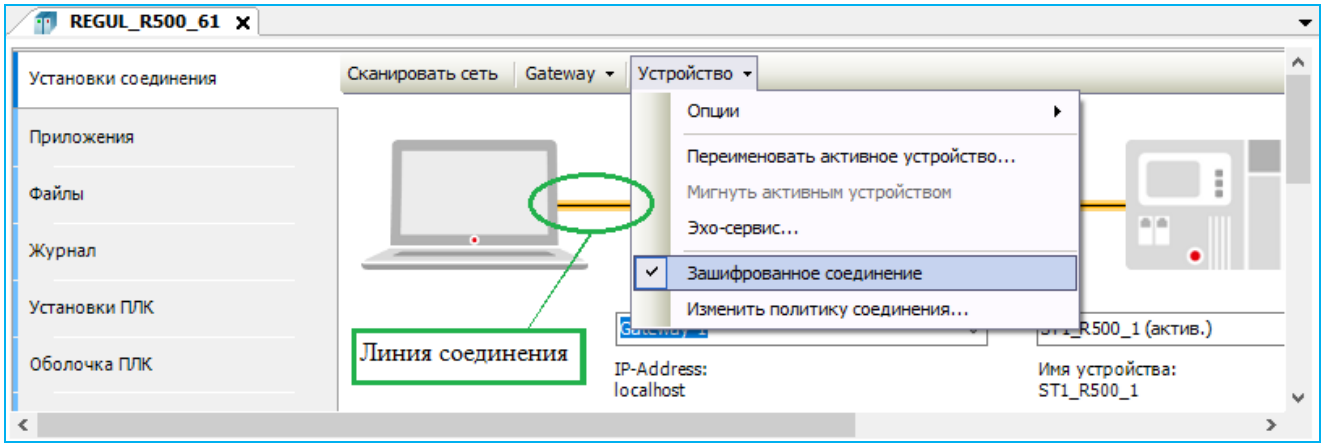


Рисунок 175 – Активация зашифрованного соединения активного

- **Принудительное шифрование (Enforced encryption)** – принудительное шифрование соединения. При каждом подключении к контроллеру соединение будет шифроваться автоматически.

При первом подключении к контроллеру (с включенным шифрованием) появится диалоговое окно с сообщением о том, что сертификат на контроллере не имеет подписи надежного центра. Последует запрос на его установку в локальном хранилище как надежного сертификата (Рисунок 176). В дальнейшем соединение с контроллером будет шифроваться автоматически этим сертификатом.

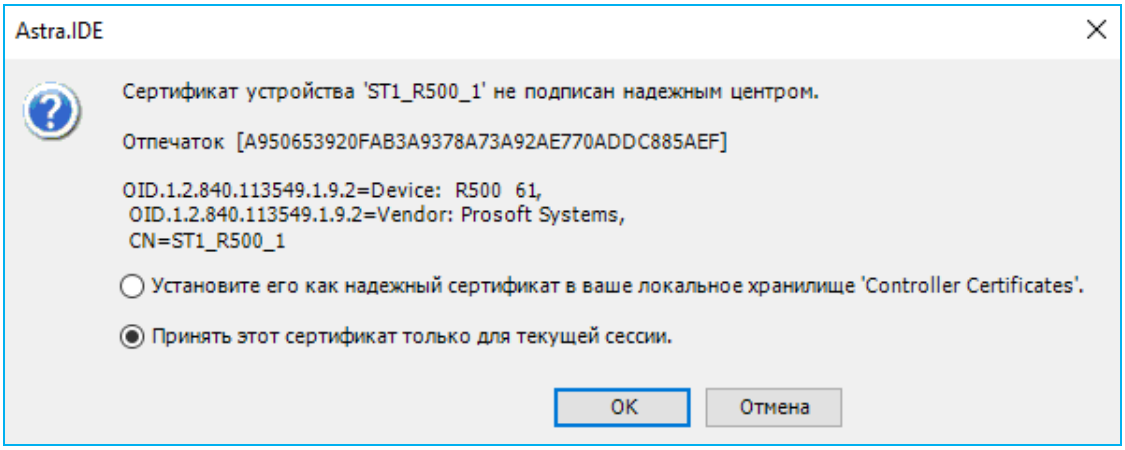


Рисунок 176 – Окно запроса

Добавление правил маршрутизации

В системном ПО контроллера заложена возможность задать статические правила маршрутизации для доступа к сетевым ресурсам. Такая необходимость может возникнуть при работе в сетях со сложной топологией. Для добавления правил маршрутизации требуется:

- настроить IP-адреса в системе;
- создать файл с правилами маршрутизации на ПК;
- перенести файл с правилами маршрутизации на контроллер.

Если IP-адреса предварительно не настроены, то маршрут не сможет быть добавлен по причине неизвестности исходного сетевого адреса (*Network is unreachable*).

На момент добавления нового правила маршрутизации системе должен быть известен маршрут до сетевого шлюза, в противном случае маршрут не будет добавлен по причине недостижимости сети (*Network is unreachable*). Для примера, приведенного ниже, маршрут до сетевого шлюза – это 192.168.1.1.

Создайте на компьютере файл *routes* (*routes.txt*) – файл с правилами маршрутизации в сети, куда будет подключен контроллер. Каждая строчка в файле - это новый маршрут.

Синтаксис**:** [-net|-host] [адрес_назначения] -netmask [маска_подсети] [шлюз]

или [-net|-host] [адрес_назначения]/[длина префикса сети] [шлюз]

Например:

-net 172.29.23.0 -netmask 255.255.255.0 192.168.1.1 или это же правило можно записать так:

-net 172.29.23.0/24 192.168.1.1

Параметры	Описание
-net	Сеть получатель
-host	Хост(узел) получатель
-netmask маска_подсети	Маска подсети, соответствующая сетевому IP-адресу. Например, 255.255.255.0 для маршрута к узлу. Либо после знака дроби указать длину префикса сети (количество единичных бит в маске)
шлюз	IP-адрес сетевого шлюза, через который будет выполняться отправка пакета

Запустите контроллер. Установите соединение контроллера с программой Astra.IDE (см. раздел «Подключение контроллера к сети»).

В дереве устройств поместите курсор на название контроллера. Двойной щелчок мыши открывает главную вкладку (окно) параметров устройства.

Перейдите на вкладку **Файлы**. В области **Хост** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на компьютере. Найдите файл *routes*.

В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc**.

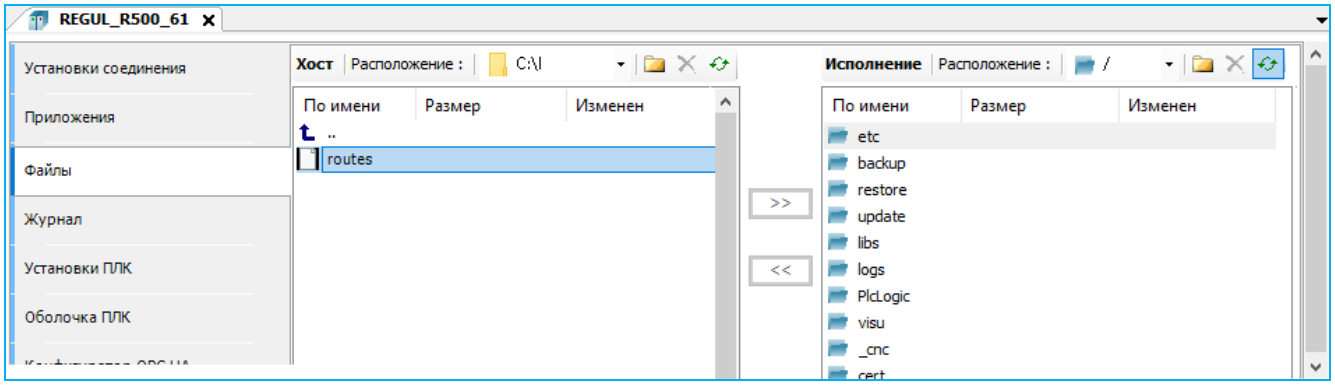


Рисунок 177 – Копирование файла с ПК на контроллер в папку etc

Кнопкой >> скопируйте файл *routes* с ПК на контроллер в папку **etc** (из **Хост** в **Исполнение**). Начнется загрузка файла на контроллер, в правом нижнем углу экрана появится индикатор хода загрузки. Дождитесь окончания процесса.

Правила маршрутизации автоматически применяются при любом изменении в файле *routes*.

С целью диагностики сетевых подключений перейдите на вкладку **Оболочка ПЛК**. Нажмите кнопку ... в нижнем правом углу. Откроется диалоговое окно с перечнем команд. Выберите нужную команду и нажмите кнопку ***Вставить***. Для выполнения команды нажмите клавишу ***Enter*** .

Команда **?** отображает список всех команд с пояснениями (Рисунок 178). В этом окне для выполнения команды введите текст в командной строке, нажмите клавишу ***Enter***.

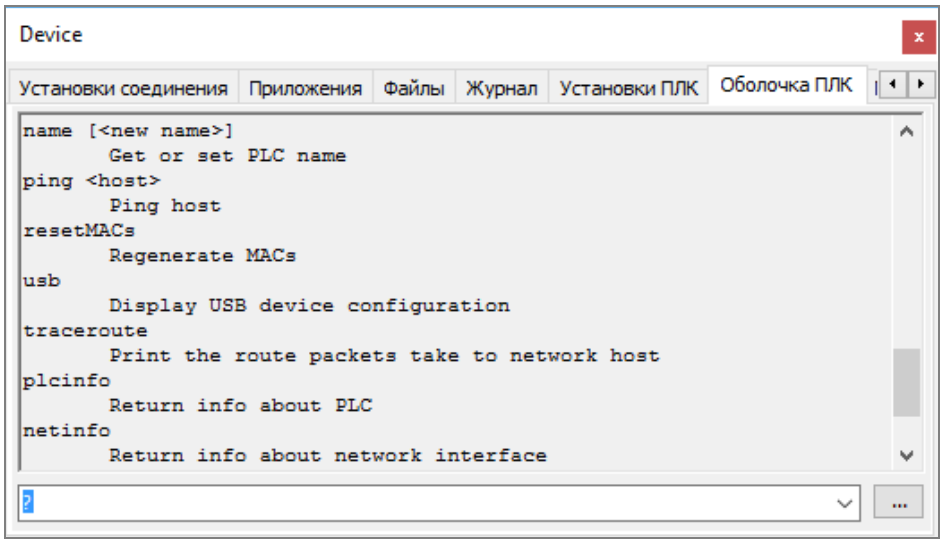


Рисунок 178 – Команды для диагностики маршрутов

Чтобы проверить, что контроллер подключается к другим устройствам сети согласно прописанным правилам маршрутизации, в программе Astra.IDE используются следующие команды из **Оболочки ПЛК**:

– **ping** – посылает на указанный хост пакет заданного размера, который затем возвращается обратно.

- **traceroute -** отображает путь пакета, то есть список узлов, через которые прошел (пройдет) пакет. В случае проблем при доставке данных до какого-либо узла программа позволяет определить, на каком именно участке сети возникли неполадки;
- **netinfo** – отображает информацию о сетевых интерфейсах контроллера.

Пакетный фильтр

Пакетный фильтр позволяет более гибко настраивать работу сетевой подсистемы и доступность сетевых сервисов контроллера для различных хостов. Данный фильтр обладает возможностью описания правил фильтрации, трансляции адресов (NAT), формирования очередей, что в свою очередь помогает в оптимизации обработки сетевых запросов для объектов, в которых контроллер интенсивно работает с сетью.

Фильтр пакетов изменяет, сбрасывает или передает пакеты в соответствии с правилами или определениями, расположенными в конфигурационном файле pf.conf (расположен в директории /etc). Если изменения в конфигурационном файле привели к невозможности подключения к ПЛК, то следует выполнить сброс до заводских настроек в соответствие с действиями, описанными в пункте «Сканирование сети» подраздела «Установка соединения с контроллером».

Журналирование ошибок и изменений правил фильтрации сетевых пакетов производится в лог-файл system.log (расположен в директории /logs/logger). Например, при внесении изменений формируется следующая запись:

«…Reloading packet filter rules».

Журналирование трафика пакетного фильтра производится в лог-файл (расположен в директории /logs/netdump, например:text_pflog.log). Для включения журналирования необходимо:

- в настройках пакетного фильтра pf.conf добавить ключевое слово log, например:

```
block in log proto icmp from 172.29.34.36
```

- в настройках netdump.conf (см. «Приложение 3») настроить журналирование интерфейса pflog0, например:

```
[config]
pflog0=on

[pflog0]
filename=text_pflog
output_format=text
packet_count=4
```

Файл pf.conf может содержать следующие компоненты:

Списки

Список позволяет объединить однородные параметры пакетного фильтра (адрес, порт и т.д.) во множества. Однородные параметры помещаются внутри правила в фигурных скобках {}. При загрузке фильтр раскладывает правило с N параметрами на N правил.

Пример:

```
block out on en1 from {192.168.1.2, 192.168.1.3} to any
```

```
block out on en1 from 192.168.1.2 to any
block out on en1 from 192.168.1.3 to any
```

Макросы

Пользователь может определить собственные переменные, содержащие параметры (адрес, порт и т.д.), и использовать их в дальнейшем, что позволяет упростить конфигурационный файл. Макросы необходимо определить до добавления ссылки на них в pf.conf. Использование макроса начинается с $.

Пример:

```
ext = “en1”
blocked = “{192.168.1.2, 192.168.1.3}”
block out on $ext from $blocked to any
```

Таблицы

Таблицы используются для хранения группы адресов IPv6 и/или IPv4. Поиск в таблице занимает меньше времени и потребляет меньше ресурсов, чем поиск в списке. Для создания таблицы используется команда table. Адреса могут также быть определены, используя модификатор типа отрицание (или "не").

Пример:

```
table <spammers> {192.0.2.0/24, !192.0.2.5}
```

В этом примере таблица spammers содержит адреса сети 192.0.2.0/24, за исключением адреса 192.0.2.5.

Опции

Опции определяют поведение механизма фильтрации пакетов. Устанавливаются командой set.

Пример:

```
set limit frags 2000
```

В этом примере опция устанавливает максимальное количество записей в пуле памяти, используемых для восстановления пакета из фрагментов.

Нормализация трафика

Нормализация трафика позволяет защитить машины во внутренней сети от конфликтов Интернет-протоколов и реализаций. Используется для очистки содержимого пакета и позволяет исключить неточную интерпретацию пакета на принимающей стороне. Нормализация пакетов запускается с помощью директивы scrub.

Пример:

```
scrub in all no-df
```

В этом примере команда удалит бит dont-fragment из всего входящего трафика на всех интерфейсах.

Очереди

Очереди обеспечивают управление полосой пропускания и установку приоритетов пакета. Интерфейсы, в которых требуется активировать формирование очередей, объявляются посредством директивы altq on.

Пример:

```
altq on port60 cbq bandwidth 5Mb queue {std, http, mail, ftp}
```

В этом примере интерфейс port60 должен формировать четыре четырехсекундные очереди до 5 Мбит/с с использованием формирования очередей на основе класса.

Трансляции

Правила трансляции определяют способы отображения адресов или перенаправления на другие адреса. Механизм трансляции модифицирует указанный адрес и/или порт пакета, выполняет перерасчет контрольных сумм IP, TCP и UDP по мере необходимости, и передает пакет в фильтр пакетов для обработки. Для активации изменения IP-адресов пакетов, при прохождении через интерфейс, используется утилита nat.

Пример:

```
nat on port60 from port50/24 to any -> port60
```

В этом примере реализуется трансляция на внешнем интерфейсе port60 для любого пакета из внутреннего интерфейса port50 заменой его IP-адреса внутреннего интерфейса на IP-адрес внешнего интерфейса.

Правила фильтрации

Правила фильтрации осуществляют выборочную фильтрацию пакетов на интерфейсах. Согласно правилу, пакет либо пропускается, либо отбрасывается. Правила фильтрации начинаются с действия, pass или block, что означает разрешить или блокировать трафик, соответственно.

Примеры:

```
block in all
```

В этом примере блокируются все входящие соединения.

```
pass in on port60 from 192.168.1.0/24 to 192.168.1.1
```

В этом примере разрешается трафик из внутренней сети на межсетевой экран с адресом 192.168.1.1.

Журналирование сетевого трафика

Пользователю доступна возможность перехватывать и анализировать сетевой трафик. Настройки по управлению журналированием сетевого трафика задаются в конфигурационном файле, который расположен в каталоге /etc/netdump.conf (см. «Приложение 3»).

Настройка Modbus

Подробное описание приведено в документе «Настройка обмена данными по протоколу Modbus на контроллерах серии REGUL RX00. Руководство пользователя».

Настройка IEC-104/101

Подробное описание приведено в документе «Настройка обмена данными по протоколу IEC 60870-5-101/IEC 60870-5-104 на контроллерах серии REGUL RX00. Руководство пользователя».

Настройка HART

Подробное описание приведено в документе «Настройка обмена данными по протоколу HART на контроллерах серии REGUL RX00. Руководство пользователя».

Настройка OPC DA

Подробное описание приведено в документе «Настройка и работа REGUL OPC DA Server. Руководство пользователя».

Настройка OPC UA

Подробное описание приведено в документе «Настройка и работа REGUL OPC UA Server. Руководство пользователя».

Настройка PROFIBUS DP

Подробное описание приведено в документе «Настройка обмена данными по протоколу PROFIBUS DP на контроллерах серии REGUL RX00. Руководство пользователя».

Настройка FIELDBUS FOUNDATION H1

Подробное описание приведено в документе «Настройка обмена данными по протоколу FIELDBUS FOUNDATION H1 на контроллерах серии REGUL RX00. Руководство пользователя».

Настройка IEC61850

Подробное описание приведено в документе «Настройка обмена данными по протоколу IEC61850 на контроллерах серии REGUL RX00. Руководство пользователя».

ПРОГРАММИРОВАНИЕ КОНТРОЛЛЕРА И ОТЛАДКА ПРОЕКТА

Создание ПЛК-программы

Редактор программы

Любой программный компонент, доступный в дереве POU или устройств, можно открыть двойным щелчком левой кнопки мыши (Рисунок 179). Также открыть объект в окне редактора можно командой контекстного меню **Редактировать объект**.

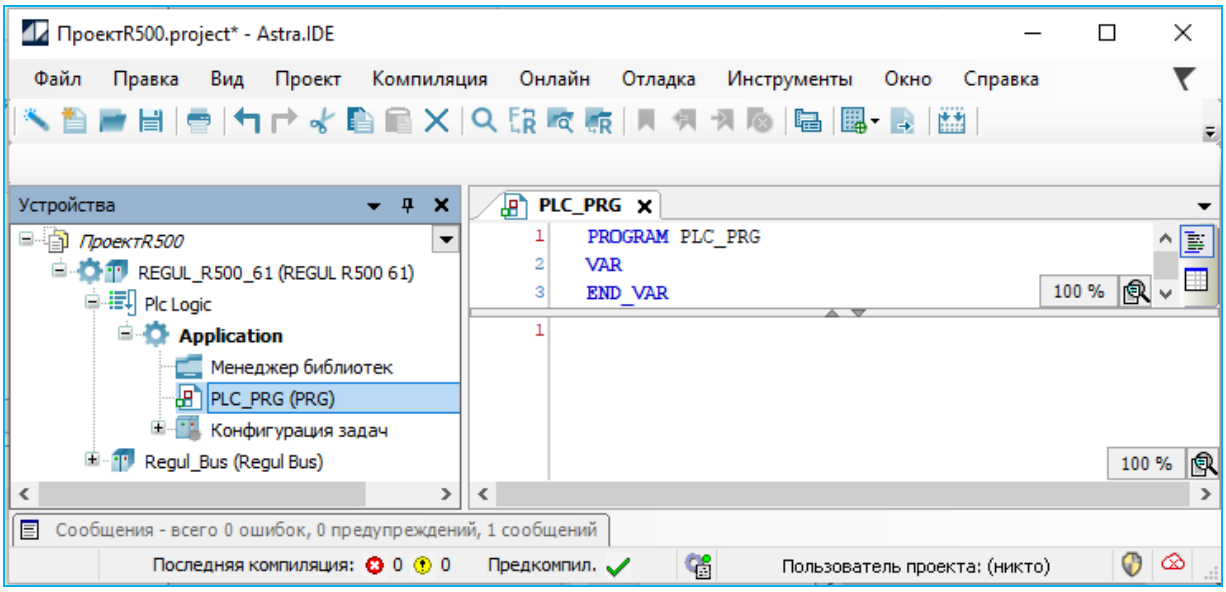


Рисунок 179 – Пример редактора программы (язык ST)

Для работы с программным компонентом PLC_PRG используется редактор, например, редактор языка ST (Рисунок 179). В этом редакторе имя объекта всегда отображается в строке заголовка окна. Окно редактора программы содержит редактор объявления в верхней части и раздела реализации (тело программы) в нижней части.

Объявление переменных

В редакторе объявления отображаются номера строк, тип и имя компонента (например, PROGRAM PLC_PRG), а также ключевые слова VAR и END_VAR для объявления переменных. Тело пусто, содержит пока только номер первой строки. В разделе объявления поместите курсор после VAR и нажмите клавишу ***Enter***. Будет вставлена пустая строка, в которой вы можете объявить переменные.

Объявление переменной должно выполняться в соответствии со следующими правилами:

– синтаксис: <Идентификатор > {AT <адрес>}:<Тип> {:=<начальное значение>};

i	**ИНФОРМАЦИЯ** То, что заключено в скобки {}, вводить не обязательно

– идентификатор не должен содержать пробелов или специальных символов;

– регистр не учитывается, то есть VAR1, Var1 и var1 это одна и та же переменная;

– учитывается символ подчеркивания (например, A_BCD и AB_CD - это два разных идентификатора. Идентификатор не должен содержать подряд более одного символа подчеркивания;

– длина идентификатора и его значимой части не ограничена.

Редактор объявлений позволяет использовать «режим быстрого набора». Эта функция выполняется, если вы завершаете ввод строки нажатием клавиш ***Ctrl*** + ***Enter***.

В онлайн-режиме редактор объявлений представляет собой окно просмотра.

Ввод программного кода

В тело программы (в область после разделительной черты) введите код программы.

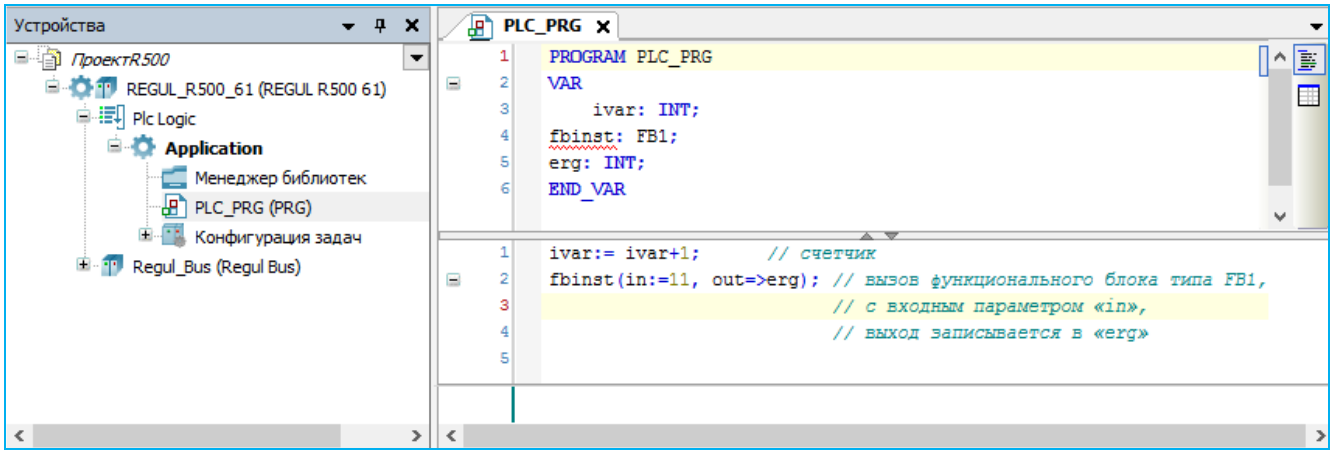


Рисунок 180 – Пример ввода программного кода

Отладка проекта

Компиляция и загрузка приложения в контроллер

Если вы хотите просто проверить вашу «активную» прикладную программу (**Application** в состоянии *Активное приложение*) на синтаксические ошибки, выберите в основном меню **Компиляция** ⇨ **Генерерировать код** или используйте клавишу ***F11*** (Рисунок 181).

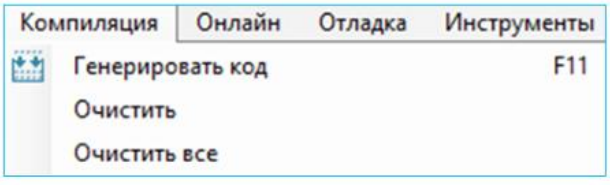


Рисунок 181 – Отладка проекта. Компиляция

Для компиляции всех приложений в проекте нажмите кнопку на панели инструментов в основном меню (Рисунок 182).

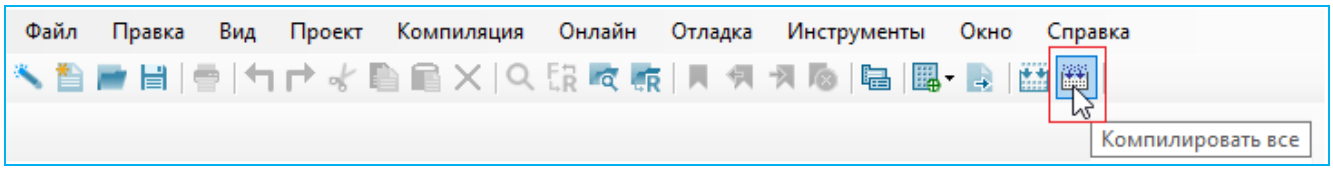


Рисунок 182 – Компиляции всех приложений

Негативные результаты проверки формируют ошибку. Информация, предупреждения и сообщения об ошибках будут отображаться в окне сообщений, которое по умолчанию располагается в нижней части интерфейса (Рисунок 183).

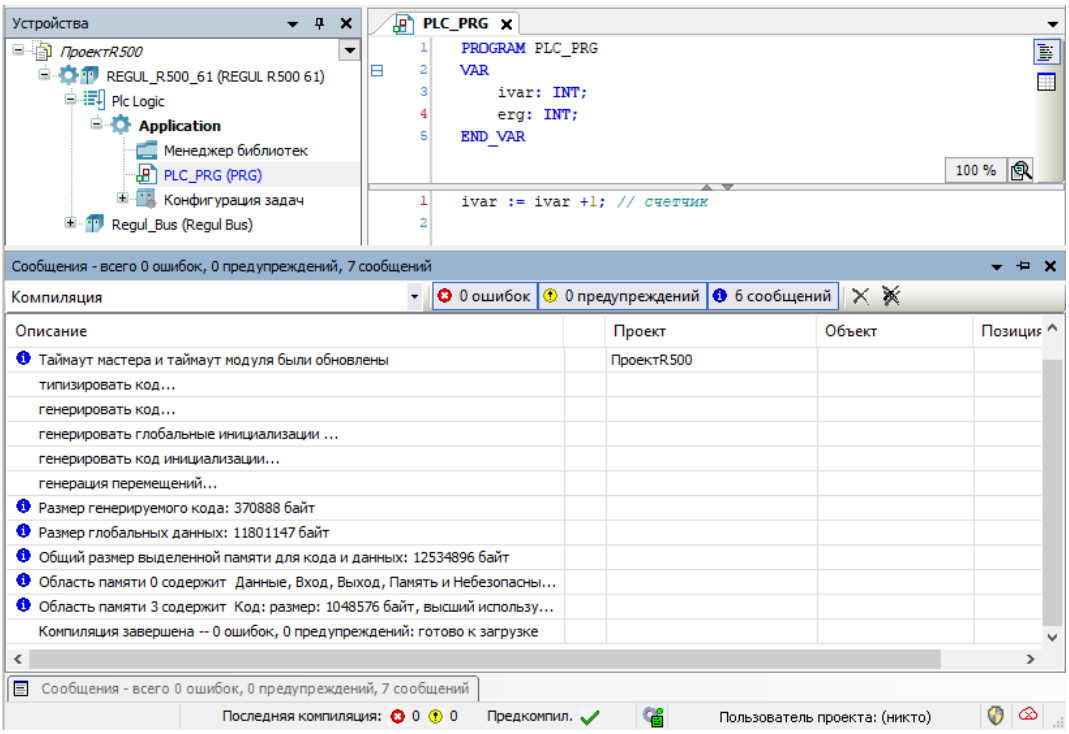


Рисунок 183 – Окно сообщений. Компиляция завершена

При компиляции производится проверка значения максимального объема памяти в проекте.

	**ВНИМАНИЕ!** Не рекомендуется менять максимальное значение объема памяти, принятое по умолчанию - 10485760 байт!

В случае крайней необходимости для изменения значения выполните следующее:

– поместите курсор на **Application** в дереве устройств, правой кнопкой мыши вызовите контекстное меню, выберите пункт **Свойства.** Откроется окно **Свойства⇨**вкладка **Опции компиляции приложения** (Рисунок 184);

– измените значение в поле **Максимальный объем памяти:**

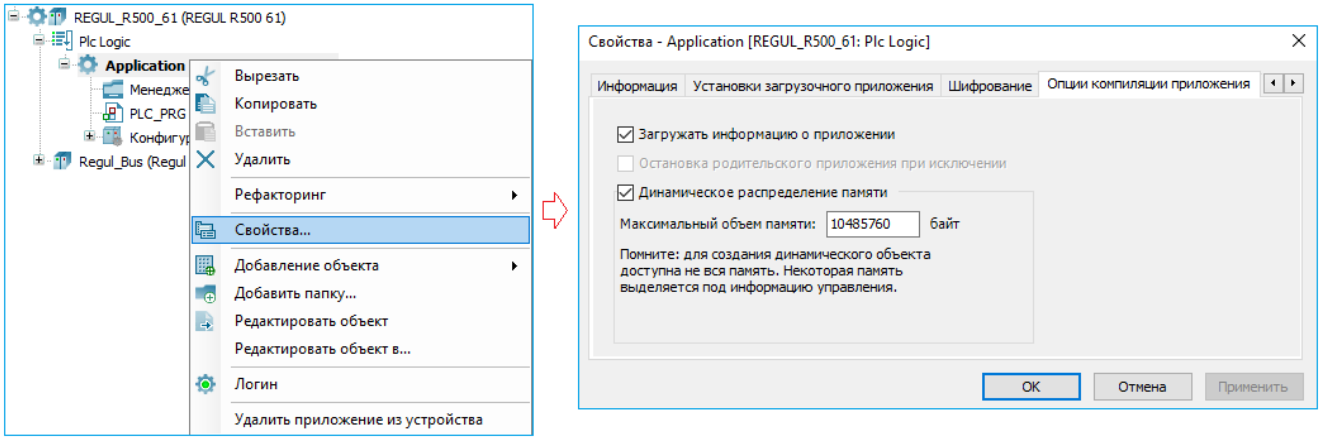


Рисунок 184 – Окно «Свойства»

Если размер динамически распределяемой памяти был указан недостаточным, то в окне сообщений появится запись об ошибке следующего вида: «Указан недостаточный размер динамически распределяемой памяти...».

Установите соединение с контроллером.

Для загрузки приложения в контроллер выберите в основном меню **Онлайн** ⇨ **Логин** или нажмите кнопку на панели инструментов. Загрузить проект в контроллер также можно через меню **Онлайн** ⇨ **Загрузка**.

Начнется загрузка проекта в контроллер. Если загружаемый проект отличается от того, что запущен в настоящий момент на контроллере, появится сообщение (Рисунок 185).

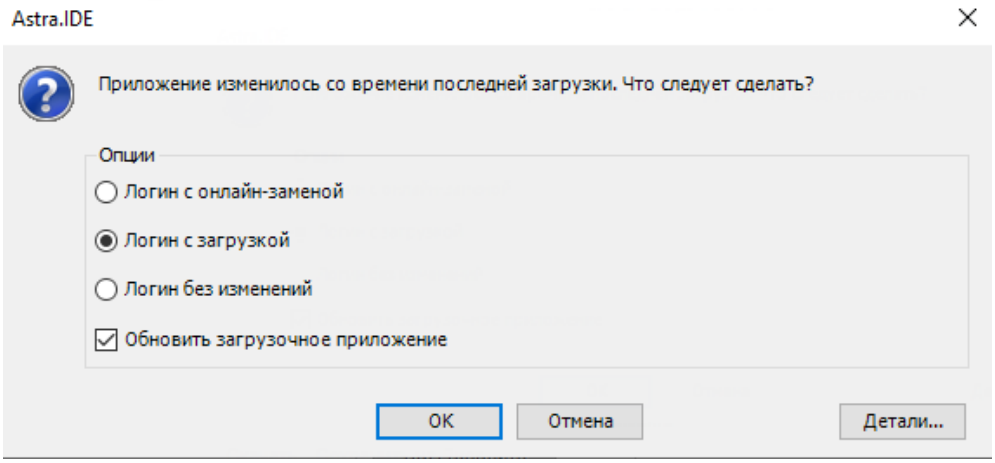


Рисунок 185 – Сообщение при загрузке проекта

Логин с онлайн-изменением позволяет загрузить не весь проект, а только измененные его части. Другими словами, выполнение онлайн-изменения будет автоматически предложено при подключении к контроллеру с прикладной программой, которая на нем уже запущена, но с момента последней загрузки была изменена.

i	ИНФОРМАЦИЯ **Логин с онлайн-изменением** будет не доступен, если при изменении проекта изменилась его конфигурация, а также внесены изменения в: – параметры устройства; – дерево устройства; – объект TaskConfiguration. В этом случае будет доступна только полная загрузка (**логин с загрузкой**) проекта с последующей перезагрузкой приложений

Логин с загрузкой – это загрузка полностью нового проекта, взамен существующего на контроллере.

Логин без изменений подключается к ранее загруженному проекту.

Обновить загрузочное приложение – установленный флажок приводит к созданию загрузочного приложения с измененным кодом и автоматической отправкой его в ПЛК. В случае последующей перезагрузки контроллера запустится уже измененное приложение. Если снять флажок с поля, то изменённый код загрузочного приложения не вступит в силу и при перезагрузке запустится старое приложение без внесенных изменений. По умолчанию флажок установлен.

Для отключения от контроллера выберите в основном меню **Онлайн ⇨ Отключение** или кнопку .

Запуск и мониторинг приложения

Для запуска загруженного приложения выберите в основном меню **Отладка ⇨ Старт,** либо с помощью кнопок Старт/Стоп (Рисунок 186).

Рисунок 186 – Кнопки Старт / Стоп в панели меню

При запуске приложения в строке состояния должна отобразиться надпись «Запуск» на зеленом фоне (Рисунок 187).

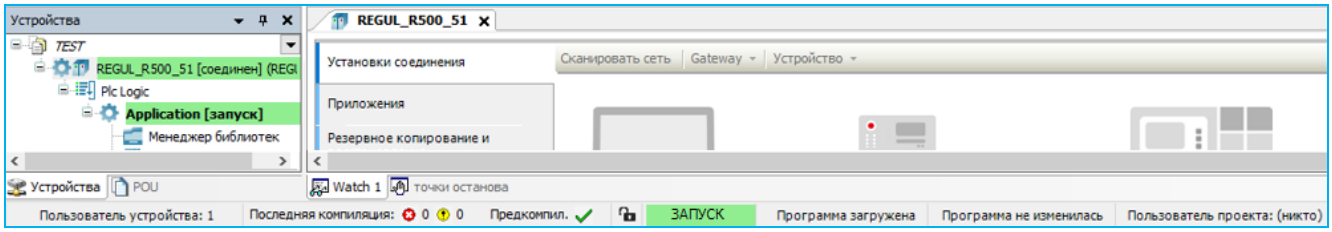


Рисунок 187 – Приложение запущено на контроллере

	**ИНФОРМАЦИЯ** При запуске приложения с большим объемом прикладного кода возможен выход за пределы времени таймаута сессии со средой Astra.IDE. При превышении таймаута происходит разрыв сессии. Длительность таймаута настраивается в конфигурационном файле *etc/runtime.cfg*. По умолчанию таймаут равен 30000 мс

Также запуск/останов приложения можно производить с помощью переключателей RUN/STOP, расположенных на лицевой панели модуля ЦП и выполняющих аналогичный функционал, что и кнопки Старт/Стоп в среде программирования.

Переключатели RUN/STOP работают в двух режимах, и при необходимости пользователь может задавать приоритеты над органами управления приложением.

Равнозначный режим (по умолчанию)

В этом режиме переключатель RUN/STOP имеет равнозначный вес по сравнению с органами управления из среды Astra.IDE и предполагает следующее:

- **смена позиции STOP→RUN или RUN→STOP** активирует событие RUN и STOP соответственно;
- **запуск приложения пользователем из Astra.IDE** при положении переключателя *STOP* не ограничен;
- **останов приложения пользователем из Astra.IDE** при положении переключателя *RUN* не ограничен;
- процедура **загрузки приложения**, а, равно как и **процедура теплого/холодного сброса** приложения (см. пункт ниже «Сброс приложений»), переводит активное приложение в состояние STOP;
- процедура **автозагрузки приложения** переводит активное приложение в состояние, определенное положением переключателя.

Первичный режим

Для активации (см. подраздел «Настройка системных параметров»), в секции PsLed конфигурационного файла *etc/runtime.cfg*, необходимо указать опцию:

```
[PsLed] RunStopButtonSuperior = 1
```

В этом режиме переключатель RUN/STOP имеет приоритетный вес по сравнению с органами управления из среды Astra.IDE и предполагает следующее:

- **смена позиции STOP→RUN или RUN→STOP** активирует событие RUN и STOP соответственно;
- **запуск приложения пользователем из Astra.IDE** будет отменен при положении переключателя *STOP*;

- **останов приложения пользователем из Astra.IDE** будет отменен при положении переключателя *RUN*;
- процедура **загрузки приложения**, а, равно как и **процедура теплого/холодного сброса** приложения, переводит активное приложение в состояние, определенное положением переключателя;
- процедура **автозагрузки приложения** переводит активное приложение в состояние, определенное положением переключателя.

Сброс приложений

Сброс приложения останавливает прикладную программу и возвращает переменным их начальные значения. В зависимости от типа сброса сбрасываются энергонезависимые переменные (RETAIN) и/или энергонезависимые постоянные переменные (PERSISTENT RETAIN). Запустить команду сброс приложения можно через основное меню **Онлайн** ⇨ **Сброс / Сброс холодный / Сброс заводской** в онлайн-режиме (Рисунок 188).

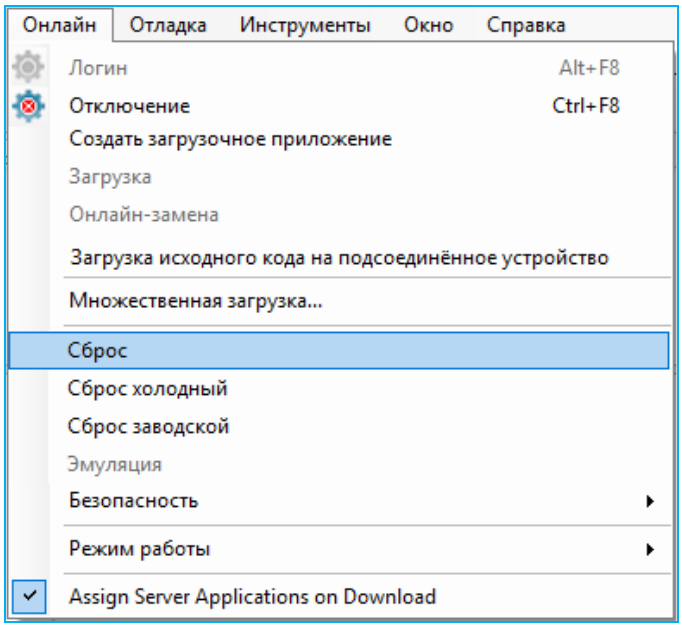


Рисунок 188 – Запуск команды Сброс через основное меню

Сброс подразделяется на следующие типы:

- **Сброс** (теплый) – сбрасываются все переменные в их начальные значения, за исключением **RETAIN** и **PERSISTENT RETAIN**. Ситуация аналогична выключению→включению питания контроллера во время работы приложения;
- **Сброс холодный** – сбрасываются все переменные в их начальные значения, за исключением **PERSISTENT RETAIN**;
- **Сброс заводской** – сбрасываются все переменные в их начальные значения и удаляется прикладная программа с контроллера.

В таблице 8 представлено поведение переменных при различных командах сброс.

Таблица 8 – Поведение переменных

Онлайн-команды	**Переменные**		
	VAR	**VAR RETAIN**	**VAR PERSISTENT RETAIN**
Сброс(теплый)	x	✓	✓
Сброс холодный	x	x	✓
Сброс заводской	x	x	x
Примечание: «✓» – переменные неизменны; «x» – переменные сбрасываются			

ВНИМАНИЕ!

В резервированном контроллере, при заводском сбросе одного из ЦП, сбрасывается приложение и на ЦП-партнере

Работа с RETAIN и PERSISTENT RETAIN переменными

Переменные объявляются с использованием ключевых слов **RETAIN** и **PERSISTENT RETAIN**. Переменные занимают общую энергонезависимую область памяти ПЛК.

RETAIN и **PERSISTENT RETAIN** переменные синхронизируются при любом изменении, даже если была произведена запись того же значения. Для уменьшения нагрузки на ЦП необходимо уменьшить частоту записи в эти переменные. Период записи определяется файловой системой, при этом максимальный период записи данных на диск составляет 10 с.

По умолчанию для **RETAIN** переменных используется область памяти **SRAM** (хранилище) объемом до 1Мб. При большом количестве **RETAIN** переменных можно изменить тип хранилища (File) для сохранения переменных в файл, размер которого можно задать. Для изменения выполните следующие действия:

– перейдите на вкладку **Сервис ПЛК ⇨Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку  (*Обновить*);

– добавьте в секцию [PsRetain] конфигурационного файла *etc/runtime.cfg* параметр *StorageType,* указав:

```
[PsRetain]
StorageType=File
```

– с помощью параметра *StorageSize* можно задать объем хранилища в Кб (по умолчанию 1 Кб, максимальный – 100 Мб), например:

```
[PsRetain]
StorageType=File
StorageSize=2048
```

Нажмите кнопку **Сохранить** Для вступления в силу изменений потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК.**

Следует учесть, что **PERSISTENT RETAIN** переменные объявляются добавлением в приложение программного объекта **Persistent-переменные…** (Рисунок 189).

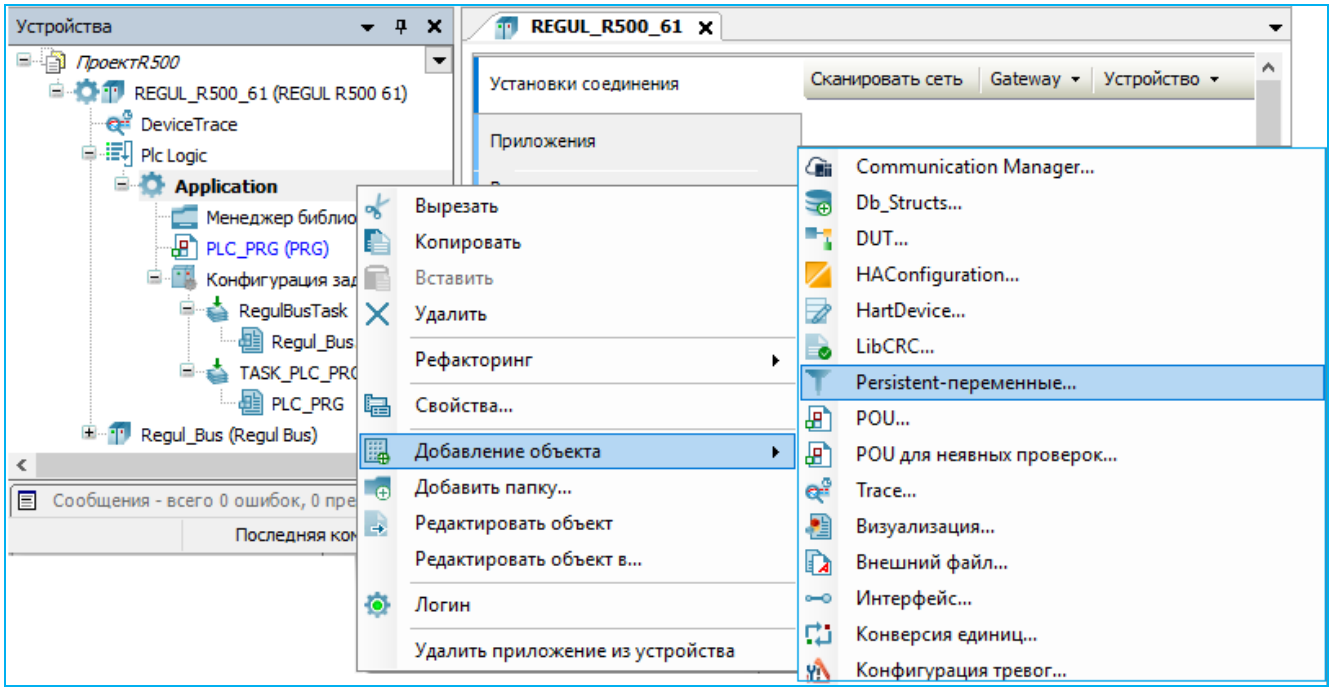


Рисунок 189 – Добавление энергонезависимых постоянных переменных

Сохранение и восстановление значений переменных RETAIN

RETAIN переменные инициализируются заново (сбрасываются в их начальные значения) при выполнении команды «холодный сброс», «заводской сброс» и при загрузке приложения. Если необходимо сохранить значения переменных RETAIN, при выполнении этих команд, то их необходимо сохранить вручную. Для сохранения и последующего восстановления можно воспользоваться командами

– **saveretain** - сохранение данных в файл [имя файла];

– **restoreretain** - восстановление данных из файла [имя файла].

Для сохранения данных остановите приложение и перейдите на вкладку «Оболочка ПЛК». В командной строке введите команду *saveretain* с [<имя_приложения>], например (имя_приложения - TEST_retain):

saveretains TEST_retain

Данные будут сохранены в файл с расширением *.ret на ПЛК: …/[<имя_приложения>].ret.

Для восстановления данных в командной строке введите команду *restoreretain* с [<имя_приложения>], например:

restoreretains TEST_retain

После каждой успешной операции будут выводиться сообщения о сохранении/восстановлении RETAINS из приложения в файл, либо в приложение из файла соответственно (Рисунок 190).

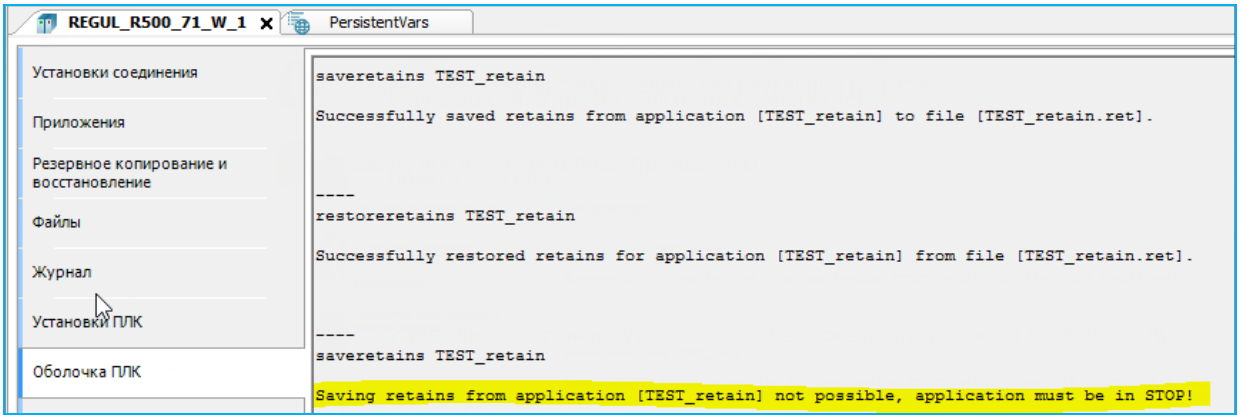


Рисунок 190 - Сохранение/восстановление RETAINS

ВНИМАНИЕ!

Сохранение и восстановление значений RETAIN переменных возможно только при остановленном приложении, иначе выполнение операции невозможно

Онлайн-наблюдение за конкретным POU

Для просмотра выполнения ПЛК-программы откройте редактор программы компонента (например, PLC_PRG) двойным щелчком левой кнопки мыши. Редактор программы, в котором ранее объявлялись переменные, вводился код программы, текст программы можно было отредактировать, в онлайн-режиме изменил вид (Рисунок 191).

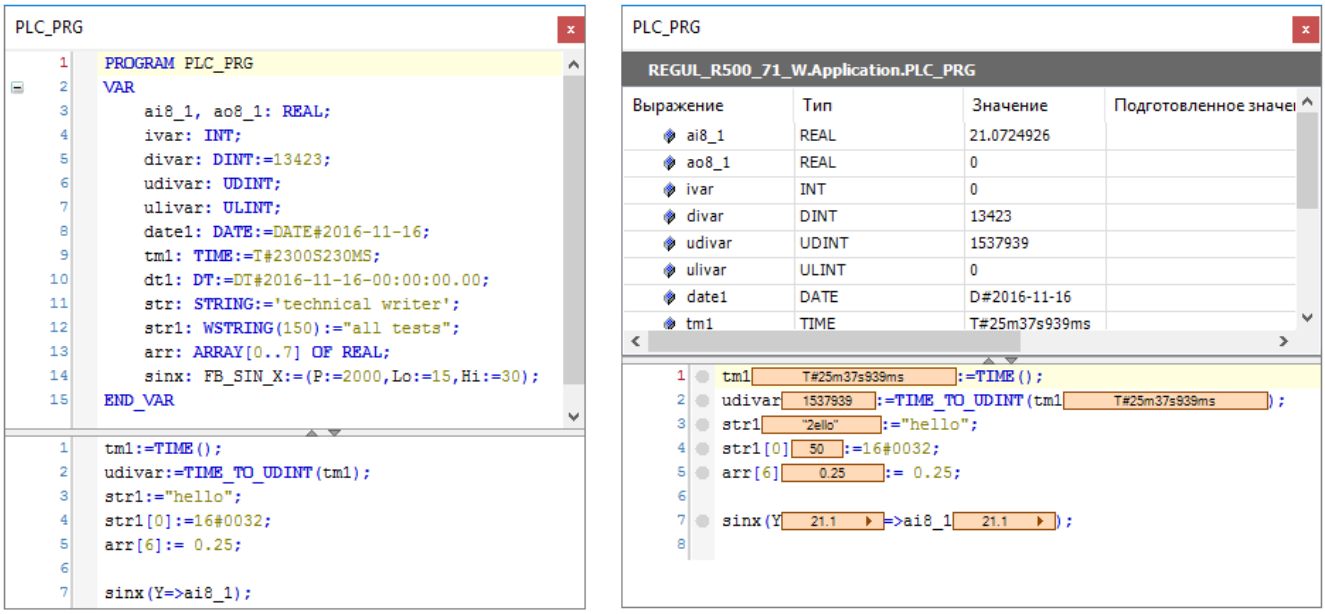


Рисунок 191 – Редактор программы 1) режим редактирования 2) онлайн-работа программы

В редакторе объявления (в верхней части окна) отображаются текущие значения переменных. В разделе реализации (в нижней части окна) отображаются строки кода программы. Рядом с каждой переменной расположено маленькое окошко мониторинга, в котором отображается текущее значение переменной (встроенный мониторинг).

Запись и фиксация переменных

С помощью команды **Записать значения** (из пункта основного меню **Отладка**) вы можете перед началом рабочего цикла записать в одну или несколько переменных заранее определенные вами значения. Запись заданного вами значения осуществляется один раз в начале цикла.

Последовательность выполнения цикла:

1. Чтение входов,
2. Запись переменных,
3. Выполнение кода программы,
4. Запись выходов.

В режиме онлайн откройте редактор программного компонента. В разделе объявлений редактора вы увидите таблицу, в которой перечислены отображаемые выражения. Щелкните по соответствующему полю в столбце **Подготовленное значение**, введите нужное значение (Рисунок 192). Выполните команду **Записать значения**. Введенное значение будет немедленно записано в соответствующем поле в столбце **Значение**. Теперь оно записано в контроллер. Поле **Подготовленное значение** станет опять пустым.

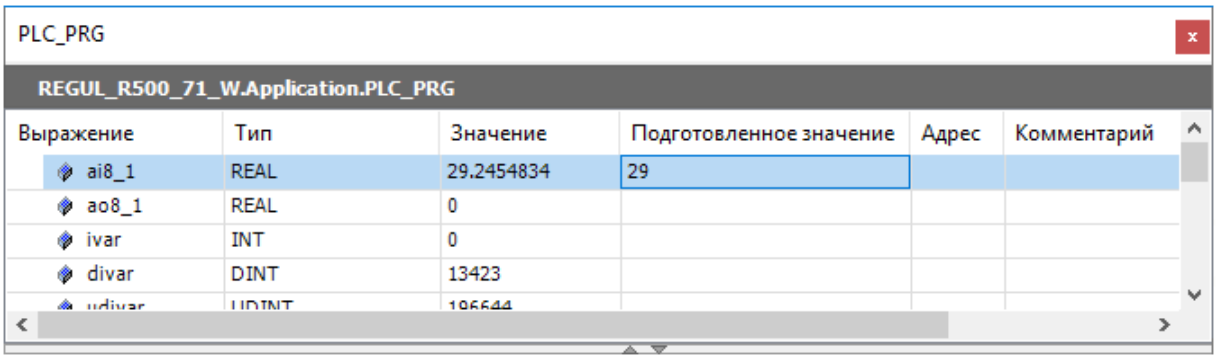


Рисунок 192 – Запись и фиксация переменных

С помощью команды **Фиксировать значения** (из пункта основного меню **Отладка**) можно зафиксировать значения одной или нескольких переменных. Запись заданного вами значения осуществляется в начале и в конце каждого управляющего цикла.

Последовательность выполнения цикла:

1. Чтение входов,
2. Фиксация переменных,
3. Выполнение кода программы,
4. Фиксация переменных,
5. Запись выходов.

Фиксация значений осуществляется аналогично действиям по записи значений.

Фиксация будет осуществляться до тех пор, пока вы не остановите ее для некоторых или всех переменных, или пока система не будет отключена.

Задание точек останова и пошаговое выполнение программы

При отладке проекта в среде разработки возможно использование точек останова. Точки останова – это места, в которых выполнение программы будет приостанавливаться, что позволяет просмотреть значения переменных на определенном этапе работы программы. После того, как программа достигла точки останова, ее выполнение можно продолжить по шагам.

В онлайн режиме откройте редактор нужного вам программного компонента, выделите требуемую строку и нажмите клавишу ***F9*** либо выберите в основном меню **Отладка** ⇨ **Переключить точку останова**. После этого точка останова появится на экране (Рисунок 193).

После попадания программы в точку останова ее (программу) можно будет выполнять по шагам с помощью клавиши ***F8*** или команды **Отладка** ⇨ **Шаг детальный**. При этом будут также выполняться шаги экземпляра функционального блока. Чтобы пропустить шаги функционального блока, используйте клавишу ***F10*** или команду **Отладка** ⇨ **Шаг поверху**.

При этом текущие значения всех переменных на контроллере будут отображаться на текущей позиции выполнения программы.

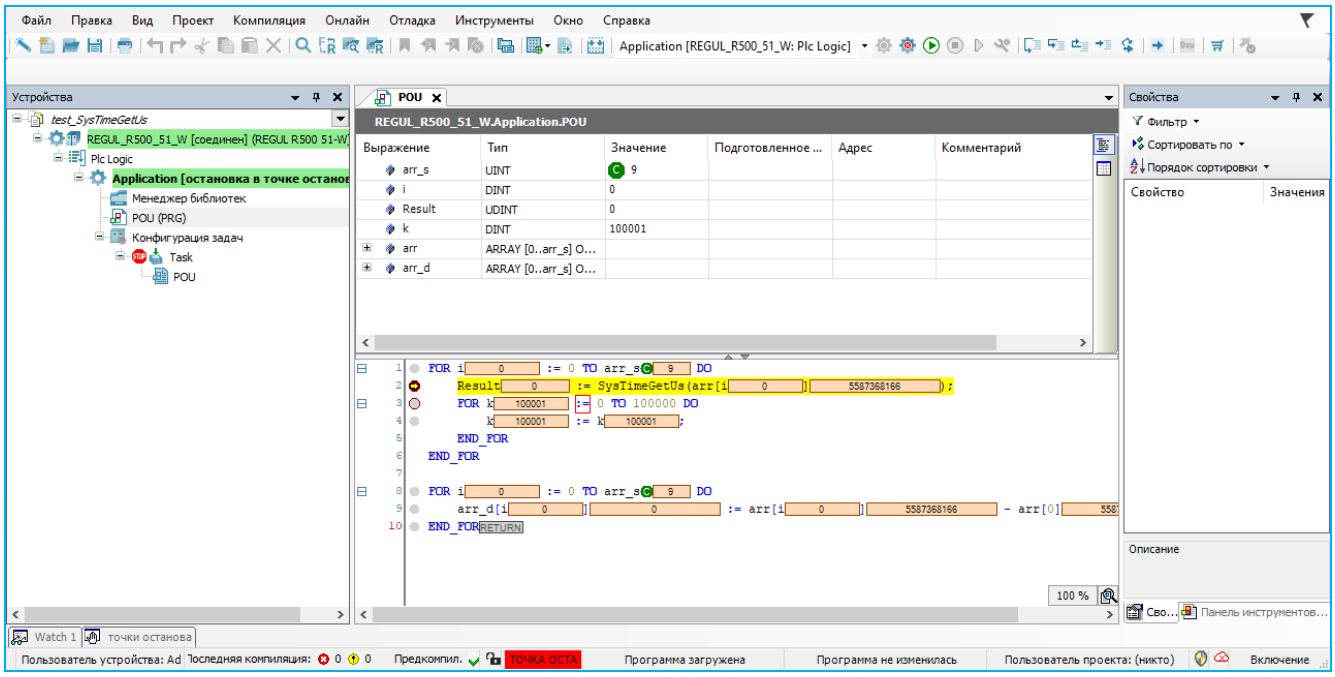


Рисунок 193 – Точка останова в ПЛК-программе

Чтобы просмотреть всю информацию по точкам останова выберите в основном меню **Вид** ⇨ **Точки останова**. Откроется окно **Точки останова**. Здесь можно просмотреть и изменить текущие точки останова, а также задать новые (Рисунок 194).

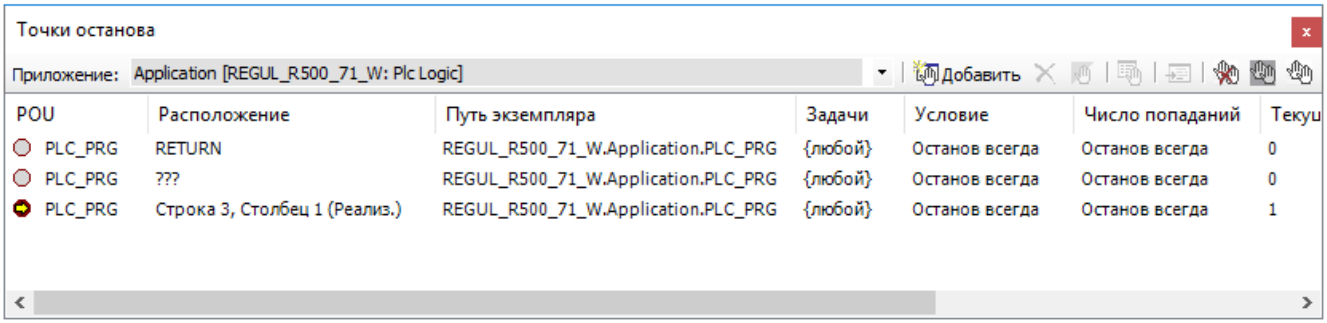


Рисунок 194 – Окно «Точки останова»

Режимы работы контроллера

Среда разработки обеспечивает возможность назначения контроллеру трех различных режимов работы:

– режим **Отладка** (, включен по умолчанию), при котором допускается выполнение любых команд, в том числе удаление и обновление приложения;

– режим **Заблокировано** ( ), текущее состояние отладки зафиксировано в приложении, сохраняется состояние «RUN» приложения, даже при перезагрузке контроллера. Этот режим необходим для предотвращения загрузки на неподходящий контроллер в сети с

несколькими ПЛК, разрешая загрузку только на те из них, которые находятся в режиме «Отладка». В режиме **Заблокировано** никакие дополнительные точки останова не могут быть установлены, и никакие дополнительные переменные не могут быть принудительно заданы. Запись переменных по-прежнему возможна, а уже установленные точки останова остаются включенными;

– режим **Рабочий** (), при котором запрещены любые команды, влияющие на приложение (загрузка, удаление и т.д.) Этот режим работы гарантирует, что контроллер перезагружает те же приложения после перезапуска и никакие функции отладки больше не активируются. Точки останова не могут быть установлены. Режим используется после ввода контроллера в эксплуатацию. Условия, при которых активируется ***рабочий*** режим:

- на контроллере присутствует загрузочное приложение;
- не должно быть установленных активных точек останова;
- все приложения должны быть запущены;
- не должно быть зафиксировано значение какой-либо переменной.

⚠	**ВНИМАНИЕ!** При вводе системы управления в эксплуатацию настоятельно рекомендуем переводить контроллер в **Рабочий** режим, для предотвращения случайного изменения приложения или активации функции отладки, что может негативно повлиять на технологический процесс

Для смены режима работы выберите в основном меню **Онлайн ⇨ Режим работы** и выберите команду **Заблокировано/Рабочий** (Рисунок 195).

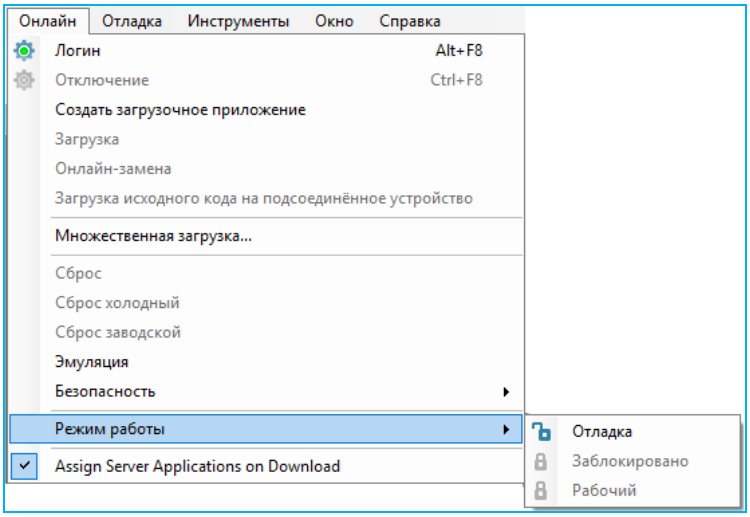


Рисунок 195 - Определение режима работы

Если контроллер переведен в режим **Заблокировано** или **Рабочий**, то при попытке загрузить новый проект появится информационное окно, с указанием ошибки. Для загрузки нового проекта: установите соединение с контроллером (без загрузки проекта) и, перейдя в основное меню **Онлайн ⇨ Режим работы,** выберите команду **Отладка**.

Переключение между режимами **Заблокировано** и **Рабочий** невозможно.

В строке состояния отображается символ (/ /), указывающий на текущий режим работы (Рисунок 196). Двойной щелчок по символу открывает окно справки.

Рисунок 196 - Текущий режим работы

ОБСЛУЖИВАНИЕ КОНТРОЛЛЕРА

Настройка системных параметров

Для начала подключитесь к контроллеру и перейдите на вкладку **Сервис ПЛК** ⇨ **Системные параметры.** Далее нажмите кнопку  (***Обновить***). На экран будет выведена информация о текущем состоянии доступных параметров (Рисунок 197). По умолчанию все параметры выключены.

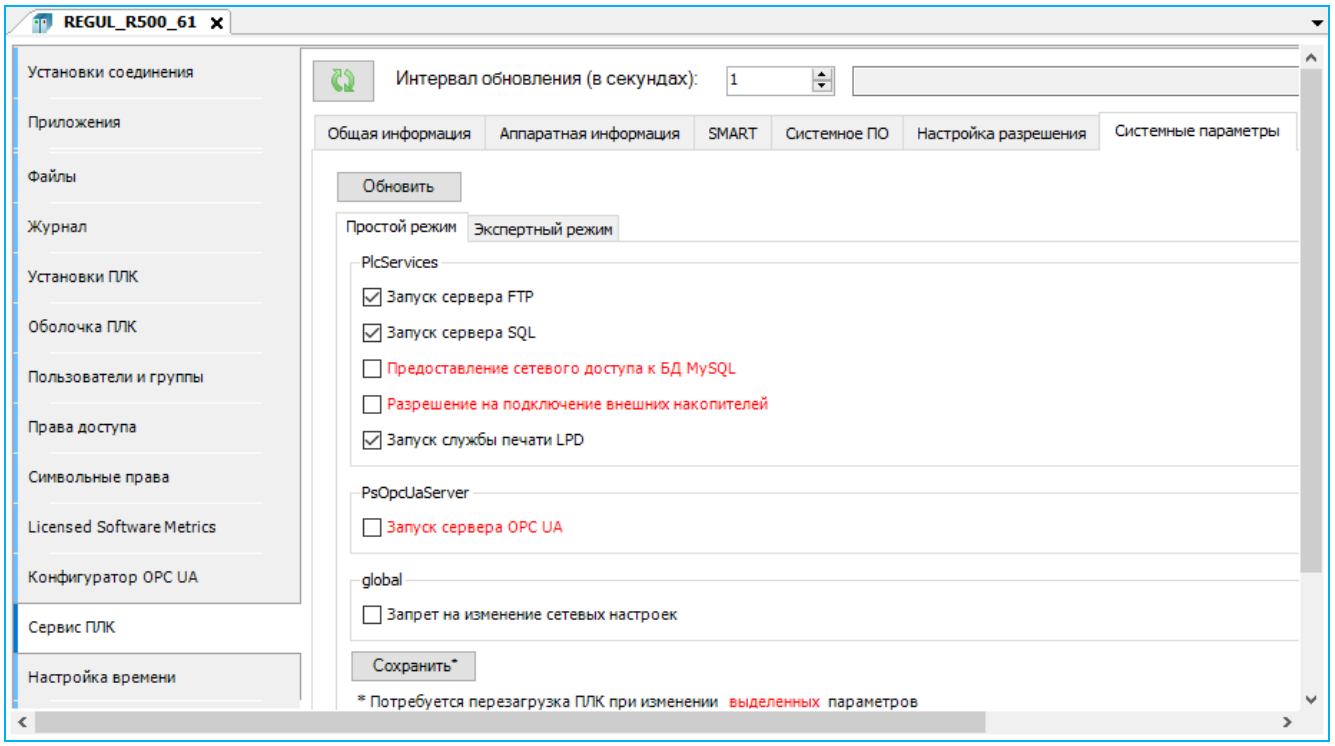


Рисунок 197 – Вкладка с системными параметрами в простом режиме

Выберите вкладку: **Простой режим** или **Экспертный режим**, для определения способа задания параметров.

На вкладке **Простой режим** представлен минимальный набор параметров для настройки системы с упрощенным способ активации. Для активации: установите флажок в выбранном поле и нажмите кнопку **Сохранить**. Учтите, для вступления в силу изменений выбранных параметров, названия которых выделены красным цветом, потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

На вкладке **Экспертный режим** представлены секции для ввода и редактирования параметров в виде текстовых строк. Выберите конфигурационный файл, в который будут внесены изменения. Перечень конфигурационных файлов представлен на рисунке 198.

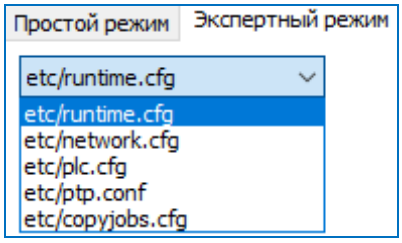


Рисунок 198 - Конфигурационные файлы

Пример конфигурационного файла **etc/runtime.cfg** приведен на рисунке 199.

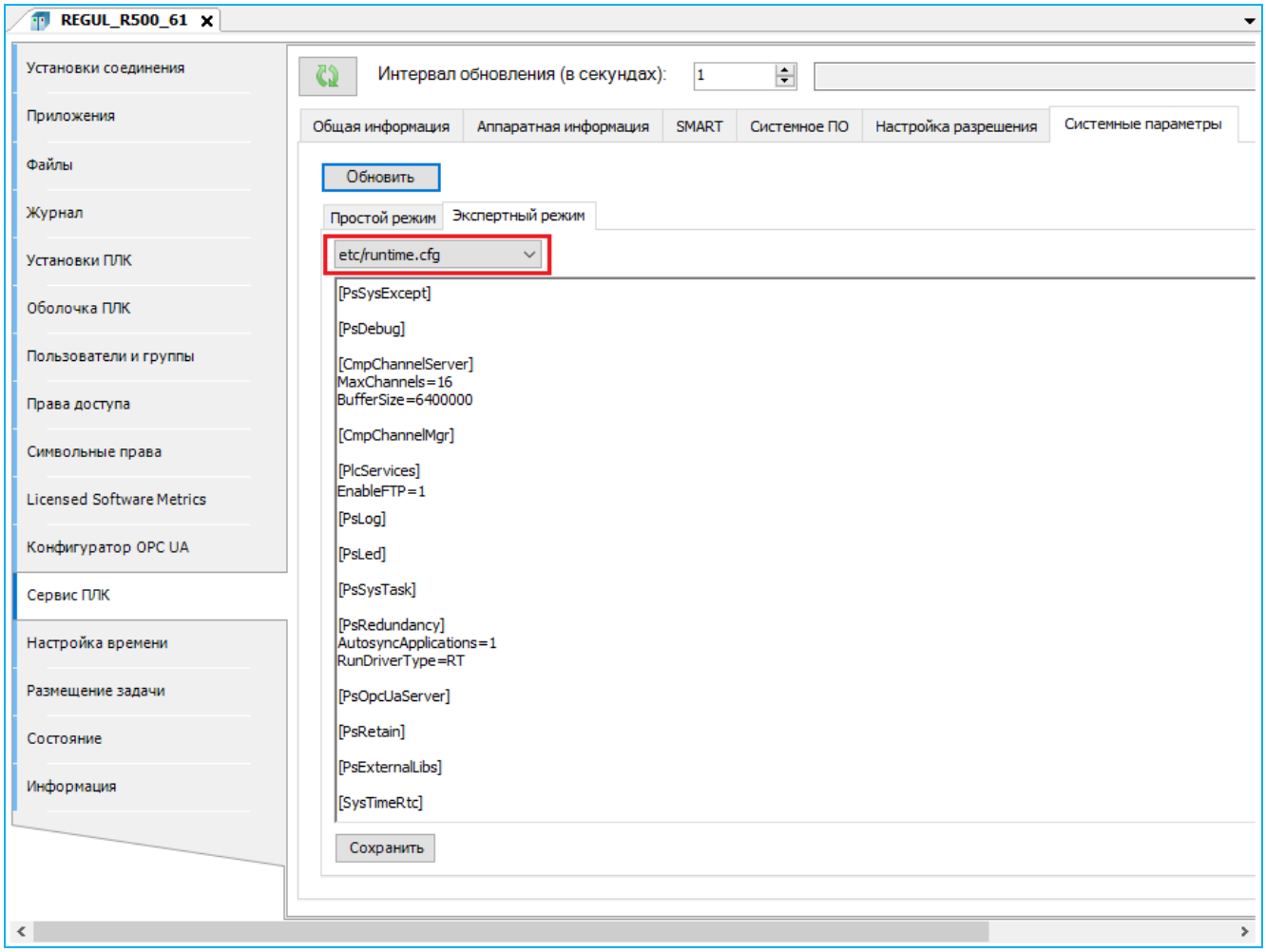


Рисунок 199 – Пример параметров в экспертном режиме для runtime.cfg

Для включения параметров, необходимо добавить строку в соответствующую секцию со значением «1». Так, например, если подключение по FTP протоколу выключено, то для включения необходимо добавить параметр *EnableFTP* равный 1. После внесения всех изменений нажмите кнопку **Сохранить.**

При внесении изменений в одном из двух режимов, для актуализации изменений в другом режиме, нажмите кнопку **Обновить**.

Настройка FTP

Конфигурирование учетных записей

Для разграничения доступа к файловой системе через FTP-сервер, добавления новых каталогов и их редактирования необходимы учетные записи. Их конфигурирование производится через файл ***ftp.userlist.*** По умолчанию файл содержит две учетные записи, которые могут быть удалены или редактированы:

- **plclogs** – доступ к каталогу лог-файлов;
- **plcuser** – доступ к пользовательскому каталогу.

Для работы с конфигурационным файлом ***ftp.userlist*** выполните следующие действия:

- на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc** (Рисунок 200);

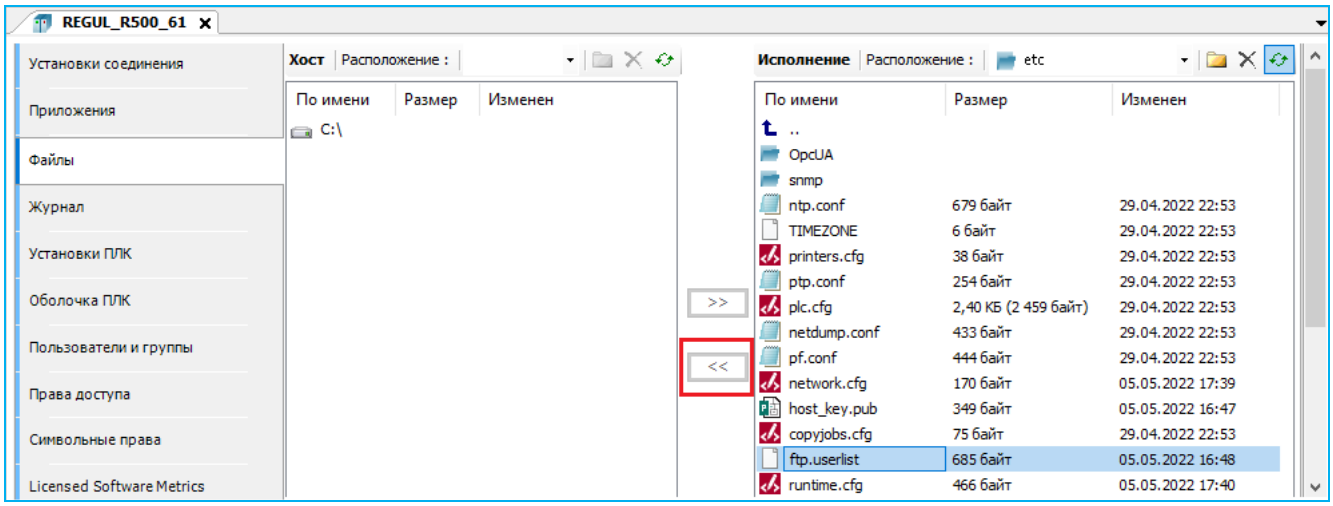


Рисунок 200 – Файл для конфигурирования учетных записей, расположенный в папке etc на контроллере

- кнопкой << скопируйте файл ***ftp.userlist*** с контроллера на ПК (из **Исполнение** в **Хост**);
- откройте на ПК файл ***ftp.userlist***;
- для добавления учетной записи создайте новую секцию *User.N*, где N - это порядковый номер пользователя. Задайте значения строк, руководствуясь приведенным описанием параметров в таблице 9. Список доступных параметров для конфигурирования описан в таблице 10;
- сохраните изменения в файле ***ftp.userlist***;
- в Astra.IDE на вкладке **Файлы** кнопкой >> скопируйте измененный файл с ПК на контроллер (из **Хост** в **Исполнение**).

Таблица 9 – Основные параметры файла ftp.userlist

Параметр	Описание
Name	Имя учетной записи
HomeDir	Домашняя директория. Будет доступна для данной учетной записи по умолчанию (при отсутствии параметра будет задан корневой каталог)
Access	Предоставление прав доступа учетной записи домашней директории (не распространяется на директори db/, logger/, stats/ в директории logs/ - они доступны только для чтения). Принимает значения: – ReadWrite – предоставить право на чтение и запись; – ReadOnly – предоставить право только на чтение. Если введено некорректное значение или оставили значение пустым, по умолчанию устанавливается *ReadWrite*. В случае удаления директории /logs/runtime/, она создастся заново из-под root(0) с правом доступа только на чтение
Password	Пароль учетной записи

Таблица 10 – Список доступных параметров для конфигурирования

Параметр	Описание
DownloadBandwidth	Максимальная скорость выгрузки данных из ftp-сервера (в kb/s)
UploadBandwidth	Максимальная скорость загрузки данных на ftp-сервер (в kb/s)
AllowedClientIPs	Список разрешенных ip-адресов или подсетей (доменов). IP-адреса могут быть разделены запятой
DeniedClientIPs	Список запрещенных ip-адресов или подсетей (доменов)
TimeRestrictions	Время суток, при котором разрешены подключения ftp-клиентов

Пример оформления файла ***ftp.userlist***:

```
[User.0]
Name=plclogs
HomeDir=/logs
Access=ReadOnly
Password=

[User.1]
Name=plcuser
HomeDir=/
Access=ReadWrite
Password=

[User.2]
Name=morty
HomeDir=/etc
Access=ReadOnly
Password=password
DownloadBandwidth=256
UploadBandwidth=512
AllowedClientIPs="192.168.0.10,172.29.23.0/24"
```

```
DeniedClientIPs="192.168.0.0/24"
TimeRestrictions="0900-1800"
```

После сохранения ftp.userlist, в течении 10-20 секунд будут созданы учетные записи.

Просмотр информации об учетных записях и операциях FTP сервера

Информация о создании и изменении учетных записей записывается в лог-файл ***ftp.users.log***. Запись информации об операциях FTP сервера производится в лог-файл ***syslog.log***. Для просмотра информации загрузите файлы на ПК.

Для загрузки файлов выполните следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Перейдите в папку **logs** ⇨**logger**⇨**user** (Рисунок 201).

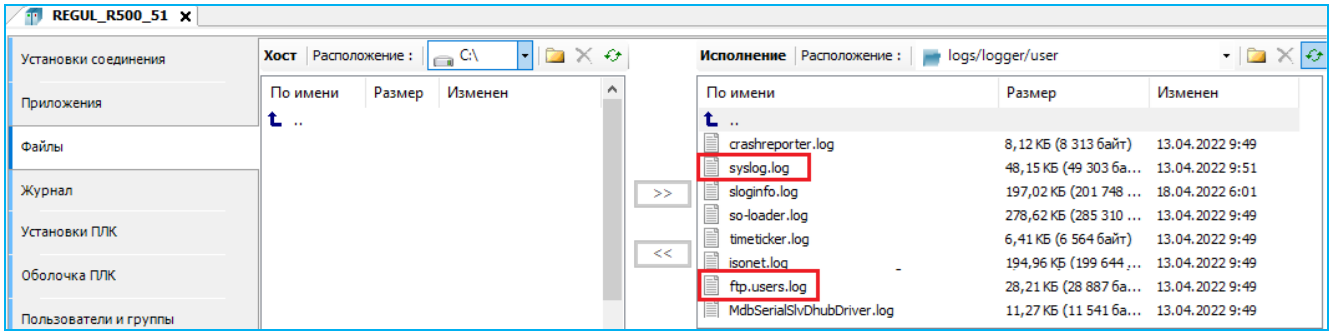


Рисунок 201 – Расположение лог-файлов на контроллере

– кнопкой << скопируйте лог-файлы ***ftp.users.log*** и ***syslog.log*** с контроллера на ПК (из **Исполнение** в **Хост**). Далее откройте их на ПК.

В файл ***ftp.users.log*** производится запись следующих параметров (Рисунок 202):

– список текущих пользователей;

– список новых пользователей;

– добавление/удаление учетных записей;

– изменение пароля;

– значение установленных параметров для каждого пользователя.

```
20.11.2019 11:09:33.111 ================================================
20.11.2019 11:09:33.122 List of current users: plclogs plcuser
20.11.2019 11:09:33.148 List of users from ftp.userlist: plclogs plcuser
20.11.2019 11:09:33.161 --------- User.0 ----------
20.11.2019 11:09:33.287 Login              : plclogs
20.11.2019 11:09:33.297 Directory          : //logs/./
20.11.2019 11:09:33.304 Download bandwidth : 0 Kb (unlimited)
20.11.2019 11:09:33.314 Upload   bandwidth : 0 Kb (unlimited)
20.11.2019 11:09:33.323 Allowed client IPs :
20.11.2019 11:09:33.332 Denied  client IPs :
20.11.2019 11:09:33.342 Time restrictions  : 0000-0000 (unlimited)
20.11.2019 11:09:33.347 --------- User.1 ----------
20.11.2019 11:09:33.436 Login              : plcuser
20.11.2019 11:09:33.448 Directory          : /./
20.11.2019 11:09:33.455 Download bandwidth : 512 Kb (enabled)
20.11.2019 11:09:33.463 Upload   bandwidth : 128 Kb (enabled)
20.11.2019 11:09:33.471 Allowed client IPs : 172.29.23.0/24
20.11.2019 11:09:33.480 Denied  client IPs :
20.11.2019 11:09:33.488 Time restrictions  : 0900-1800 (enabled)
```

Рисунок 202 – Запись параметров в файл ftp.users.log

В файл ***syslog.log*** производится запись по следующим операциям (Рисунок 203):

– логин/логаут пользователей;

– загрузка/выгрузка/удаление файлов.

```
(?@172.29.34.20) [INFO]  New connection from 172.29.34.20
(?@172.29.34.20) [INFO]  TLS: Enabled TLSv1.2 with ECDHE-RSA-AES256-GCM-SHA384, 256 secret bits cipher
(?@172.29.34.20) [INFO]  plcuser is now logged in
(plcuser@172.29.34.20) [INFO]  TLS: Enabled TLSv1.2 with ECDHE-RSA-AES256-GCM-SHA384, 256 secret bits cipher
(plcuser@172.29.34.20) [NOTICE]  /mnt/user/codesys//logs/runtime/test.txt uploaded  (0 bytes, 0.00KB/sec)
(plcuser@172.29.34.20) [INFO]  TLS: Enabled TLSv1.2 with ECDHE-RSA-AES256-GCM-SHA384, 256 secret bits cipher
(plcuser@172.29.34.20) [INFO]  TLS: Enabled TLSv1.2 with ECDHE-RSA-AES256-GCM-SHA384, 256 secret bits cipher
(plcuser@172.29.34.20) [NOTICE]  /mnt/user/codesys//logs/runtime/SysTaskStatistic.log downloaded  (1352 bytes, 1320.41KB/sec)
(plcuser@172.29.34.20) [NOTICE]  Deleted /mnt/user/codesys//logs/runtime/test2.txt
(plcuser@172.29.34.20) [INFO]  Logout.
```

Рисунок 203 – Запись в файл syslog.log о производимых операциях FTP сервера

Защищенное TLS соединение

Сервер FTP поддерживает защищенное TLS соединение, защищённую передачу данных между узлами в сети. Для включения TLS соединения необходимо выполнить следующие действия (Рисунок 204):

– перейдите на вкладку **Сервис ПЛК ⇨Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку  (*Обновить*);

– добавьте в секцию FTP конфигурационного файла *etc/plc.cfg* параметр *TLS* равным 0, 1 или 2 (описание приведено в таблице 11);

– нажмите кнопку **Сохранить**.

Таблица 11 – Значения параметра TLS

Параметр	Описание
0	SSL/TLS выключен
1	SSL/TLS включен, но при этом доступно соединение без шифрования

Параметр	Описание
2	SSL/TLS включен

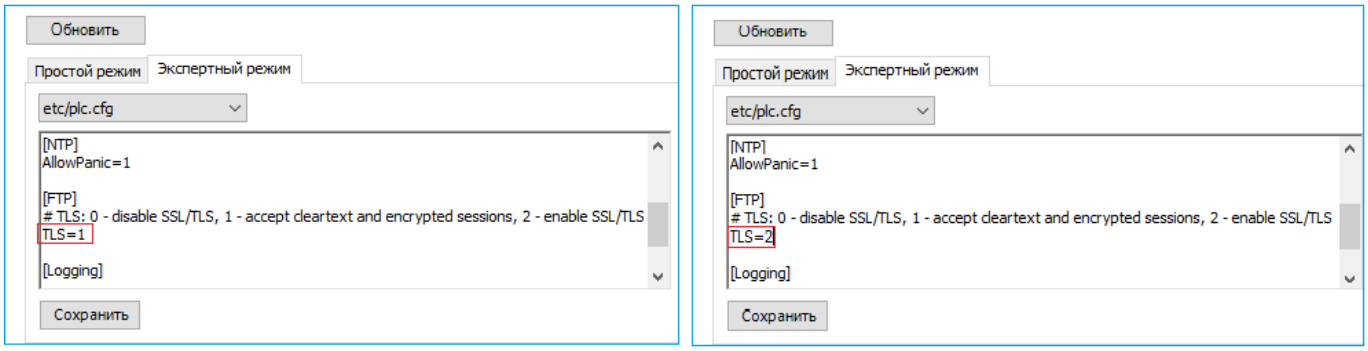


Рисунок 204 – Включение соединения *TLS* в конфигурационном файле plc.cfg

ИНФОРМАЦИЯ

В случае возникновения ошибок при подключении, убедитесь, что сертификат не просрочен. Для этого подключитесь к контроллеру и выберите в главной вкладке параметров устройства внутреннюю вкладку **Оболочка ПЛК.** В командной строке введите команду cert-getapplist. Появится список всех применяемых сертификатов. Если для компонента CmpSecureChannel сертификат просрочен, то в командной строке введите команду cert-genselfsigned * (* - укажите порядковый номер компонента в списке). Спустя несколько секунд снова введите команду cert-getapplist и убедитесь в создании нового сертификата

Изменение диапазона портов, используемых FTP сервером в пассивном режиме

Сервер FTP может работать как в активном, так и в пассивном режиме работы. В пассивном режиме подключение управляющего соединения, как и соединение для передачи данных, инициируется клиентом. При включении пассивного режима сервер открывает порт для передачи данных, сообщает его клиенту и ожидает подключения.

Для задания диапазона портов при работе FTP сервера в пассивном режиме выполните следующие действия:

– перейдите на вкладку **Сервис ПЛК ⇨Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку  (*Обновить*);

– добавьте в секцию FTP конфигурационного файла *etc/plc.cfg* параметр *PassivePortRange*, указав значение строки, руководствуясь следующим:

```
PassivePortRange = <Начальный порт> <Конечный порт>,
```

где <**Начальный порт**> - номер порта начала диапазона;

<**Конечный порт**> - номер порта конца диапазона;

При условии, что значение диапазона находится в разрешенных границах:

1024 ≤ <**Начальный порт**> ≤ <**Конечный порт**> ≤ 65534

– нажмите кнопку **Сохранить**.

Журналирование изменений и некорректное задание диапазона портов производится в лог-файл system.log (расположен в директории /logs/logger).

Главное отличие активного и пассивного режимов работы протокола FTP состоит в том, кто из связки клиент-сервер производит подключение для передачи данных, то есть, грубо говоря, кто к кому подключается. Также отличаются порты, на которые производится передача данных. При активном режиме работы, клиент производит управляющее соединение с сервером, а вот подключение для передачи данных производит уже сам сервер. При пассивном режиме работы подключение для передачи данных, равно как и управляющее соединение с сервером инициируется только клиентом. То есть, в активном режиме сервер подключается к клиенту для передачи данных, а в пассивном – клиент к серверу.

Запуск службы SNMP-сервера

Версия SNMP v3 с USM

В контроллерах реализована поддержка SNMP-сервера (агента), принимающего сообщения событий от устройств по протоколу SNMP. Для включения службы SNMP агента необходимо выполнить следующие действия:

– перейдите на вкладку **Сервис ПЛК ⇨ Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку  (*Обновить*);

– добавьте в секцию [PlcServices] конфигурационного файла *etc/runtime.cfg* параметр *EnableSNMP* равный 1 для включения:

```
[PlcServices] EnableSNMP=1;
```

– нажмите кнопку **Сохранить**.

Основные параметры необходимые для агента SNMP представлены в таблице 12.

Таблица 12 – Основные параметры настройки

Параметр	Описание
Сетевая настройка агента SNMP	Входящие подключения от менеджеров SNMP (клиентов) принимаются и обрабатываются на стандартных портах tcp:161 и udp:161
Версия протокола SNMP	Для взаимодействия с внешними клиентами SNMP поддерживается только версия **SNMPv3 с USM** (User-based Security Model)

Параметр	Описание
Аутентификация	Параметры безопасности, используемые для подключения: – имя пользователя – Administrator; – протокол аутентификации – SHA; – пароль аутентификации – Administrator; – тип шифрования – AES (AES-128); – пароль для шифрования – Administrator

ИНФОРМАЦИЯ

Режим TSM (Transport Security Model с поддержкой протоколов TLS и DTLS) не поддерживается!

Настройки на примере вкладки менеджера - PowerSNMP Free Manager (Рисунок 205).

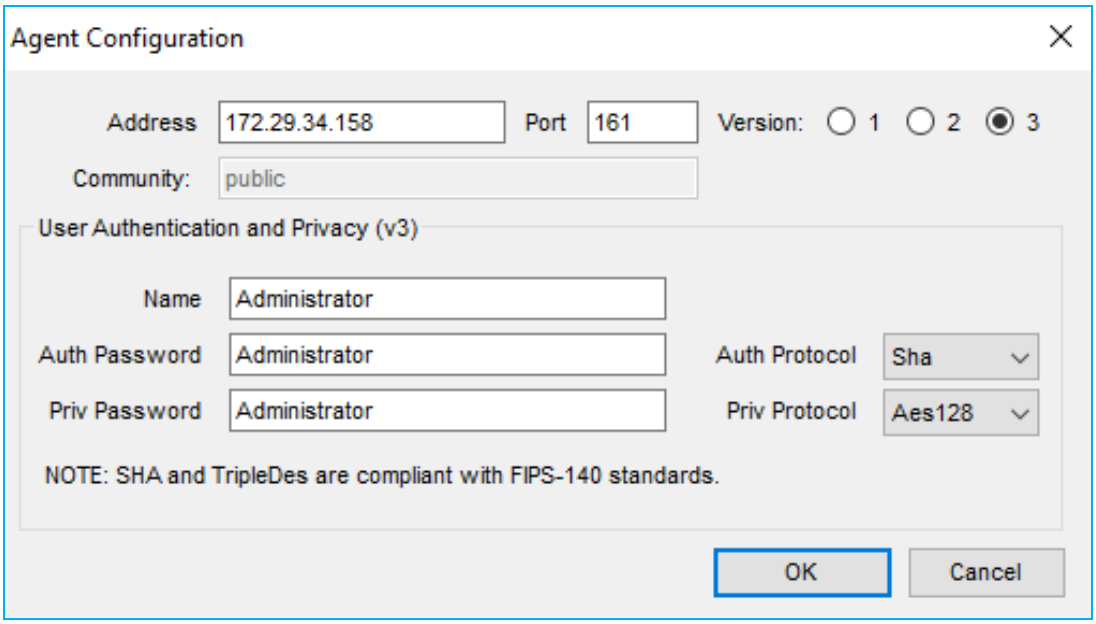


Рисунок 205– Пример вкладки с заданными параметрами

В каталоге ***/etc/snmp/mibs*** на контроллере хранится список стандартных поддерживаемых MIB-файлов, которые можно скопировать и импортировать в менеджер **SNMP**.

Для загрузки MIB-файлов выполните следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc⇨snmp⇨mibs** (Рисунок 206);

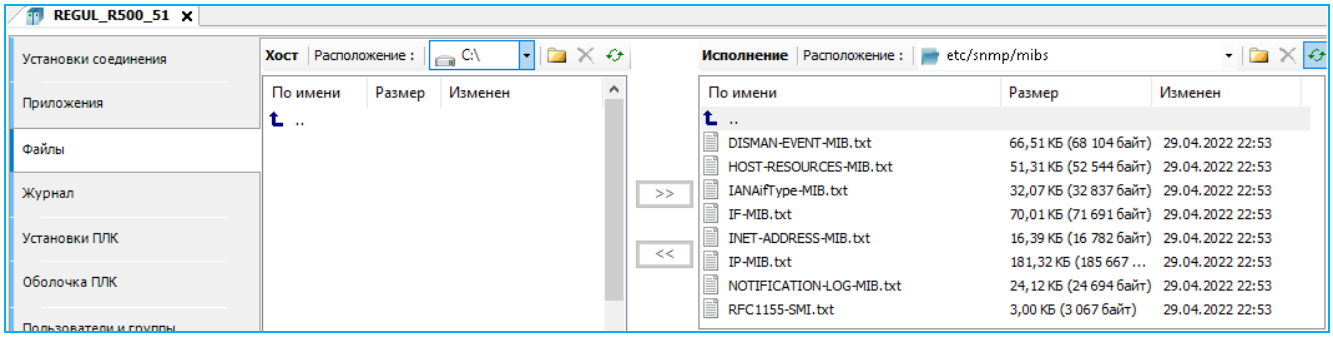


Рисунок 206 – Загрузка файла с контроллера на ПК

– кнопкой << скопируйте файл **mibs** с контроллера на ПК (из **Исполнение** в **Хост**), далее загрузите его с ПК в используемый вами менеджер SNMP.

К указанным выше спискам параметров предоставляется доступ только на чтение.

Версия SNMP v1

По умолчанию, работа SNMP агента по протоколу версии SNMPv1, как небезопасного, отключена. Для включения службы SNMP агента необходимо выполнить следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc⇨snmp (**Рисунок 207**)**;

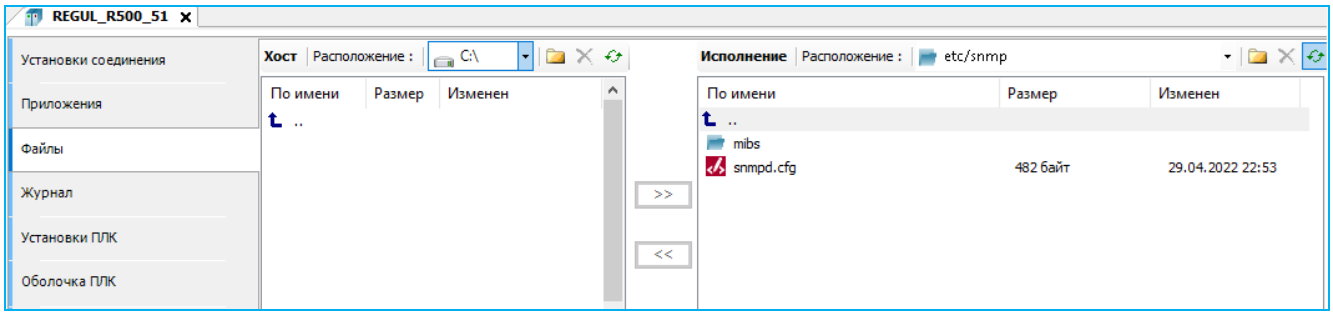


Рисунок 207 – Файл для конфигурирования SNMP агента по версии протокола SNMPv1, расположенный в папке snmp на контроллере

– кнопкой << скопируйте файл ***snmpd.cfg*** с контроллера на ПК (из **Исполнение** в **Хост**);

– откройте на ПК файл ***snmpd.cfg***. Содержимое файла представляет собой инструкцию по добавлению имен сообществ (community name) с доступом на чтение или запись. Задайте значения строк, руководствуясь приведенным описанием параметров в таблице 13;

```
rocommunity COMMUNITY SOURCE
rwcommunity COMMUNITY SOURCE
```

Таблица 13– Основные параметры файла snmpd.cfg

Параметр	Описание
rocommunity	Директива создания имен сообществ на чтение
rwcommunity	Директива создания имен сообществ на запись

Параметр	Описание
COMMUNITY	Имя сообщества – произвольная строка Например: public, my_comm_name
SOURCE	IP-фильтр входящих подключений от SNMP-клиентов. Может быть задан как диапазон разрешенных IP-адресов или ключевым словом “default” (разрешено всем). Ограничивающий фильтр можно указать как имя (адрес) хоста (localhost и пр.), либо в виде подсети IP/MASK (10.10.10.0/255.255.255.0) или IP/BITS (10.10.10.0/24).
Примеры: **1.** создание имени сообщества public с доступом на чтение без ограничений по IP: rocommunity public default **2.** сообщество regul_private_1Hkso9763 с доступом на запись для клиентов из подсети: rwcommunity regul_private_1Hkso9763 172.29.34.0/24	

– сохраните изменения в файле ***snmpd.cfg***;

– в Astra.IDE на вкладке **Файлы** кнопкой  скопируйте измененный файл с ПК на контроллер (из **Хост** в **Исполнение**).

Добавленные директивы создания имен сообществ автоматически применятся без перезагрузки контроллера.

	**ИНФОРМАЦИЯ** Для подключения имеющимся у пользователя менеджером (клиентом) SNMP по протоколу версии SNMPv1 к ПЛК Regul необходимо будет в настройках подключения указать адрес ПЛК в сети и имя сообщества

Запись данных на внешний накопитель

Контроллер поддерживает функцию записи пользовательских данных на внешний USB-накопитель либо MMC/SD-карту (далее по тексту – USB, MMC/SD).

По умолчанию механизм автомонтирования внешних накопителей USB, MMC/SD выключен. Для включения перейдите на вкладку **Сервис ПЛК ⇨ Системные параметры** (см. подраздел «Настройка системных параметров»). Нажмите кнопку  (***Обновить***). Выберите вкладку: **Простой режим** или **Экспертный режим**. Активируйте одним из удобных для вас способов:

– на вкладке **Простой режим** установите флажок ☑ в поле **Разрешение на подключение внешних накопителей**;

– либо, на вкладке **Экспертный режим** в секции PlcServices конфигурационного файла */etc/runtime.cfg* добавьте параметр *AutomountStorage* равный 1:

```
[PlcServices] AutomountStorage=1
```

Нажмите на кнопку **Сохранить**. В обоих случаях, для вступления в силу изменений, перезагрузите контроллер путем выключения/включения питания, либо командой *reboot* на вкладке **Оболочка ПЛК**.

Сценарии копирования пользовательских данных

Для пользовательских данных опишите сценарий копирования в конфигурационном файле ***copyjobs.cfg.*** Для работы с файлом выполните следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc** (Рисунок 208);

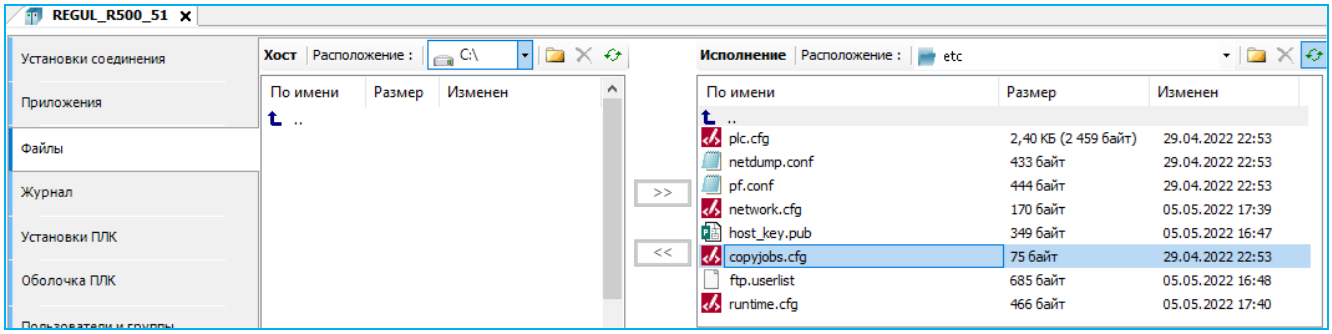


Рисунок 208 – Конфигурационный файл расположенный в папке etc на контроллере

– кнопкой << скопируйте файл ***copyjobs.cfg*** с контроллера на ПК (из **Исполнение** в **Хост**);

– откройте на ПК файл ***copyjobs.cfg*** (Рисунок 209). Задайте значения строк, руководствуясь приведенным описанием параметров в приложении Г;

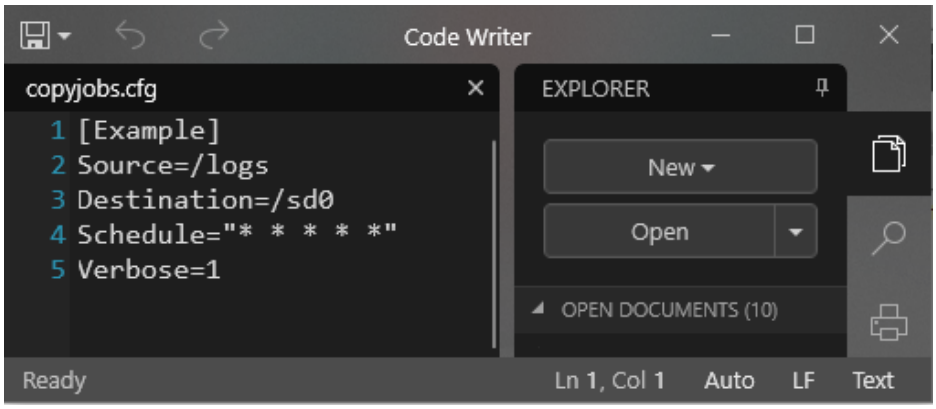


Рисунок 209 – Конфигурационный файл copyjobs.cfg

– в Astra.IDE на вкладке **Файлы** кнопкой >> скопируйте измененный файл с ПК на контроллер (из **Хост** в **Исполнение**).

Для вступления в силу изменений потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Альтернативный вариант настройки - перейти на вкладку **Сервис ПЛК ⇨Системные параметры⇨Экспертный режим** и выбрать для редактирования конфигурационный файл ***copyjobs.cfg***.

Диагностика контроллера

Получение диагностической информации о контроллере

В программе Astra.IDE в дереве устройств поместите курсор на название контроллера. Двойной щелчок левой кнопкой мыши открывает главную вкладку (окно) параметров устройства. Запустите контроллер. Установите соединение контроллера с программой Astra.IDE (см. раздел «Подключение контроллера к сети»).

Для получения информации о состоянии контроллера в реальном времени перейдите на внутреннюю вкладку **Сервис ПЛК**. В поле **Интервал обновления (с):** задайте интервал обновления информации, далее нажмите кнопку  (***Обновить***). Начнется диагностика контроллера (Рисунки 210, 211).

На вкладке **Общая информация** при выборе режима *сеть и хранение данных* отображается номер версии системного ПО контроллера, название и версия среды исполнения. Сведения по сетевым интерфейсам и хранению данных позволяют оценить возможности использования ресурсов операционной системы по хранению и накоплению данных, а также и их передачу по сети.

В блоке *Сетевые интерфейсы* приведена информация по каналам соединения с ПЛК (сконфигурирован интерфейс, подключен порт), статистика по входящим и исходящим пакетам/байтам на канале, тут же регистрируется количество ошибок соединения в счетчиках FCS (ошибки контрольной суммы), количество ошибок выравнивания. Аналогичная статистика и количество ошибок приводятся по работе внутренней шины данных (интерфейс ecatbus0 – для модулей I/II типа).

В блоке *Хранение данных* приведена информация по общему и занятому объему памяти в соответствующих разделах:

- ***system*** – системные данные и системное ПО;
- ***archive*** – файлы журналов, резервных копий и базы данных;
- ***app*** – пользовательское приложение и файлы.

В режиме *загрузка ПЛК* отображаются графики загрузки центрального процессора и оперативной памяти. Если в контроллере присутствует два процессора, то будет показана информация по каждому из них (Рисунок 212).

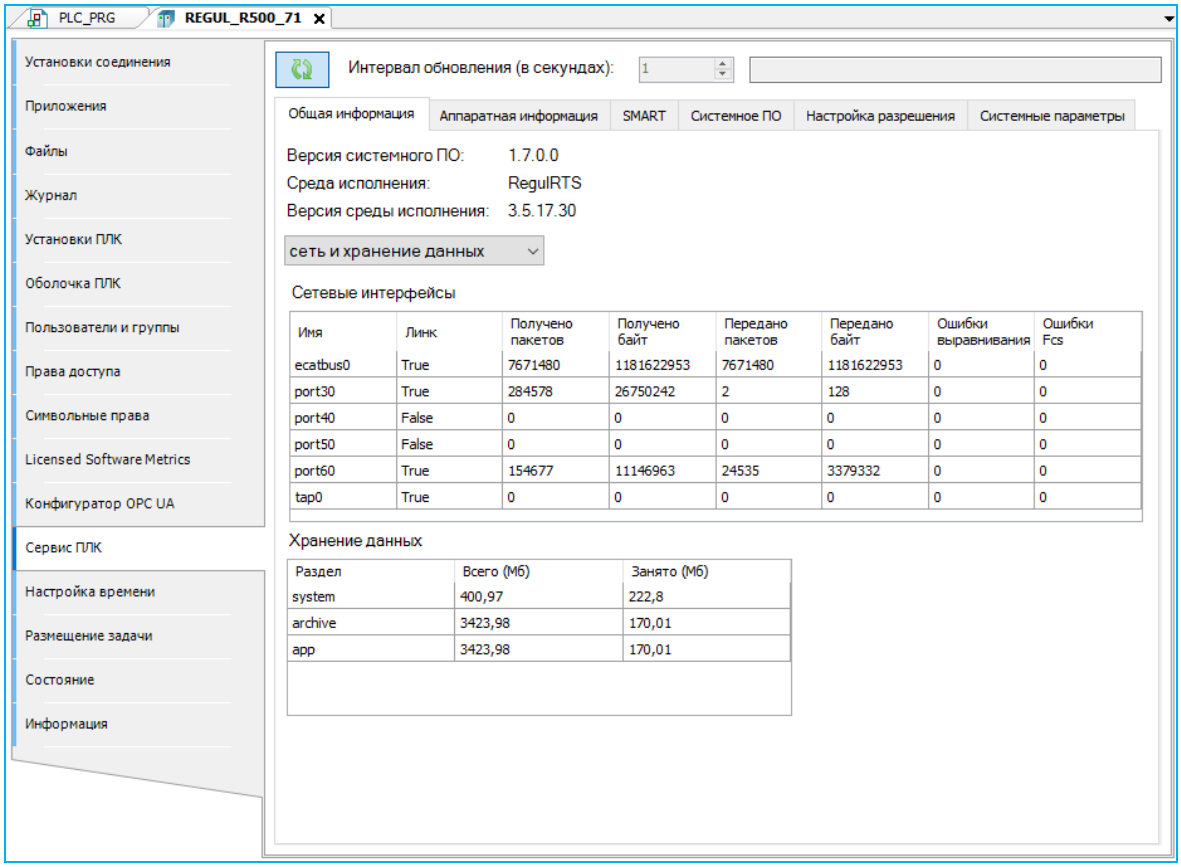


Рисунок 210 – Общая диагностическая информация о контроллере с модулем ЦП I/II-го типа

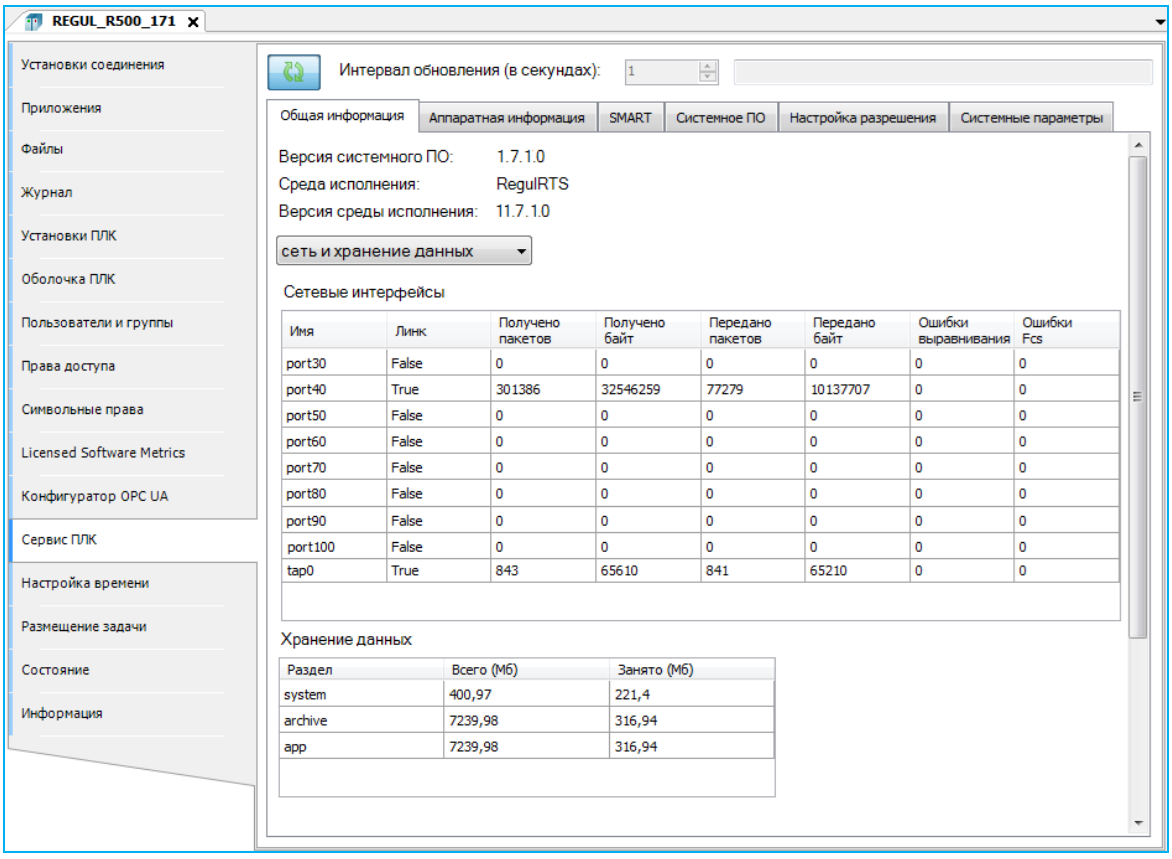


Рисунок 211 – Общая диагностическая информация о контроллере с модулем ЦП III-го типа и одним блоком расширения

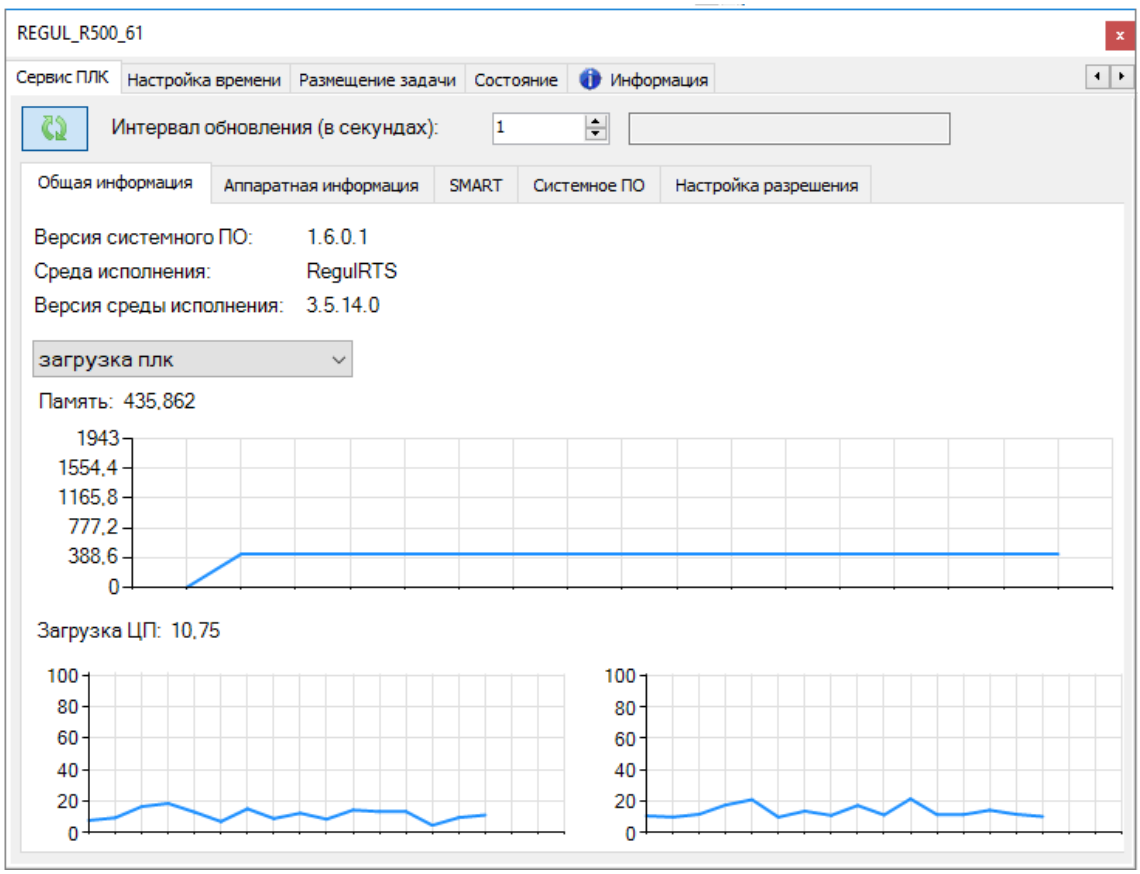


Рисунок 212 – Пример данных по загрузке ПЛК

Помимо вывода загрузки процессоров на график, возможно настроить журналирование загрузки процессора в лог-файл *cpu.log*, который расположен в директории /logs/stats/sysinfo (см. «Приложение Д»).

Для этого нужно выполнить следующее (Рисунок 213):

– перейдите на вкладку **Сервис ПЛК ⇨Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку (*Обновить*);

– добавьте в секцию [Logging] конфигурационного файла *etc/plc.cfg* (см. «Приложение Е»). параметр *CpuLoad*, равный необходимому значению (в секундах), которое определяет период записи в лог-файл информации о загрузке процессора;

– нажмите кнопку **Сохранить**.

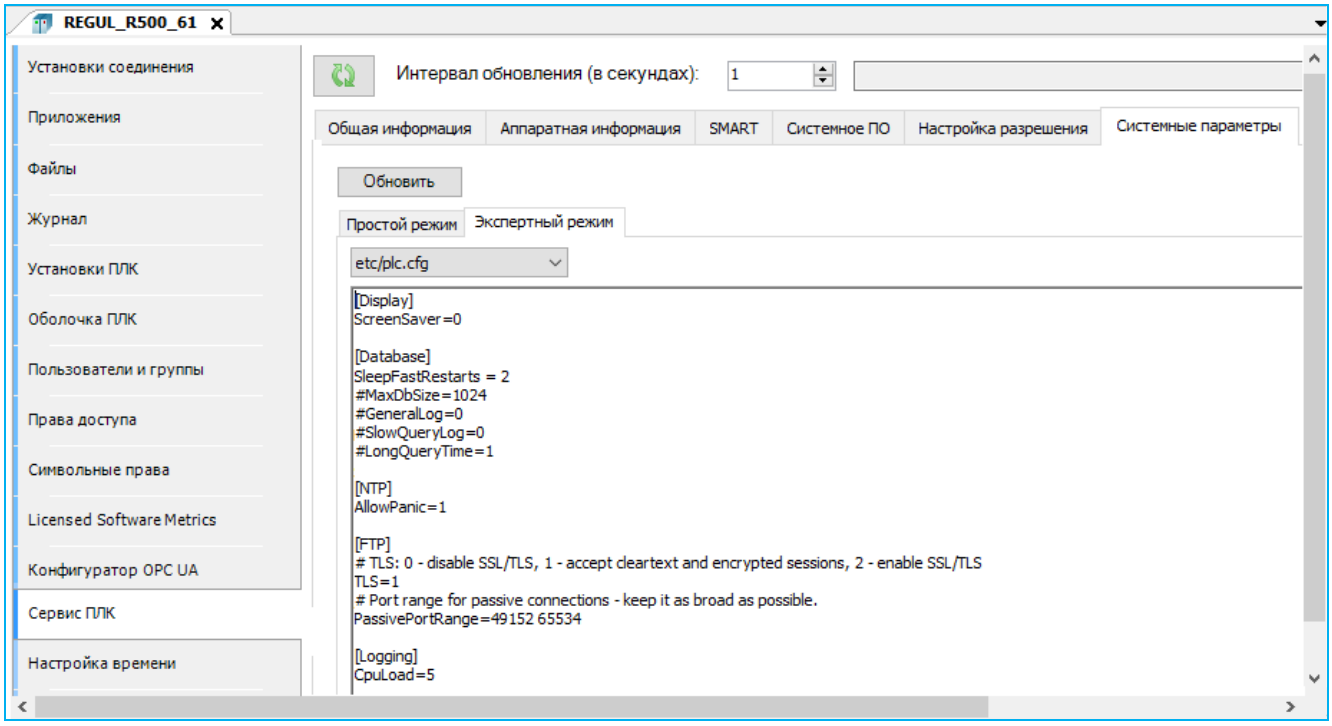


Рисунок 213 – Установка периода журналирования загрузки процессора

Аналогично можно настроить журналирование сведений по сетевым интерфейсам и хранению данных, добавив в секцию [Logging] параметры *Ram, Hdd, Network* (см. «Приложение Е»).

Для отключения журналирования присвойте соответствующему параметру значение 0. Если в секции [Logging] отсутствует запись параметра, то журналирование осуществляется по умолчанию. По умолчанию загрузка процессора (CpuLoad) журналируется каждую секунду, а остальные параметры (Ram, Hdd, Network) каждую минуту.

На вкладке **Аппаратная информация** (Рисунок 214) отображаются сведения:

- **температура процессора и платы.** Значения *Максимальная:* и *Минимальная:* относятся ко всему сроку службы контроллера, в том числе в период тестирования на предприятии-изготовителе;
- **общие параметры**:
  - обратный счетчик WDT и начальное значение WDT в секундах,
  - текущее время работы,
  - причина последнего перезапуска;
- **все параметры**. В табличном виде приведена текущая статистика по всем основным параметрам. Также, можно запросить информацию по каждому из параметров (ValName) с помощью функции **HwmonGetIntVaL** или всех параметров – **HwmonGetParamNamesList**, библиотеки **PsPlcInfo** (см. «Приложение К»).

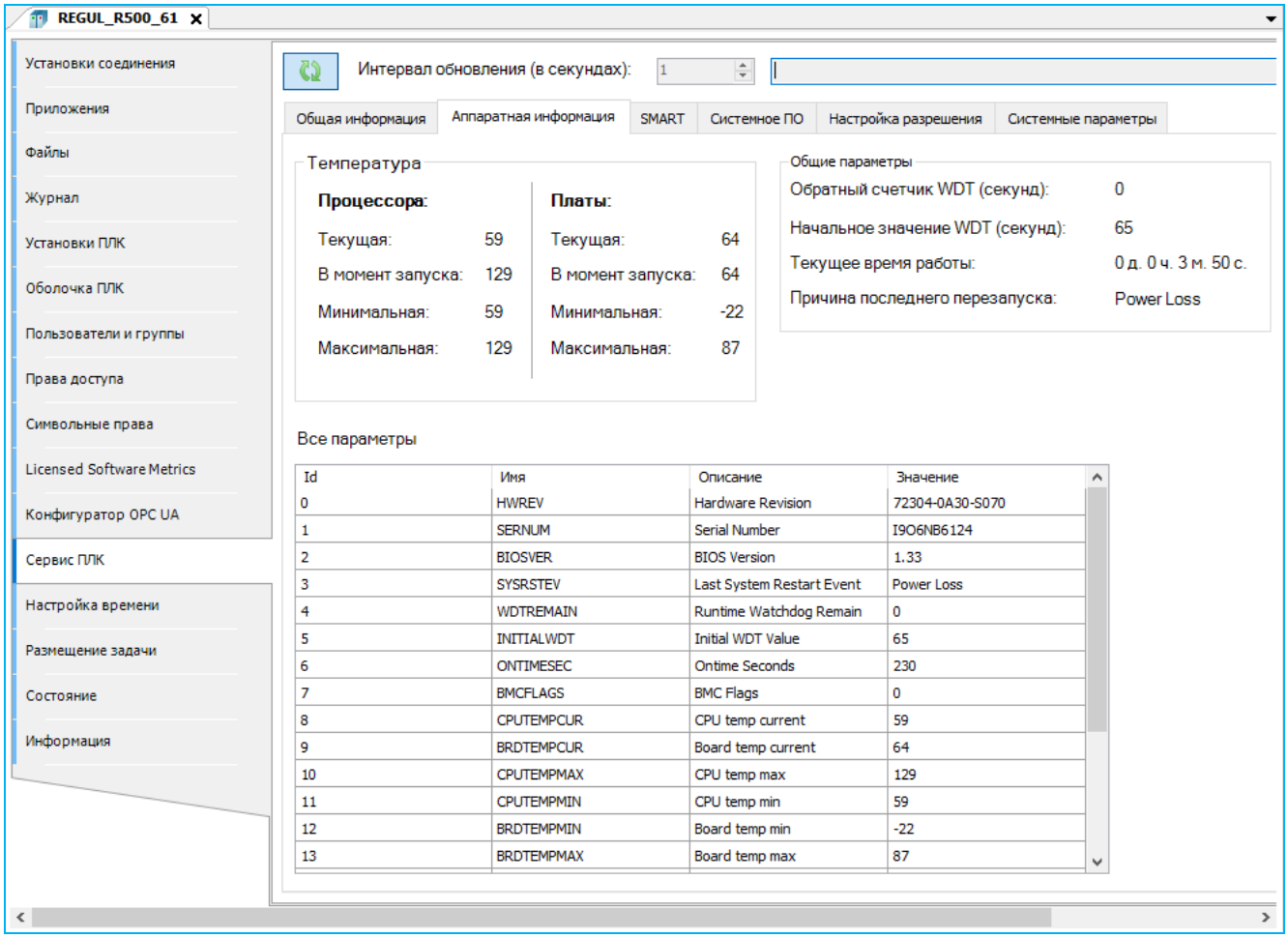


Рисунок 214 – Аппаратная информация

Вкладка **SMART** предназначена для просмотра данных о состоянии твердотельного накопителя контроллера. Для этого нажмите кнопку ***Запросить данные***.

В окне будут отображены все сведения о состоянии твердотельного накопителя контроллера (Рисунок 215). Если на контроллере установлено два накопителя, то есть возможность просматривать SMART-данные по каждому из них. Для этого выберите нужный накопитель в раскрывающемся списке, расположенном под кнопкой ***Запросить данные***.

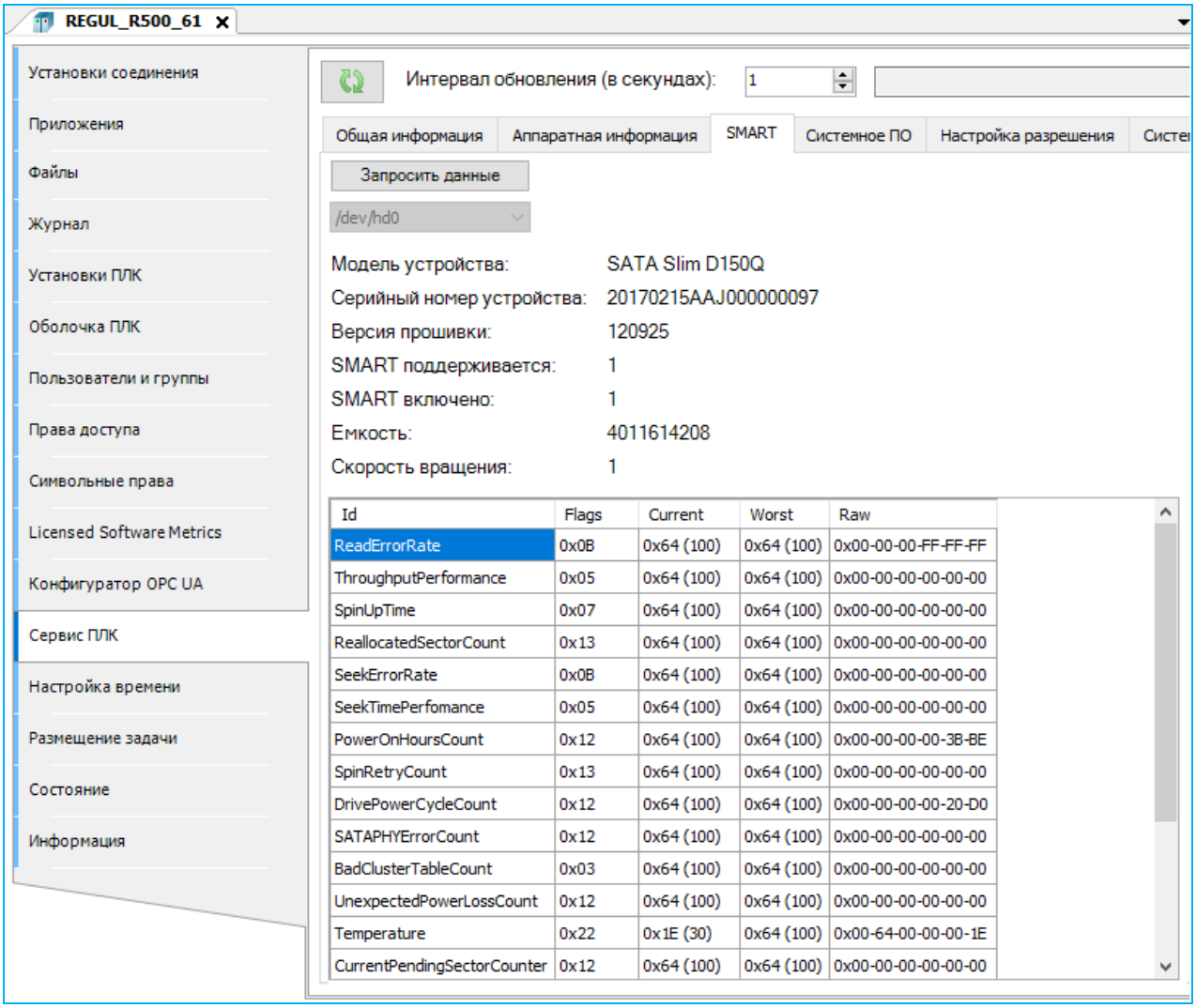


Рисунок 215 – SMART данные

Включение отладочного режима для крейтов и модулей

Необходимо помнить, что общий размер журнала событий (лог-файлов) ограничен. Скорость заполнения журнала событий зависит от количества включенных диагностических опций проекта. Не следует одновременно включать избыточное количество диагностических опций, так как это может привести к слишком быстрому циклу перезаписи лог-файлов. Правильным решением является включение лишь тех диагностических опций, которые действительно необходимы для поиска причин сбоя в конкретном модуле или драйвере. Все остальные диагностические опции должны быть отключены! В штатном режиме работы контроллера также желательно отключить все диагностические опции в целях уменьшения времени исполнения основного цикла программы.

В главном меню программы Astra.IDE выберите **Инструменты** ⇨ **Опции** и в открывшемся окне пункт **Редактор устройств**. Установите флажок в поле **Показывать общие окна конфигурации устройств** (Рисунок 216).

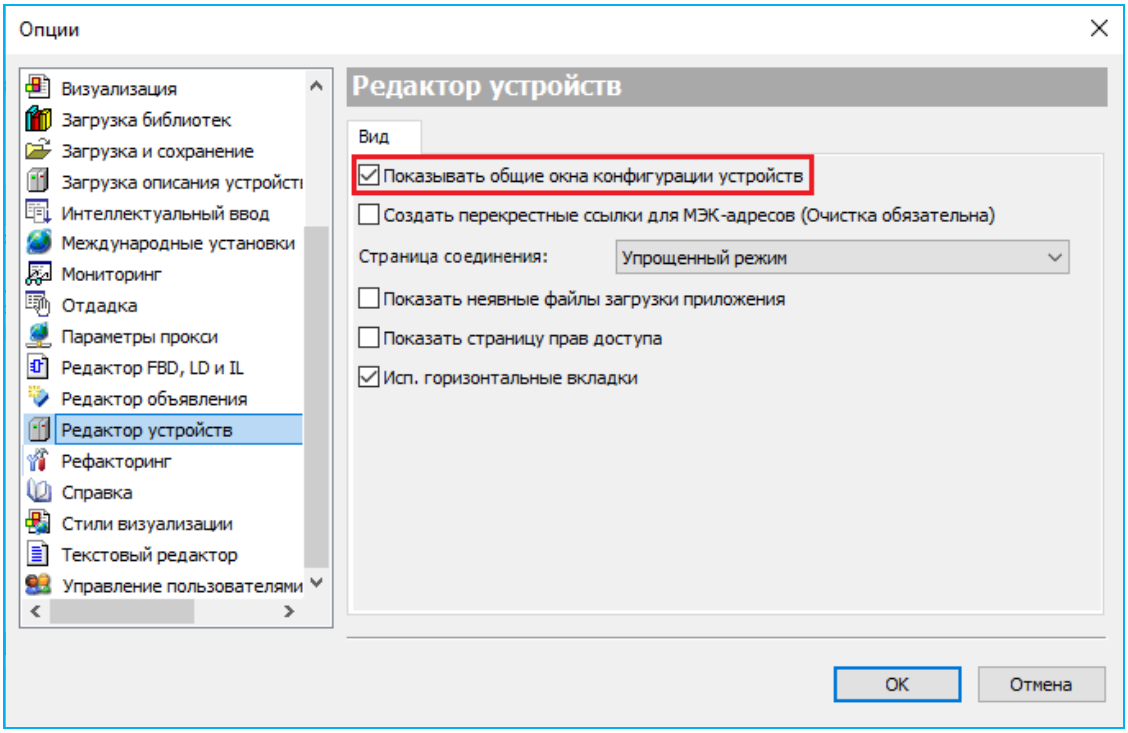


Рисунок 216 – Настройки редактора устройств

В окне устройств найдите элемент **RegulBus**, раскройте дерево устройств. Выберите нужный крейт, двойным щелчком левой кнопкой мыши откройте вкладку (окно) параметров крейта. Перейдите на внутреннюю вкладку **<Имя крейта> Конфигурация** (Рисунок 217).

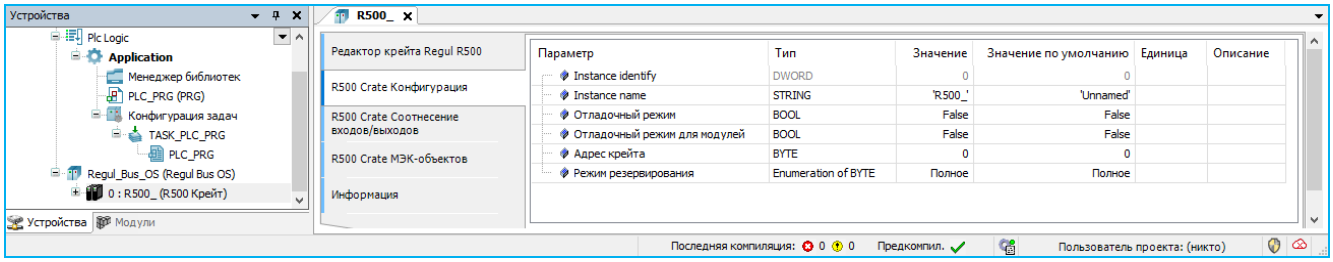


Рисунок 217 – Параметры крейта

При необходимости отладки всех модулей данного крейта найдите строки **Отладочный режим** и **Отладочный режим для модулей**. В ячейке **Значение** двойным щелчком левой кнопкой мыши измените значение с *FALSE* на *TRUE* (Рисунок 218).

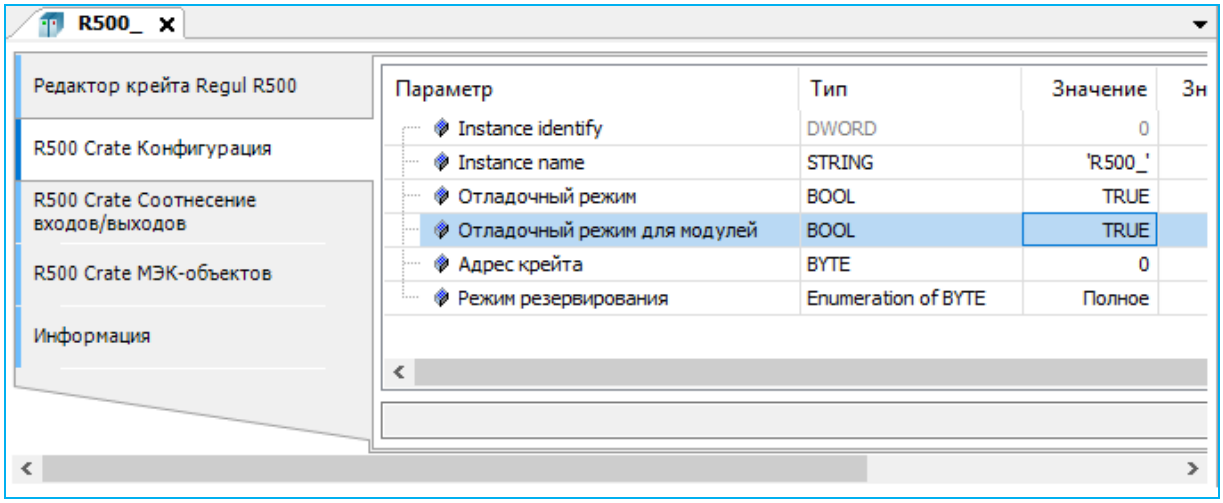


Рисунок 218 – Включение отладочного режима для всех модулей

Для отладки отдельного модуля выберите этот модуль в дереве устройств. Двойным щелчком левой кнопкой мыши откройте вкладку (окно) параметров модуля. Перейдите на внутреннюю вкладку **<Имя модуля> Конфигурация**. Найдите строку **Отладочный режим** и в ячейке **Значение** двойным щелчком левой кнопкой мыши измените значение с *FALSE* на *TRUE* (Рисунок 219).

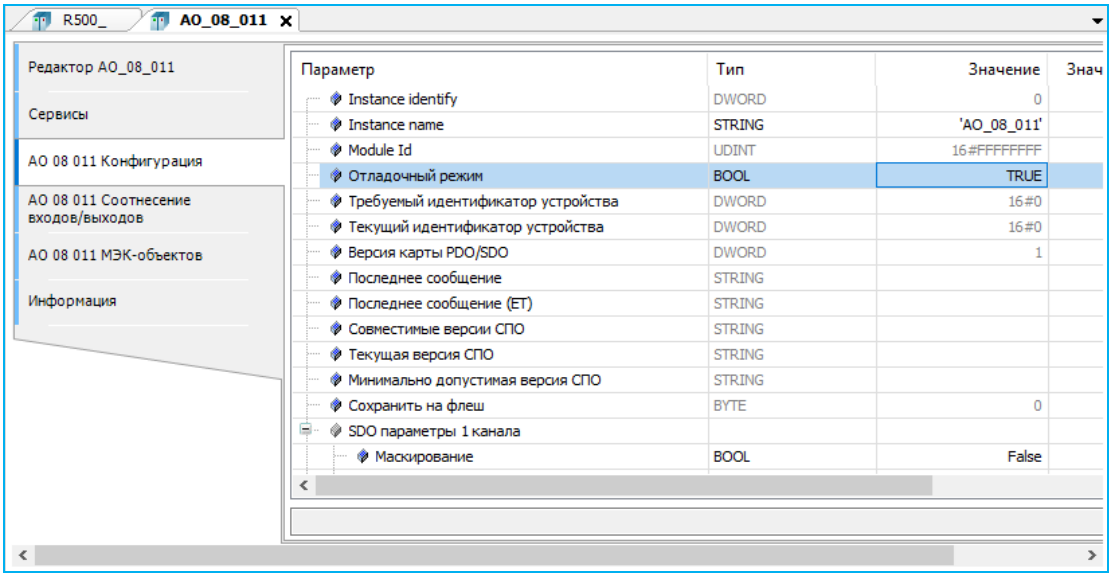


Рисунок 219 – Включение отладочного режима для модуля

Включение отладочного режима для драйверов Modbus, IEC-104/101, Foundation Fieldbus H1

В окне устройств раскройте дерево устройств, выберите необходимое устройство и на вкладке с общими параметрами установите флажок в строке **Отладочный режим**. Например:

для настройки **Modbus** выберите двойным щелчком левой кнопкой мыши следующее устройство:

– <X>: Regul_<X>\Modbus_Serial_Master\Modbus_Serial_Outer_Slave_<Y>,

– <X>: Regul_<X>\Modbus_Serial_Slave_<Y>,

– Modbus_TCP_Master\Modbus_TCP_Outer_Slave,

– Modbus_TCP_Slave_Device;

для настройки **МЭК-104** выберите двойным щелчком левой кнопкой мыши следующее устройство:

Master 104

– _OuterSlave_104_<X> (Драйвер OuterSlave 104),

Slave 104

– _Slave_104_<X> (Драйвер Slave 104);

для настройки **МЭК-101** выберите двойным щелчком левой кнопкой мыши следующее устройство:

Serial 60870-101 Slave

– <X>: Regul__Unbalanced_Secondary_101_<Y>,

Serial 60870-101 Master

– <X>: Regul__Unbalanced_Primary_Master_101_<X>_OuterSlave_101_<Y>;

для настройки **Foundation Fieldbus H1** выберите двойным щелчком левой кнопкой мыши следующее устройство:

– <X>: FF_H1_Link_Master_<Y>.

При установке флажка в поле **Отладочный режим** в блоке **Общие параметры устройства** (например, рисунок 220) в журнал работы контроллера будут выводиться сообщения об ошибках, предупреждениях и ключевых моментах работы драйвера.

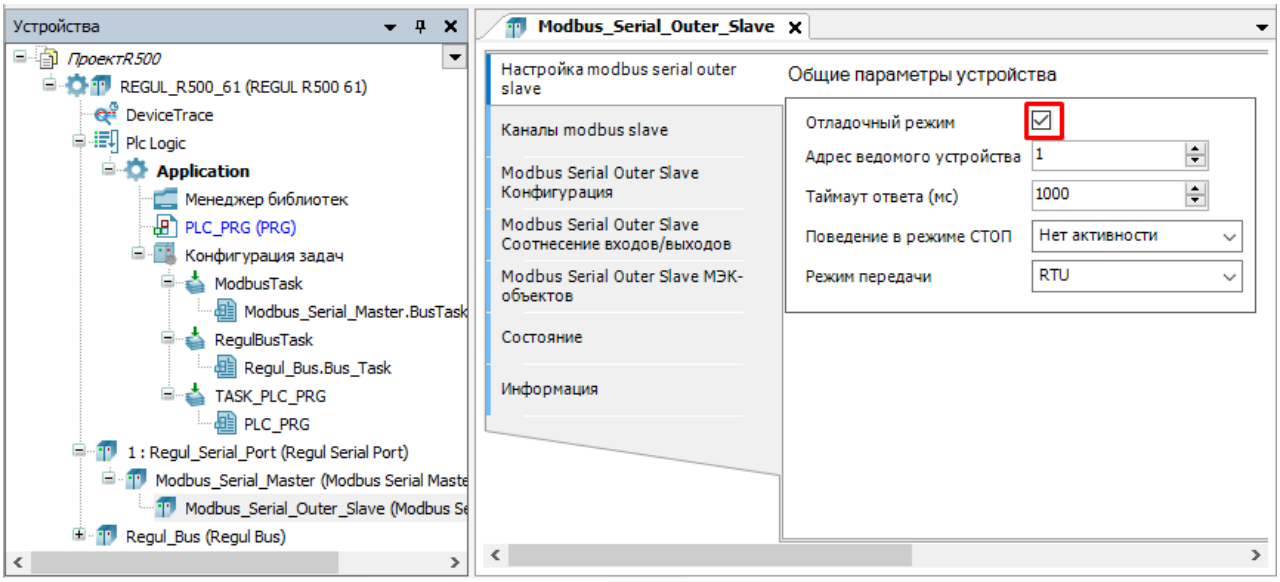


Рисунок 220 – Включение отладочного режима

Журнал событий

В программной среде Astra.IDE предусмотрены три журнала для просмотра событий, произошедших в системе исполнения (оперативный, полный и статистика по МЭК задачам). К таким событиям относятся:

– события при запуске и остановке системы (загруженные компоненты и их версии);
– загрузка приложения и загрузочного проекта;
– пользовательские записи;
– запись в журнале I/O-драйверов и запись в журнале сервера данных;
– статистика и добавление/удаление МЭК задач (SysTaskStatistic).

Журналы заполнены записями, и каждая запись относится к одной из категорий:

– : предупреждение;
– : ошибка;
– : исключение;
– : сообщение;
– : отладка.

Описание типовых событий в контроллере приведено в приложении Ж.

Присутствует возможность отправки сообщений журнала событий на удаленный сервер. Перейдите на вкладку **Сервис ПЛК ⇨Системные параметры** и нажмите кнопку (Обновить). Выберите вкладку **Экспертный режим⇨etc/plc.cfg**. Конфигурирование отправки сообщений на удаленный сервер осуществляется в секции [Syslog] конфигурационного файла etc/plc.cfg согласно описанию, приведенному в приложении Е.

Оперативный журнал событий

Оперативный журнал отображает все события от момента старта контроллера. Установите связь с контроллером. Перейдите на основной вкладке параметров устройства на вкладку **Журнал**. Нажмите кнопку (***Обновить***). Журнал заполнится записями (Рисунок 221).

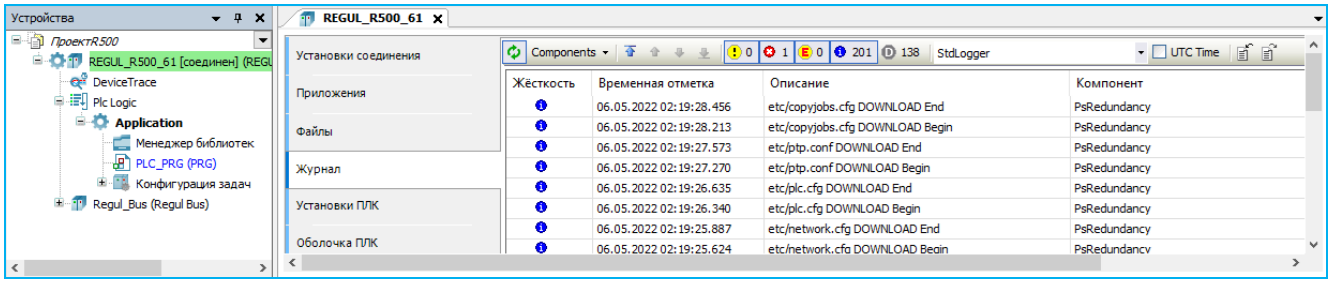


Рисунок 221 – Оперативный журнал событий (Stdlogger)

Отображение записей каждой категории можно включить или выключить с помощью соответствующих кнопок на панели сверху. На каждой кнопке показано количество записей соответствующей категории (Рисунок 222).

Рисунок 222 – Оперативный журнал событий

Для каждой записи журнала заданы следующие характеристики:

– **Временная отметка** – дата и время (с точностью до миллисекунд). Если установлен флажок в поле **UTC-время**, то в журнале отображается время runtime-системы контроллера. Если поле неактивно, то отображается локальное время компьютера (в соответствии с установленной временной зоной);

– **Описание** – описание события;

– **Компонент** – ID и имя компонента. Можно задать отображение только записей, касающихся конкретного компонента. По умолчанию выбрана опция *<Все компоненты>*.

Журнал имеет ограничение 5000 записей. При достижении этого порогового значения журналирование останавливается.

Для сохранения журнала нажмите на панели инструментов кнопку  (***Экспорт отображаемых элементов в XML-файл***) (Рисунок 223). Откроется диалоговое окно для выбора места сохранения файла. Файл будет сохранен с расширением *.xml*.

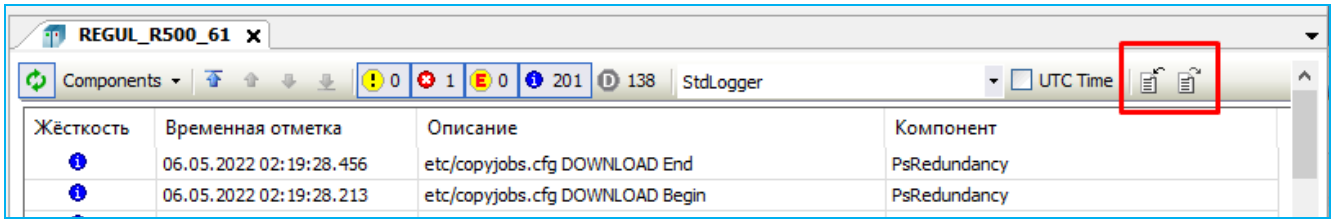


Рисунок 223 – Функции экспорта, импорта, удаления в оперативном журнале

Чтобы просмотреть ранее сохраненные журналы нажмите кнопку  (***Импорт элементов через существующий XML-файл***), найдите на компьютере файлы типа *xml files*. Выберите нужный log-файл, его записи будут показаны в отдельном окне.

Полный журнал событий

Полный журнал содержит записи обо всех событиях контроллера, включая ошибки и перезагрузку. Объем журнала можно задать, прописав параметры в конфигурационном файле для каждого лог-файла в */etc/plc.cfg*. Для изменения перейдите на вкладку **Экспертный режим**

(см. подраздел «Настройка системных параметров»), выберите конфигурационный файл **etc/plc.cfg** и в секции Logging, пропишите значения параметров: *DefaultMaxFileSize* (размер одного лог-файла, байт) и *DefaultMaxFiles* (количество лог-файлов для ротации). По умолчанию заданы следующие значения:

```
[Logging]
DefaultMaxFileSize = 5242880
DefaultMaxFiles = 5
```

ИНФОРМАЦИЯ

Параметр Logging/DefaultMaxFileSize ограничен диапазоном 65536-10485760 байт.

Для параметра Logging/DefaultMaxFiles ограничен диапазоном от 1 до 100 шт., значение по умолчанию применяется, если параметр не задан или задан, но он ≤ 0

При заполнении журнала самые старые файлы удаляются и замещаются более новыми. Поэтому, чтобы избежать потери информации, содержащейся в журнале, необходимо копировать лог-файлы сразу же после обнаружения сбоя в работе программы, до того, как они будут замещены. Скорость заполнения журнала событий зависит от количества включенных диагностических опций проекта.

Полный журнал событий из контроллера можно загрузить двумя способами: из среды разработки Astra.IDE или при помощи FTP – клиента.

Для загрузки через Astra.IDE выполните следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **logs** (Рисунок 224);

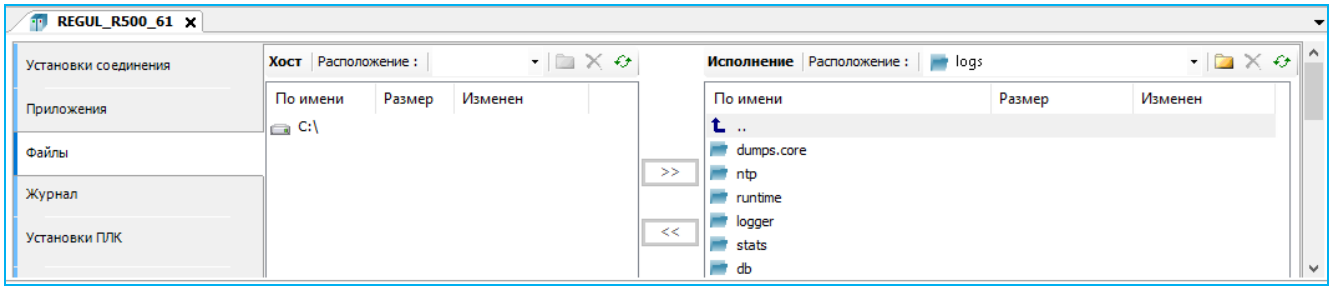


Рисунок 224 – Загрузка файлов с контроллера на ПК. Журнал событий

– в папке **logs** выберите необходимые файлы. Кнопкой << скопируйте файлы с контроллера на ПК (из **Исполнение** в **Хост**). Откройте на ПК файлы для просмотра.

Для загрузки журнала при помощи FTP – клиента выполните следующие действия:

– подключитесь к IP-адресу контроллера, используя следующие параметры: **порт 21**, протокол **FTP**, логин **plclogs**, пароль **service**. При возникновении проблем проверьте настройки (см. подраздел «Настройка системных параметров»);

– скопируйте все лог-файлы на свой ПК;

– откройте журнал с помощью редактора FTP-клиента (Рисунок 225). Временная метка для каждого события содержит данные с точностью до микросекунд.

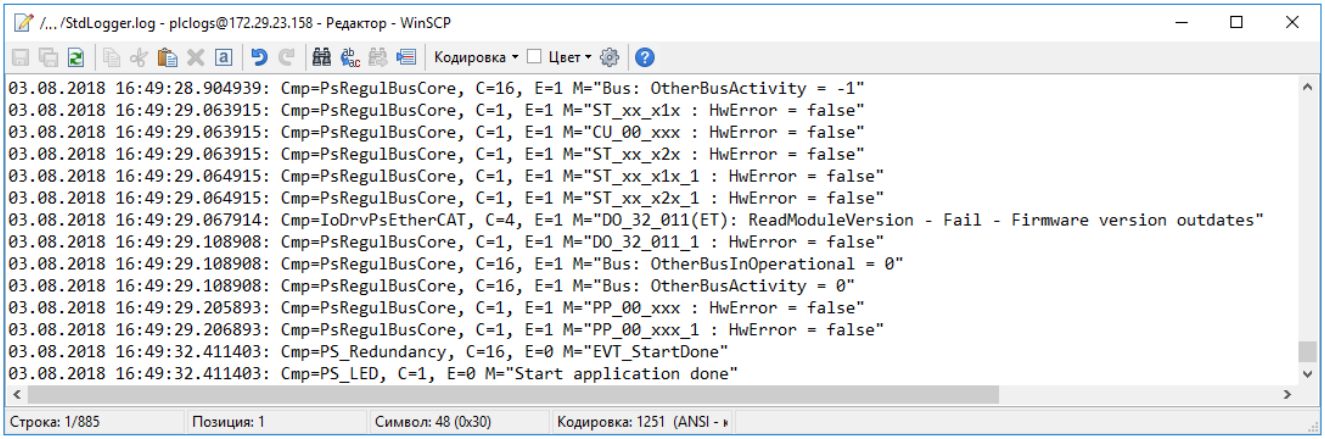


Рисунок 225 – Полный журнал событий

Журнал событий по МЭК задачам

Для ведения статистики событий по МЭК задачам присутствует отдельный журнал **SysTaskStatistic**, с которого дублируются записи в одноименный лог-файл ***SysTaskStatistic.log*** (Рисунок 226)

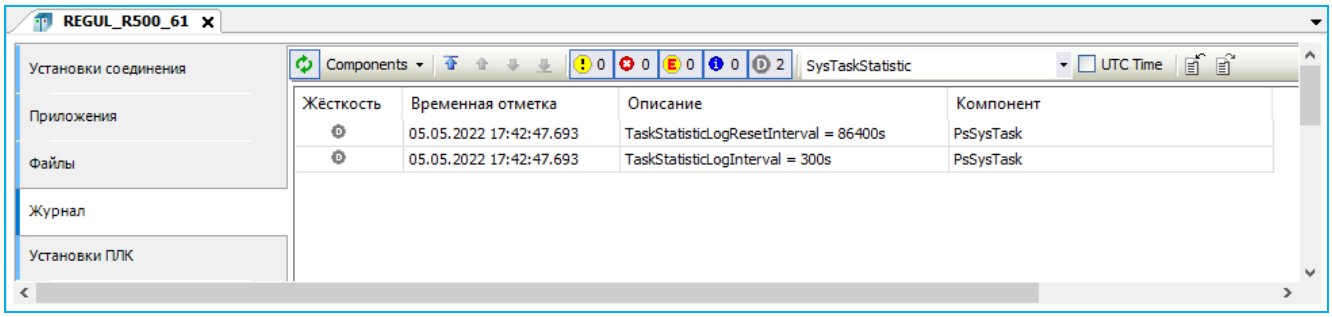


Рисунок 226 - Журнал событий по МЭК задачам

Интервал журналирования и сброс статистики можно задавать, прописав параметры в конфигурационном файле */etc/runtime.cfg*. Для изменения перейдите на вкладку **Экспертный режим** (см. подраздел «Настройка системных параметров»), выберите конфигурационный файл **etc/runtime.cfg** и в секции [PsSysTask], пропишите значения параметров: *TaskStatisticLogInterval* (интервал журналирования,) и *TaskStatisticLogResetInterval* (интервал сброса статистики). По умолчанию заданы следующие значения (интервалы задаются в секундах):

```
[PsSysTask]
TaskStatisticLogInterval=300
TaskStatisticLogResetInterval=86400
```

Для отключения параметров укажите значение 0.

В режиме онлайн информация о состоянии и текущей статистике по каждой МЭК задаче отображается на вкладке **Мониторинг** (см. подраздел «Конфигурация задач»). Данные по статистике журналируются в соответствии с описанием, приведенным в таблице 4.

Помимо журналирования статистики задач, в журнале **SysTaskStatistic** приводятся сообщения о следующих событиях:

– добавление/удаление МЭК задач;

– сброс статистики задачи;

– инициализация журнала и текущее значение интервала журналирования в лог-файл.

Журнал действий пользователя

Присутствует отдельный журнал **Audit** для отслеживания следующих действий пользователя (Рисунок 227):

– подключение к контроллеру (логин/логаут);

– старт/стоп приложения;

– сброс приложения (теплый, холодный, заводской);

– загрузка/онлайн-изменение приложения.

Записи дублируются в одноименный лог-файл ***Audit.log*** (расположен в каталоге /logs/runtime)

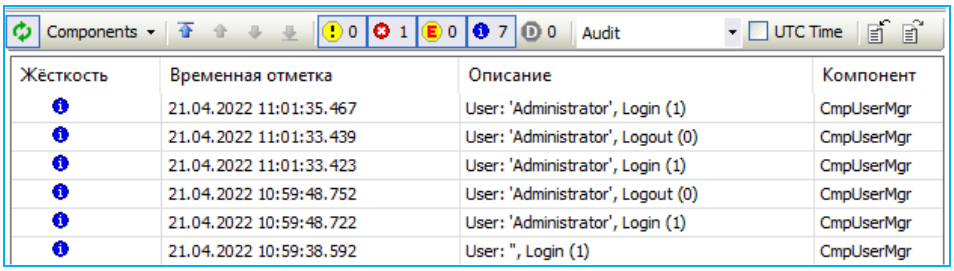


Рисунок 227 - Журнал действий пользователя Audit

Журнал сообщений операционный (системный)

Для оперативного отслеживания внесенных изменений/ошибок при конфигурировании пакетного фильтра (правил фильтрации), а также при включении/выключении механизма автомонтирования внешних накопителей USB, MMC/SD присутствует отдельный журнал **System** (Рисунок 228), с которого дублируются записи в одноименный лог-файл ***system.log*** (расположен в каталоге /logs/logger).

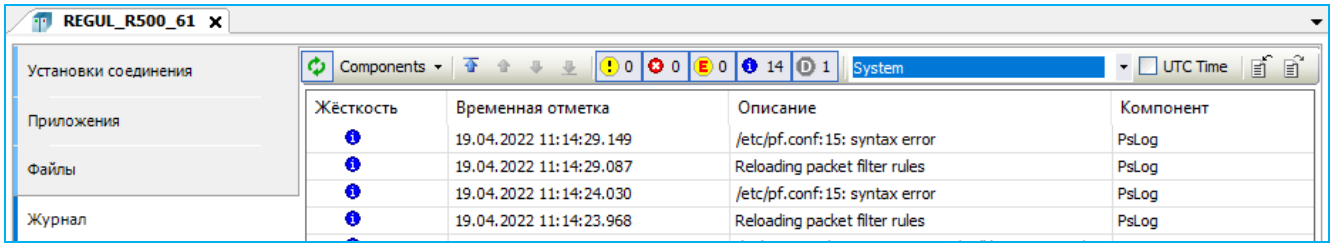


Рисунок 228 – Журнал System

Журнал работы шины RegulBus OS

Для оперативного отслеживания работы шины RegulBus OS присутствует отдельный журнал **regul-bus-driver** (Рисунок 229), с которого дублируются записи в одноименный лог-файл ***regul-bus-driver.log*** (расположен в каталоге /logs/logger).

Рисунок 229 - Журнал regul-bus-driver

Конфигурационный файл, находящийся в каталоге **/etc/runtime.cfg**, содержит настройки журналов **regul-bus-driver** и **system** (заданы по умолчанию), доступные пользователю для редактирования.

Настройка времени

Служба времени, входящая в состав СПО контроллера, работает по стандартам NTP и SNTP в соответствии с RFC 5905, RFC 5906, RFC 5907, RFC 5908.

Для просмотра и редактирования настроек времени и NTP перейдите на вкладку **Настройка времени**. Нажмите кнопку  (***Обновить***). На экран будет выведена информация о текущем времени на контроллере, данные NTP, источники синхронизации (Рисунок 230).

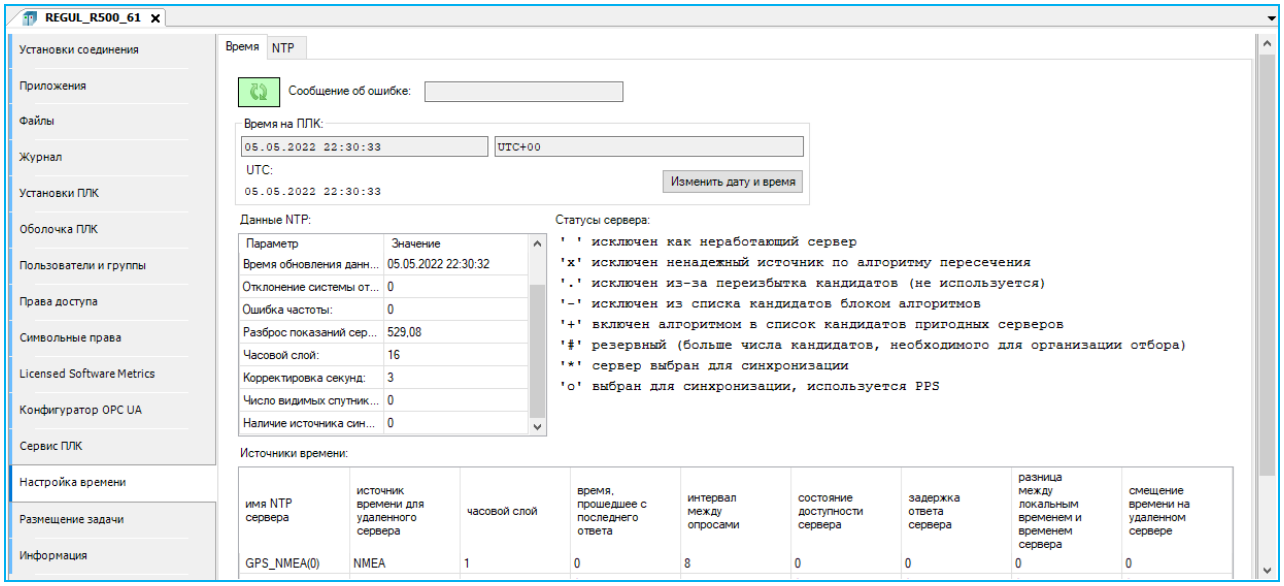


Рисунок 230 – Просмотр и настройка установок времени контроллера

Для смены времени контроллера нажмите кнопку ***Изменить дату и время***. Откроется диалоговое окно **Изменить дату и время** (Рисунок 231).

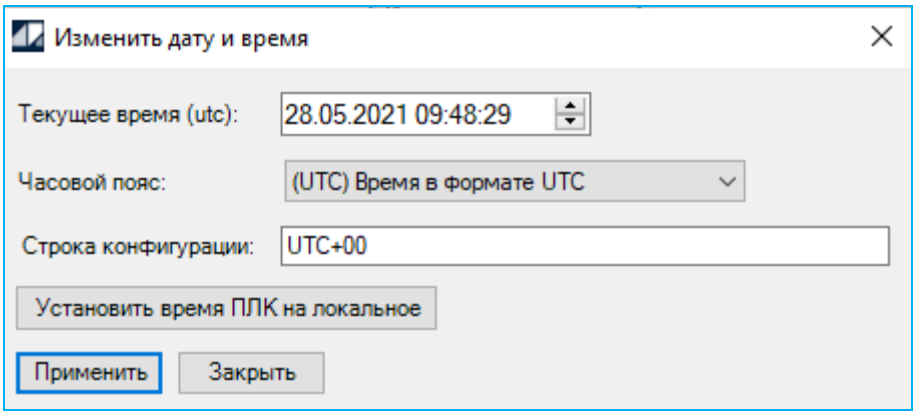


Рисунок 231 – Настройка времени и даты

Для удобства выбора часового пояса предусмотрен раскрывающийся список в поле **Часовой пояс** (на контроллере по умолчанию установлен часовой пояс в UTC). При выборе какого-либо пункта из этого списка автоматически проставляется значение в поле **Строка конфигурации**. Так, если в списке **Часовой пояс** выбран пункт *(UTC+05:00) Екатеринбург* (локальное время, относительно UTC), то в поле **Строка конфигурации** будет установлено значение *EKATST-05* (UTC относительно локального). Предусмотрена возможность вручную указать временную зону в строке конфигурации (в формате posix).

Предусмотрена возможность установки на ПЛК локального времени (текущее время с ПК). Для этого нажмите кнопку ***Установить время ПЛК на локальное***. Произойдет автоматическая перезапись значений полей, с пометкой символом «*» (Рисунок 232). Если вы согласны со значениями, нажмите кнопку ***Применить***. Если нет, то закройте окно.

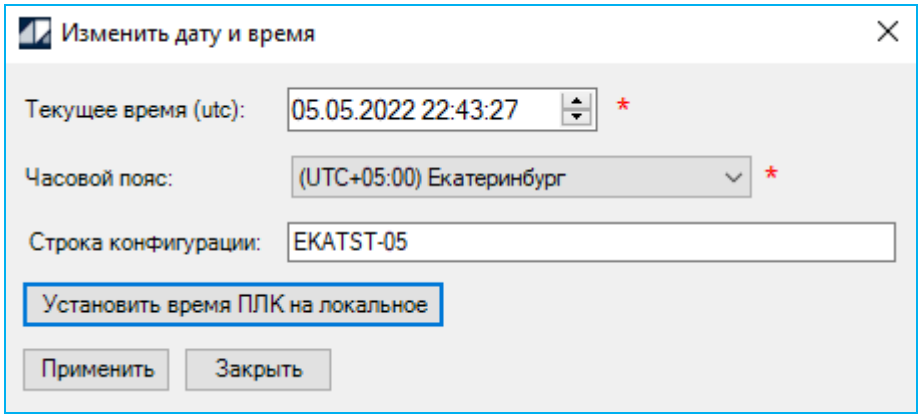


Рисунок 232 – Пример установки локального времени на ПЛК

При нажатии кнопки ***Применить*** откроется диалоговое окно (Рисунок 233).

Если вы согласны, то нажмите кнопку ***Да***. Для применения новых настроек потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Если не согласны, то закройте это окно или нажмите кнопку ***Нет***.

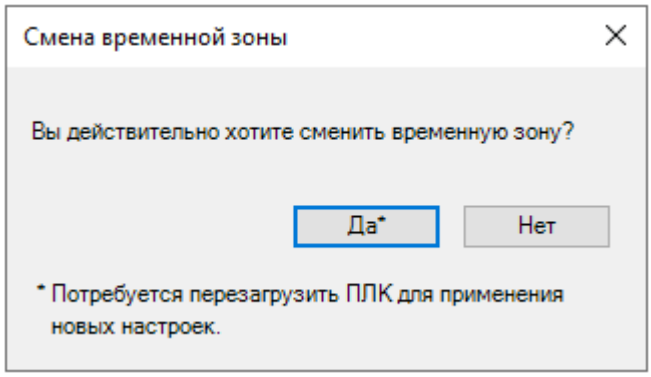


Рисунок 233 – Диалоговое окно на подтверждение «Смена временной зоны»

В области **Данные NTP:** отображаются значения параметров, описывающих статистику работы NTP (протокол взаимодействия с серверами точного времени). Описание основных параметров представлено в таблице 14.

Таблица 14 – Основные параметры описывающие работу NTP

Параметр	Описание
Доступность NTPD	Состояние службы NTPD. Возможные значения: 1 – работает, 0 – не работает. NTPD читает файл конфигурации ntp.conf при запуске, чтобы определить источники синхронизации и режимы работы
Признак наличия сигнала точного времени	Возможные значения: 1 – системное время синхронизировано и актуально, 0 – не актуально. Параметр принимает значение «1» при соблюдении одновременно трех условий: – значение параметра «Наличие источника синхронизаци» равно 1, – значение параметра «Отклонение системы от источника» не более 8 мс, – значение параметра «Разброс показаний сервера» не более 250 мс
Время обновления данных	Дата и время последнего обновления области данных
Отклонение системы от источника (offset)	Разница между локальным временем и временем удаленного узла/сервера, в мс. Значения с «-» - отставание, с «+» - часы спешат. В ходе синхронизации это значение должно приближаться к нулю
Ошибка частоты	Дрейф (смещение) часов, когда часы работают с разной скоростью отсчёта времени
Разброс показаний сервера	Максимальное значение временной ошибки, в миллисекундах
Часовой слой	Уровень сервера NTP (значения от 1 до 16)

Параметр	Описание
Корректировка секунд	Параметр предупреждает о корректировке секунд. Возможные значения: – 0 – нет коррекции; – 1 – последняя минута дня содержит 61 секунду; – 2 – последняя минута дня содержит 59 секунд; – 3 – неисправность сервера (время не синхронизировано)
Число видимых спутников	Количество спутников, с которых принимается информация
Наличие источника синхронизаци	Значение наличия источника синхронизации. Возможные значения: 1 – есть, 0 – нет

В области **Источники времени:(Peers)** отображается информация об источнике времени. Описание параметров источника времени представлено в таблице 15.

Таблица 15 – Параметры источника времени

Параметр	Описание
Имя NTP сервера (remote)	Имя удаленного узла/NTP-сервера (источник синхронизации)
Источник времени для удаленного сервера (refid)	Источник времени для удаленного узла/сервера (вышестоящий сервер/узел)
Часовой слой (stratum)	Часовой уровень источника времени (значения от 1 до 16)
Время, прошедшее с последнего ответа (when)	Время, прошедшее с последнего момента ответа узла/сервера (определяет, как давно была произведена синхронизация)
Интервал между опросами (poll)	Значение определяется частотой опроса в 2^n секунд (то есть, если poll=8, то опрос происходит раз в 2^8=256 секунд)
Состояние доступности сервера (reach)	Восьмеричное число, показывающее результаты последних восьми попыток соединения с сервером. Например, 0 – ни одной успешной попытки, 377_8=11111111_2 – все попытки успешны. Последовательно должны отображаться значения reach: 0, 1, 3, 7, 17, 37, 77, 177, 377 (значение стабилизируется на уровне 377)
Задержка ответа сервера (delay)	Время, необходимое для получения ответа на запрос, в мс
Разница между локальным временем и временем сервера (offset)	Разница между локальным временем и временем удаленного узла/сервера, в мс. Значения с «-» - отставание, с «+» - часы спешат. В ходе синхронизации это значение должно приближаться к нулю
Смещение времени на удаленном сервере (jitter)	Чем меньше это значение, тем более точная возможно синхронизация, в мс

В области **Статусы сервера: (Peer status)** описаны спецсимволы, которые используются для обозначения статуса узла/сервера (спецсимвол устанавливается перед именем узла/сервера).

Для настройки NTP перейдите на вкладку **NTP** и нажмите кнопку ***Получить файл настроек ntp***. В этом же окне будет открыт файл ntp.conf (Рисунок 234). Поместите курсор на ту строку, которую требуется отредактировать, внесите изменения. Для сохранения файла *ntp.conf* с новыми данными нажмите кнопку ***Сохранить файл настроек ntp в ПЛК***.

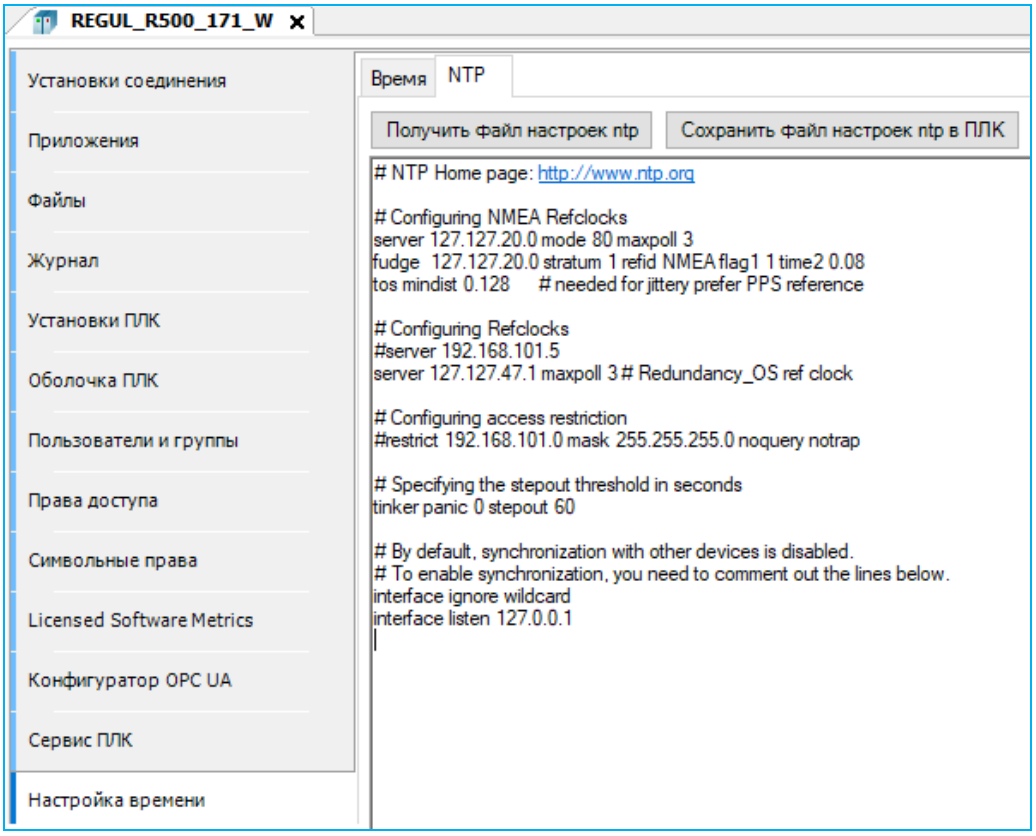


Рисунок 234 – Редактирование настроек NTP

Пустые строки и строки, начинающиеся с символа "#", игнорируются.

По умолчанию сервер NTP работает в режиме клиента и синхронизация с другими устройствами отключена, кроме синхронизации времени по GPS, согласно следующих строк:

– *interface ignore wildcard* - игнорировать список всех интерфейсов;

– *interface listen 127.0.0.1* - прослушивать только адрес 127.0.0.1, т.е. данные от GPS приемника.

Чтобы включить синхронизацию с другими устройствами, необходимо снять ограничения, поставив символ "#" в начале каждой строки (*#interface ignore wildcard, #interface listen 127.0.0.1).*

Настройки передачи информации по протоколу NMEA и PPS при использовании GPS приемника определяется следующими строками:

```
#Configuring NMEA Refclocks
server 127. 127.20.0 mode 80 maxpoll 3
fudge 127. 127.20.0 stratum 1 refid NMEAflag1 1 time2 0.08
```

```
tos mindist 0.128 # needed for jittery prefer PPS reference
```

Для добавления сервера синхронизации используйте параметр *server*:

– локальный сервер времени:

```
# Configuring Refclocks
server 127.127.1.0
fudge 127.127.1.0 stratum 2
```

– свой сервер времени (возможно указать несколько, по одному в каждой строке):

```
# Configuring Refclocks
server 172.29.23.25
```

Если потребуется раздавать время, то необходимо, чтобы NTP сервер получил время с любого другого источника времени (gps, local, ntp server) и обозначил этот источник как systems peer.

Параметр *restrict* позволяет контролировать, какие устройства могут обращаться к вашему серверу. Если вы хотите разрешить синхронизировать часы с вашим сервером только устройствам в вашей сети, но запретить им настраивать сервер или быть равноправными участниками синхронизации времени, то уберите символ "#" в начале строки:

```
restrict 192.168.101.0 mask 255.255.255.0 noquery notrap
```

(noquery – запрет любых запросов для синхронизации времени, поступающих с других серверов, notrap – запрет приема управляющих сообщений).

Для того, чтобы время скорректировалось максимально быстро, прописана следующая строка с параметрами:

```
# Configuring Refclocks
tinker panic 0 stepout 60
```

Параметры:

– tinker panic 0 – снимает запрет на запуск ntpd, в случае, когда время на серверах (к которым производится обращение) отличается от локального времени более, чем на 300 секунд.

– stepout 60 – таймаут ожидания сервером перед началом синхронизации времени. Если установлен параметр *panic 0*, сервер запущен, и при этом он обнаруживает, что время вышестоящих серверов отличается от его времени более, чем на 0.128 секунд, то он не синхронизирует его сразу, а по истечении 60 секунд (по умолчанию 900 секунд).

Источник времени от МЭК-приложения

В NTPD существует возможность получения времени из прикладного ПО. В библиотеке SysTimeRtc реализована возможность установки времени через источник времени 127.127.47.x. Данный источник будет являться поставщиком времени для NTP.

Для включения функционала необходимо изменить значение параметра SetTimeWithNTPD, выполнив следующие действия:

– перейдите на вкладку **Сервис ПЛК ⇨ Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку (***Обновить***);

– выберите название каталога конфигурационный файл **etc/runtime.cfg**. В секции [*SysTimeRtc*] установите значение параметра SetTimeWithNTPD равным 1 (по умолчанию отключен или равен 0):

– [*SysTimeRtc*] SetTimeWithNTPD=1

– нажмите кнопку **Сохранить**.

Для установки времени из ПО используются стандартные функции SysTimeRtcSet() или SysTimeRtcHighResSet() из библиотеки SysTimeRtc. Данные функции принимают в качестве параметра метку времени в формате UTC.

Для применения источника времени на вкладке **NTP** добавьте в конфигурационный файл ntp.conf строки, например:

```
server 127.127.47.0
fudge 127.127.47.0 stratum 10 time1 XXXX time2 YYYY flag1 1

(time1 XXXX – калибровочная константа для регулировки номинального временного
сдвига, где XXXX - десятичное число с фиксированной точкой в секундах;
time2 YYYY – определяет сброс таймера обнуления, где YYYY - десятичное число с
фиксированной точкой в секундах. Если опция не указана, то применяется значение по
умолчанию – 300 сек;
flag1 1 – для фиксации в лог файле NTPD, полученного от МЭК-приложения времени).
```

Блокировка сигналов точного времени от спутников

Предусмотрена возможность блокировки сигналов точного времени от спутников.

Для этого необходимо изменить значение параметра, выполнив следующие действия (Рисунок 235):

– перейдите на вкладку Сервис ПЛК ⇨ Системные параметры ⇨ Экспертный режим. Нажмите на кнопку (*Обновить*);

– выберите название каталога конфигурационный файл **etc/plc.cfg.** В секции [*GNSS*] измените значения параметра *GNSSparam* согласно описанию:

- **0** – все отключены;
- **1** – только от спутников системы GPS;
- **2** – только от спутников системы GLONASS;
- **3** – от спутников системы GPS и GLONASS;
- **4** – только от спутников системы QZSC;

- **5** – от спутников системы GPS и QZSC;
- **6** – от спутников системы GLONASS и QZSC;
- **7** (по умолчанию) – от спутников системы GPS, GLONASS и QZSC;

– нажмите кнопку **Сохранить**.

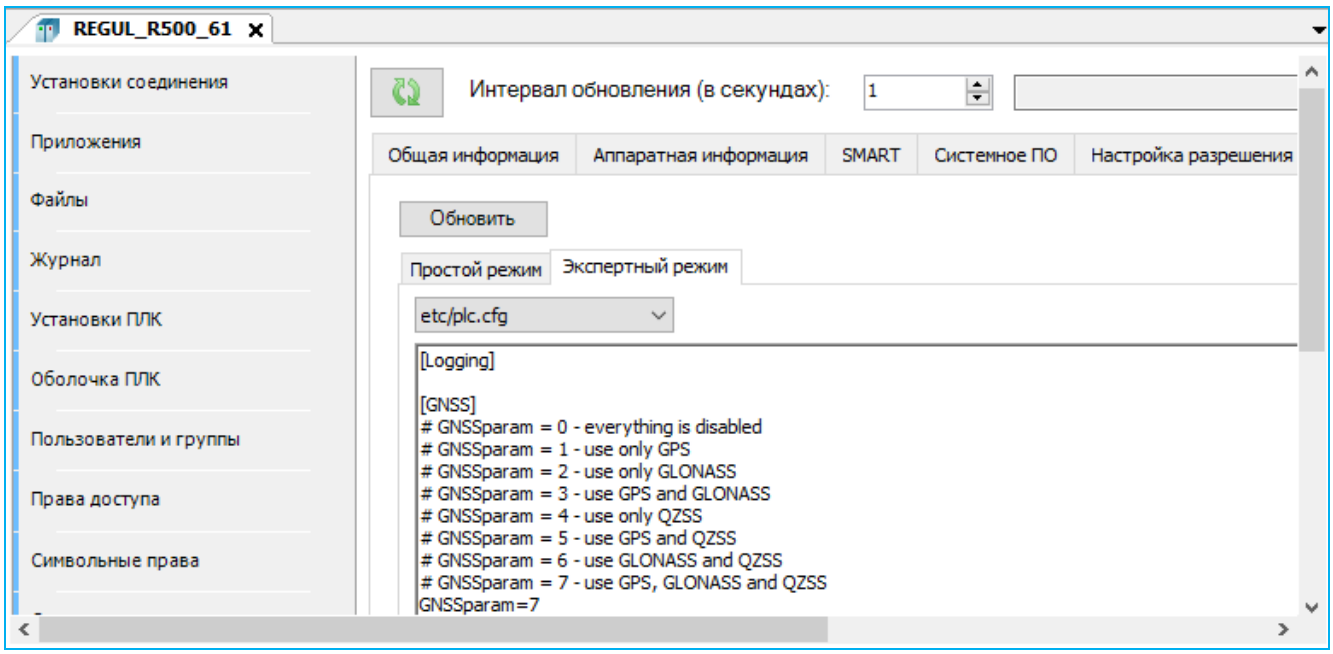


Рисунок 235 – Возможность блокировки сигналов точного времени от спутников

Остановка службы NTP при старте

Предусмотрена возможность остановки службы NTP при старте контролера. Во время первоначальной синхронизации с сервером будет производится проверка разницы между полученным временем и локальным. В случае, когда допустимое время расхождения (параметр tinker panic) с источником времени будет превышено - локальное время контроллера не перезаписывается и служба NTP останавливается.

Для этого необходимо изменить значение параметра, выполнив следующие действия:

– в Astra.IDE на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку  (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Найдите папку **etc** (Рисунок 236);

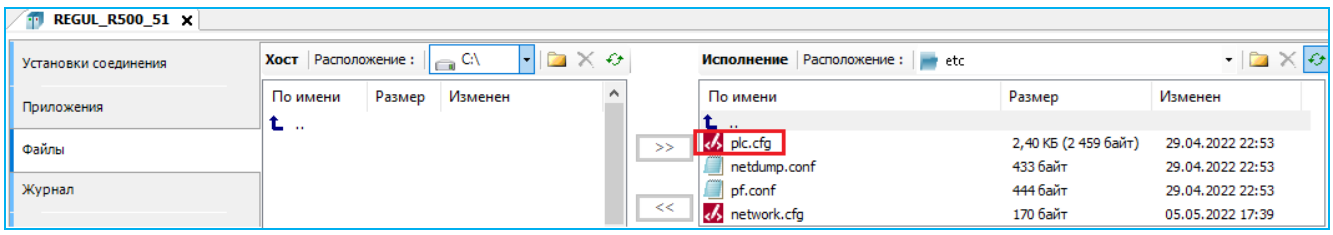


Рисунок 236 – Обмен файлами между ПК и контроллером

– в папке **etc** найдите файл **plc.cfg**. Кнопкой << скопируйте этот файл с контроллера на ПК (из **Исполнение** в **Хост**);

– откройте на ПК файл **plc.cfg**. (Рисунок 237). В секции [*NTP]* установите значение параметра *AllowPanic* равным 0 (по умолчанию равен 1):

– [NTP] AllowPanic=0

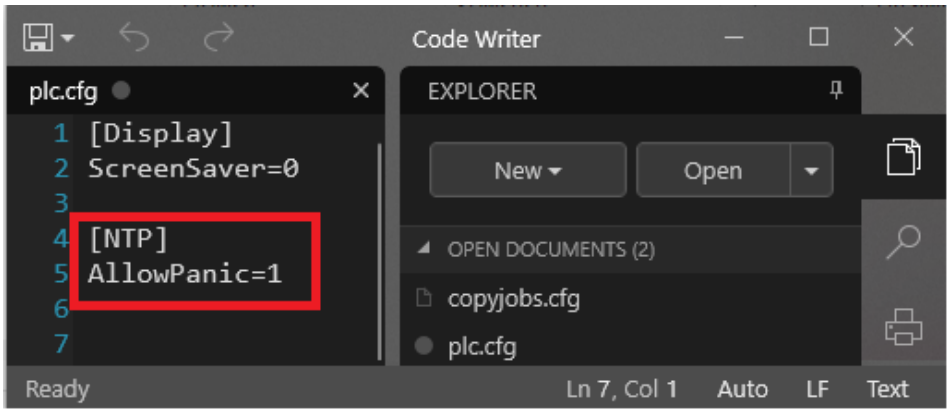


Рисунок 237 – Конфигурационный файл plc.cfg

– в Astra.IDE на вкладке **Файлы** кнопкой >> скопируйте измененный файл с ПК на контроллер (из **Хост** в **Исполнение**).

Для вступления в силу изменений потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Альтернативный вариант настройки - перейти на вкладку **Сервис ПЛК ⇨Системные параметры⇨Экспертный режим** и выбрать для редактирования конфигурационный файл ***plc.cfg***.

Настройка синхронизации времени по протоколу PTP

Контроллеры модели R500 (с модулями ЦП I-го/III-го типа) поддерживают синхронизацию времени по протоколу PTP (версия протокола 2.2.2). Изначально необходимо определить какой из двух модулей ЦП будет «эталоном» – ведущим (*Master*), с которого будет считываться системное время, а какой будет ведомым (*Slave*). Канал синхронизации времени между двумя модулями ЦП организован на портах с интерфейсом Ethernet.

⚠	**ВНИМАНИЕ!** В резервированном контроллере запрещается применять линию синхронизации между модулями ЦП для осуществления синхронизации времени по протоколу PTP

Далее необходимо поочередно подключиться к каждому модулю ЦП (из среды разработки Astra.IDE или при помощи FTP – клиента) и задать параметры в конфигурационном файле */etc/ptp.conf*.

Для работы с файлом через Astra.IDE выполните следующие действия:

– на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на ЦП. Найдите папку **etc** (Рисунок 238);

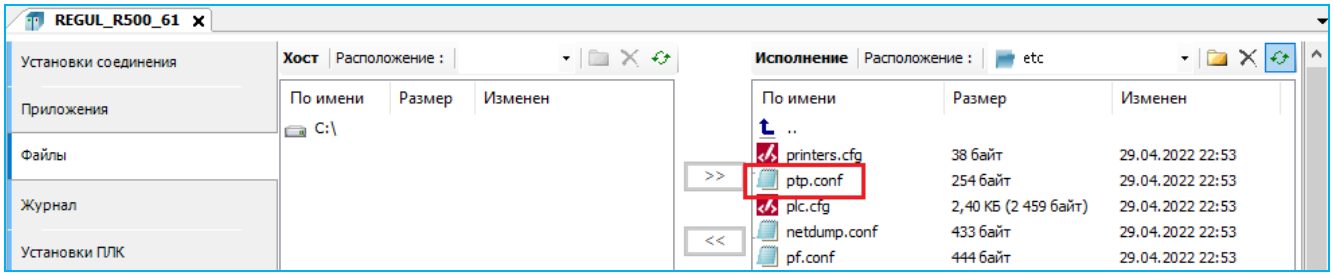


Рисунок 238 – Расположение конфигурационного файла ptp.conf

– кнопкой << скопируйте файл с модуля ЦП на ПК (из **Исполнение** в **Хост**);

– откройте на ПК файл ***ptp.conf*** (Рисунок 239). Задайте значения строк, руководствуясь приведенным описанием параметров в приложении З;

```
#Состояние: Enable или Disable
State=Disable

#Режим работы ptp: Master или Slave
Mode=Master

#Уровень логирования: Error, Warning, Notice, Info, Debug
LogLevel=Error

#Порт
Port=30
```

Рисунок 239 – Конфигурационный файл ptp.conf

– сохраните изменения в файле ***ptp.conf***;

– в Astra.IDE на вкладке **Файлы** кнопкой >> скопируйте измененный файл с ПК на модуль ЦП (из **Хост** в **Исполнение**).

Аналогичные настройки можно произвести, подключившись к модулю ЦП с помощью FTP.

Для работы с файлом при помощи FTP – клиента выполните следующие действия:

– подключитесь к IP-адресу модуля ЦП, используя следующие параметры: порт 21, протокол **FTP**, логин **plclogs**, пароль **service**. При возникновении проблем проверьте настройки (см. подраздел «Настройка системных параметров»);

– скопируйте файл ***ptp.conf*** на свой ПК и задайте основные параметры согласно описанию в приложении З.

Альтернативный вариант настройки - перейти на вкладку **Сервис ПЛК ⇨Системные параметры⇨Экспертный режим** и выбрать для редактирования конфигурационный файл ***ptp.conf***.

Настройка дисплея

Выбор разрешения дисплея

Для контроллеров с видеовыходом в программе предусмотрена возможность установки разрешения дисплея.

Перейдите на вкладку **Сервис ПЛК**, далее на внутреннюю вкладку **Настройка разрешения**. Нажмите кнопку ***Обновить список***. В блоке **Поддерживаемое разрешение** будет показан список разрешений дисплея, поддерживаемых данным контроллером (Рисунок 240).

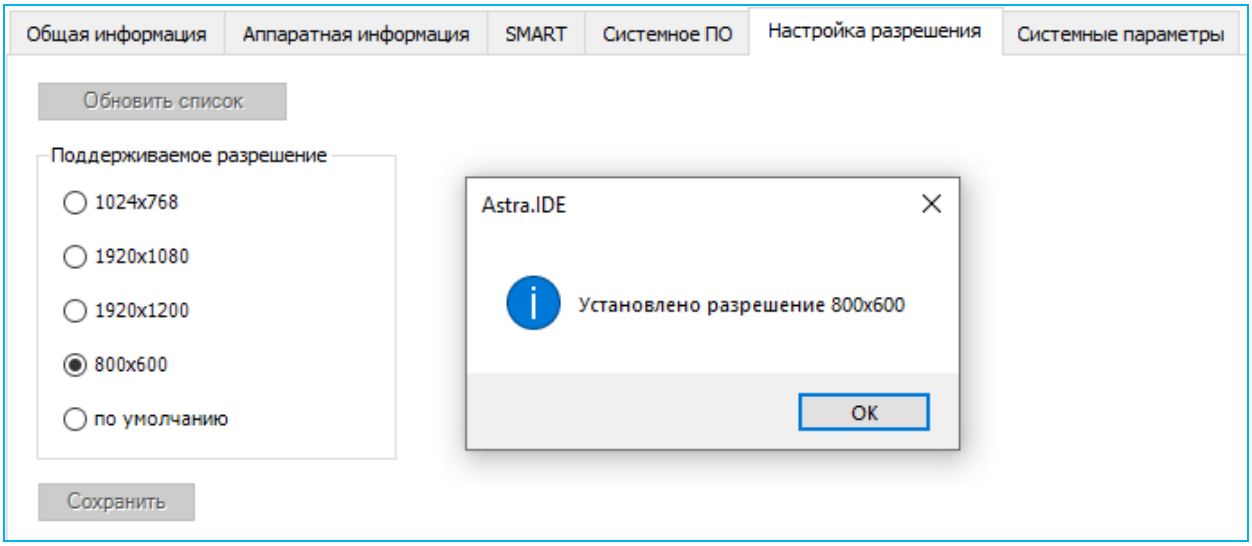


Рисунок 240 – Выбор разрешения дисплея

Выберите необходимое разрешение и нажмите кнопку ***Сохранить***. Появится информационное окно, где будет указано, что разрешение дисплея изменено на выбранное. Если вы согласны с этим, нажмите кнопку ***ОК***. Если нет, то закройте это окно и выберите другое значение.

	**ИНФОРМАЦИЯ** При изменении разрешения дисплея потребуется повторное проведение калибровки

Выбрав разрешение, убедитесь, что присутствует физическое подключение монитора к модулю ЦП. Перезагрузите контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

	**ВНИМАНИЕ!** Если на момент загрузки ПЛК отсутствует физическое подключение монитора к модулю ЦП (вне зависимости от разрешения экрана), то на «горячую» подключить монитор уже не получится. Если ПЛК проходил процесс загрузки с подключенным монитором, то «горячее» отключение/подключение монитора будет функционировать. При выборе разрешения 1920x1080, загрузка контроллера произойдет только при наличии физического подключения монитора к модулю ЦП. При этом монитор может быть не включен или обесточен

Калибровка сенсорного экрана

Калибровка сенсорного экрана возможна по трем и четырем точкам. Как правило, калибровка по трем точкам применяется при использовании емкостного типа экрана, по четырем точкам при использовании резистивного типа экрана.

Для проведения процедуры калибровки экрана перейдите на вкладку **Оболочка ПЛК**. В нижней части экрана расположено окно ввода команд. Введите команду **touchpanel calib3** для калибровки экрана по трем точкам или **touchpanel calib** для калибровки экрана по четырем точкам. Нажмите клавишу ***Enter*** (Рисунок 241).

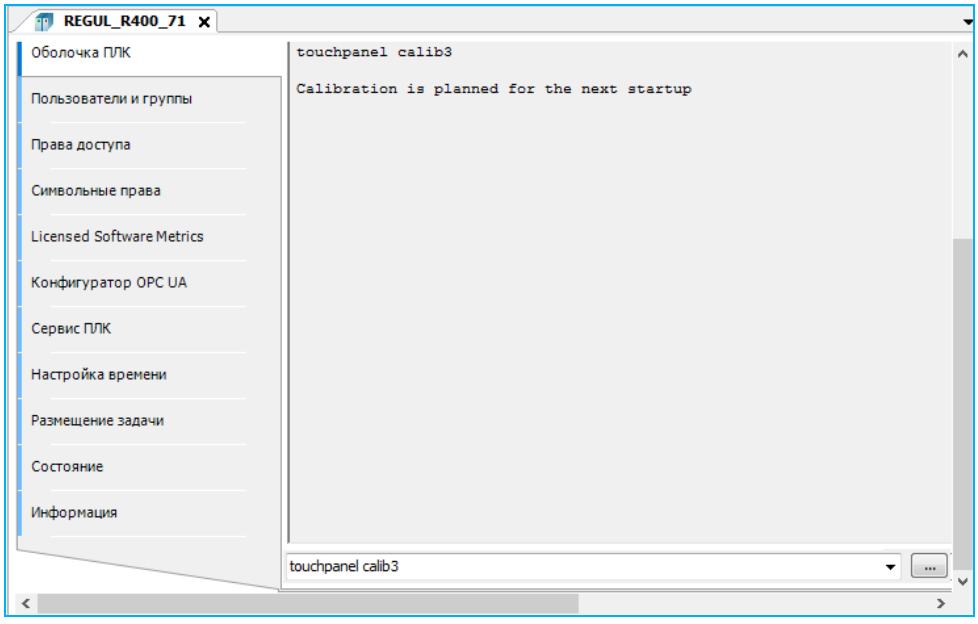


Рисунок 241 – Ввод команды калибровки экрана

После ввода команды требуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

При загрузке контроллера на дисплее появится экран калибровки (Рисунок 242).

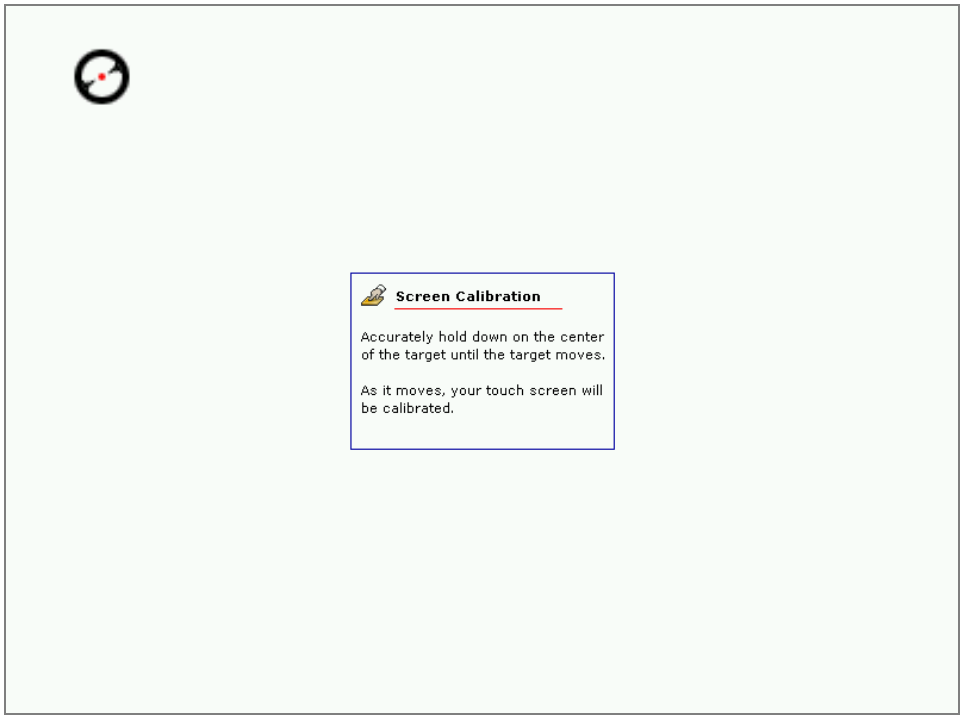


Рисунок 242 – Экран калибровки

В соответствие с инструкцией на экране, требуется точно нажимать (с удержанием) на предлагаемую точку, пока она не переместится в другую область экрана. После того, как все предлагаемые точки будут пройдены, появится диалоговое окно принятия результата калибровки (Рисунок 242). Нажмите кнопку ***Accept***.

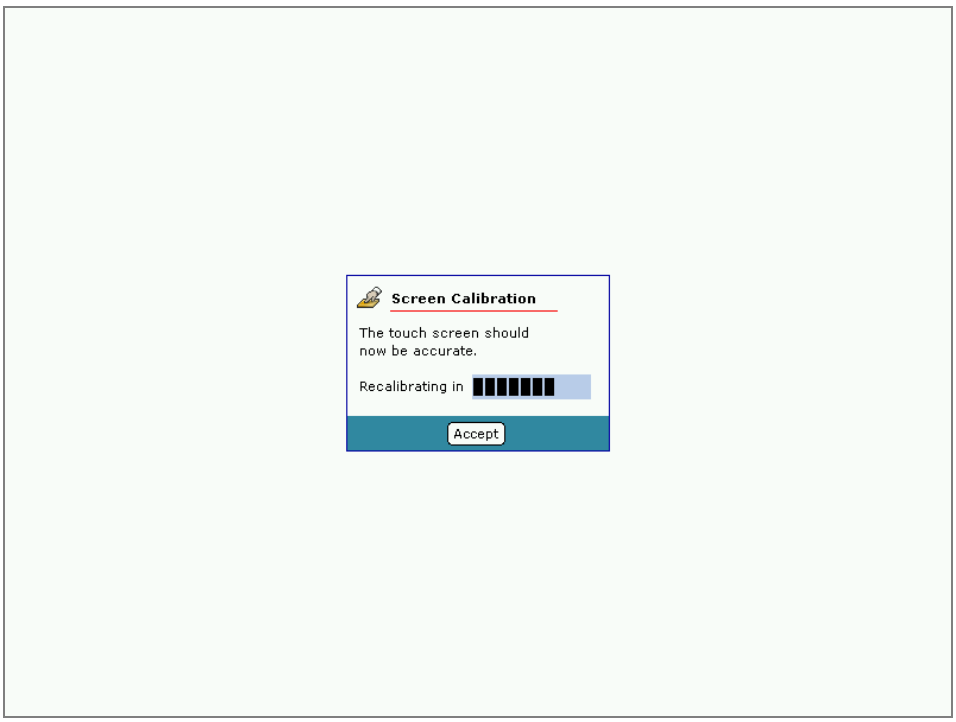


Рисунок 243 – Диалоговое окно принятия результата калибровки

Затем контроллер перезагрузится для принятия и сохранения координат. Процедура калибровки завершена.

i	**ИНФОРМАЦИЯ** При изменении разрешения дисплея потребуется повторное проведение калибровки

Изменение фонового изображения при загрузке контроллера

В контроллере предусмотрена возможность изменения фонового изображения, появляющегося на экране при загрузке. Для этого необходимо выполнить следующие действия:

– в Astra.IDE на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Создайте директорию (например, папку под названием **background**, рисунок 244);

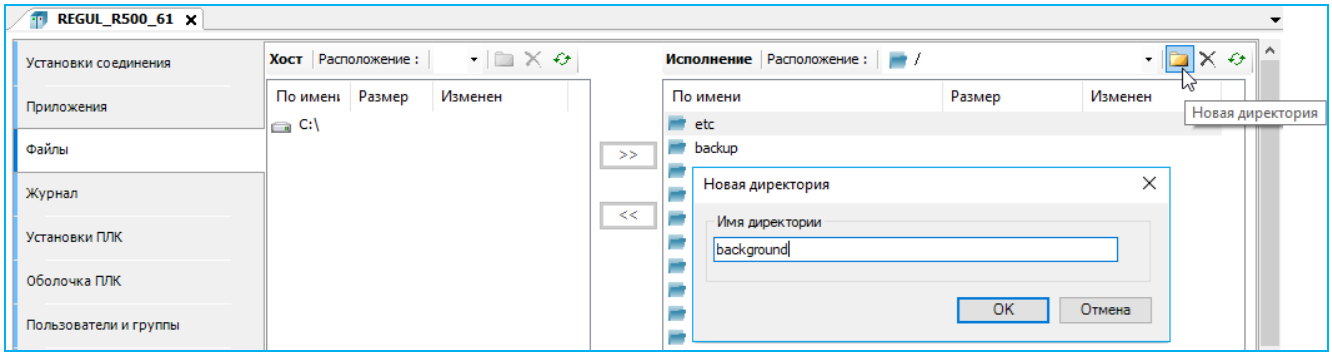


Рисунок 244 - Создание новой директории

– откройте папку **background** и кнопкой >> скопируйте необходимый файл с расширением *.png с ПК на контроллер (из **Хост** в **Исполнение,** рисунок 245);

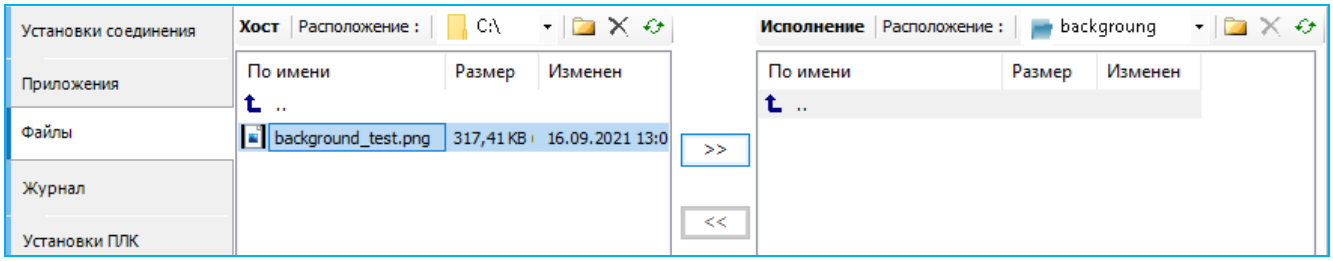
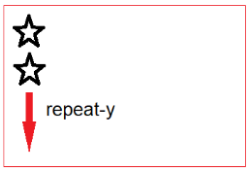


Рисунок 245 - Обмен файлами между ПК и контроллером

– перейдите на вкладку **Сервис ПЛК ⇨Системные параметры ⇨ Экспертный режим**. Нажмите на кнопку (*Обновить*);

– в секции [BackgroundImage] конфигурационного файла *etc/plc.cfg* (см. «Приложение Е») задайте значения строк, руководствуясь приведенным описанием параметров в таблице 16.

Таблица 16 – Параметры секции BackgroundImage

Параметр	Описание	Значение по умолчанию
File	Определяет путь до пользовательского изображения на ПЛК. Пример: File=background/background_test.png	**-**
Fill	Определяет повтор изображения по горизонтали(x)/вертикали(y), со следующими возможными значениями (при некорректном вводе применяется значение по умолчанию): – none – параметр выключен; – repeat-x – повторить изображение по горизонтали; –  – repeat-y – повторить изображение по вертикали; –  – repeat-xy – повторить изображение по горизонтали и вертикали. Пример: Fill=repeat-xy	**none**
Align	Определяет место расположения изображения на экране, со следующими возможными значениями (при некорректном вводе применяется значение по умолчанию): – center – выравнивание по центру; – top – выравнивание по верхнему краю; – bottom – выравнивание по нижнему краю; – left – выравнивание по левому краю; – right – выравнивание по правому краю. Пример: Align=top	**center**
ShowPlcInfo	Включает/выключает отображение информации о ПЛК (модель ЦП и версия СПО) (при некорректном вводе применяется значение по умолчанию): – yes – включено; – no – выключено. Пример: ShowPlcInfo=no	**yes**

Нажмите кнопку **Сохранить**. Для вступления в силу изменений потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Изменение режима работы подсветки дисплея R400

В целях сохранения ресурса и продления срока службы экрана, предусмотрена возможность отключения подсветки дисплея по истечении заданного интервала времени, в случае отсутствий манипуляций с контроллером со стороны пользователя. При отключении подсветки экрана выполнение запущенных программ на контроллере продолжается. Чтобы включить подсветку дисплея обратно, необходимо коснуться экрана или нажать клавишу на клавиатуре (или переместить мышку).

Для изменения настроек подсветки дисплея, выполните следующие действия:

– в Astra.IDE на главной вкладке параметров устройства перейдите на вкладку **Файлы**. В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере R400. Найдите папку **etc** (Рисунок 246);

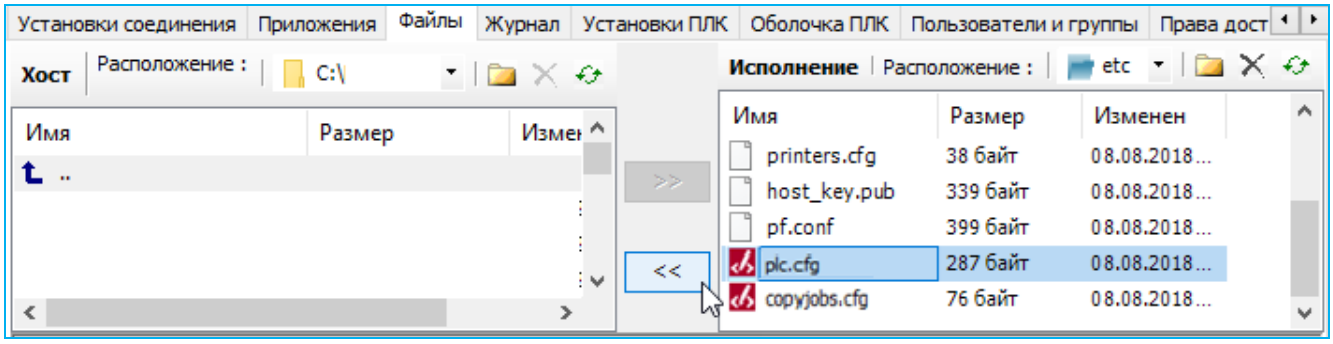


Рисунок 246 – Обмен файлами между ПК и контроллером

– в папке **etc** найдите файл **plc.cfg**. Кнопкой << скопируйте этот файл с контроллера на ПК (из **Исполнение** в **Хост**);

– откройте на ПК файл **plc.cfg**. (Рисунок 247). Задайте необходимое значение времени бездействия в строке *ScreenSaver=...*, по истечении которого, дисплей экрана будет гаснут (например, через 20 сек)

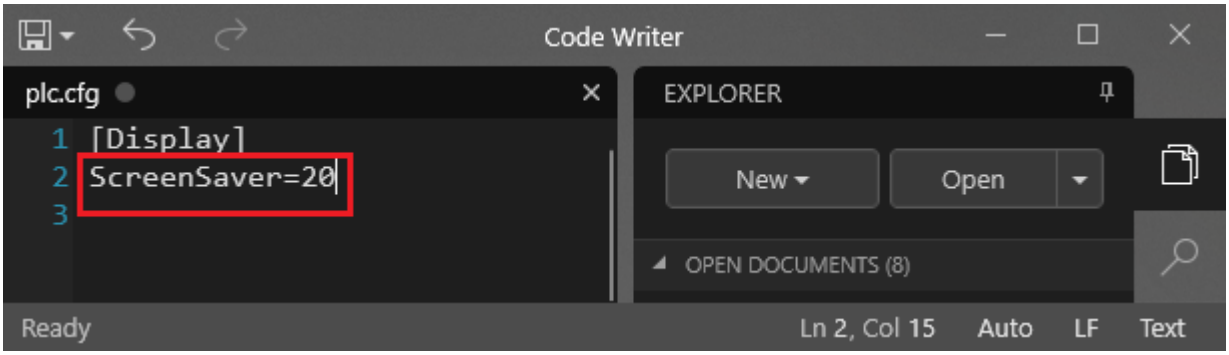


Рисунок 247 – Конфигурационный файл plc.cfg

– в Astra.IDE на вкладке **Файлы** кнопкой >> скопируйте измененный файл с ПК на контроллер (из **Хост** в **Исполнение**).

Альтернативный вариант настройки - перейти на вкладку **Сервис ПЛК ⇨Системные параметры⇨Экспертный режим** и выбрать для редактирования конфигурационный файл ***plc.cfg***.

Для вступления в силу изменений потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Отключение сенсорного экрана на время загрузки контроллера

Предусмотрена возможность отключения сенсорного экрана на время загрузки контроллера.

Для этого необходимо добавить значение параметра, выполнив следующие действия (Рисунок 248):

– перейдите на вкладку Сервис ПЛК ⇨ Системные параметры ⇨ Экспертный режим. Нажмите на кнопку (*Обновить*);

– выберите название каталога конфигурационный файл **etc/plc.cfg.** Добавьте секцию [*Startup*] с параметром *TouchScreenOnStartup* равным Disable;

– нажмите кнопку **Сохранить**.

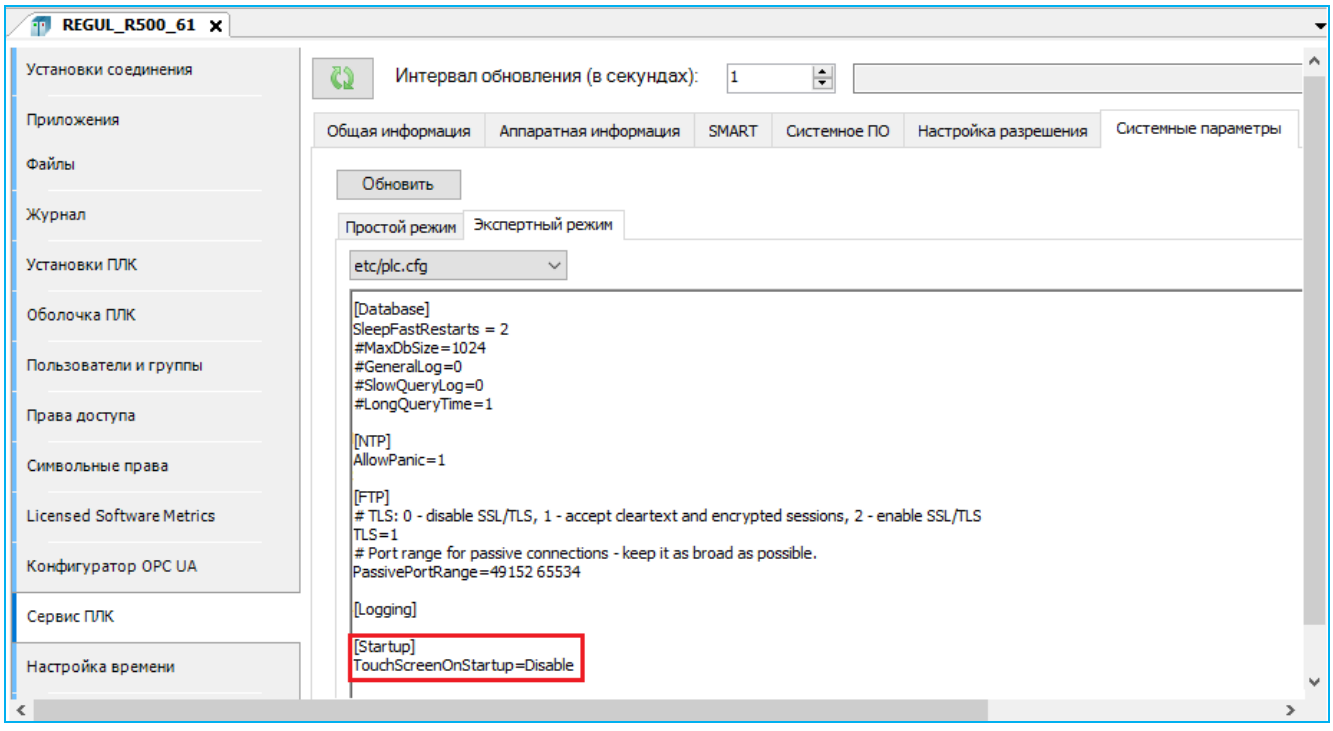


Рисунок 248 – Отключение сенсорного экрана контроллера на время загрузки

Управление яркостью подсветки сенсорного экрана контроллера R400

Управление яркостью экрана доступно с помощью функции *setBacklight* в библиотеке **PsLed** (см. «Приложение Л»). Если устройство не поддерживает данную функцию, то в файле sloginfo.log (путь …logs/logger/user/sloginfo.log) появятся сообщения об ошибке:

```
«…: [E] 11024.0 register_supported Command 0xfd not supported register 0x03
…: [E] 11024.0 plc400tch_dev_devctrl CTRL_DRV_PLC400_TFT error 48 (id=0xfd,
reg=0x03, value=50) »
```

	**ИНФОРМАЦИЯ** Обновление ПО, отвечающего за управление экраном контроллера, доступно только на предприятии-изготовителе

Резервное копирование и восстановление

В программе предусмотрена возможность создать резервную копию текущего состояния системы, восстановить систему с помощью ранее сохраненного файл-образа.

	**ВНИМАНИЕ!** Резервная копия, выполненная на системе с одной версией ОС, не может быть установлена в ПЛК с другой версией ОС (например, KPDA)

	**ВНИМАНИЕ!** Допускается устанавливать резервные копии только на ПЛК с идентичной версией СПО. В противном случае корректное восстановление данных не гарантируется. Для обеспечения данного требования необходимо перед резервным восстановлением копии выполнить обновление СПО ПЛК до нужной версии

	**ВНИМАНИЕ!** В ПЛК невозможно установить резервную копию с другой аппаратной конфигурацией. Установка файл-образа с другой аппаратной конфигурацией завершится записью сообщения в файл restore/restore.err: «Restore operation can not be performed: file not supported on this platform»

Для создания в контроллере резервной копии установите связь с контроллером и перейдите на вкладку **Сервис ПЛК**, далее на внутреннюю вкладку **Системное ПО**. В блоке **Резервные копии** поставьте переключатель на нужную позицию:

- **Все** – сохраняется все ПО контроллера;
- **Выборочно**:
  - **Системные данные и ПО** – сохраняется только системная часть ПО контроллера (без прикладных данных);

- **Прикладные данные** – сохраняются только прикладные данные (приложения, логи и так далее):
  - **Приложение и файлы**– сохраняются только приложения и файлы;
  - **База данных**– сохраняется только база данных.

	**ИНФОРМАЦИЯ** При создании резервной копии прикладных данных учетные записи FTP сервера не сохраняются

Далее нажмите кнопку ***Создать резервную копию***. Устанавливается файл-триггер, содержащий описание того, что будет сохранено в резервной копии. На экране выводится сообщение: «*Триггер установлен, перезагрузите контроллер для создания снимка текущего состояния системы*». Нажмите кнопку ***ОК***. Выполните перезагрузку контроллера. На контроллере будет создана резервная копия (снимок состояния системы, файл-образ), никаких дополнительных сообщений пользователю не выдается.

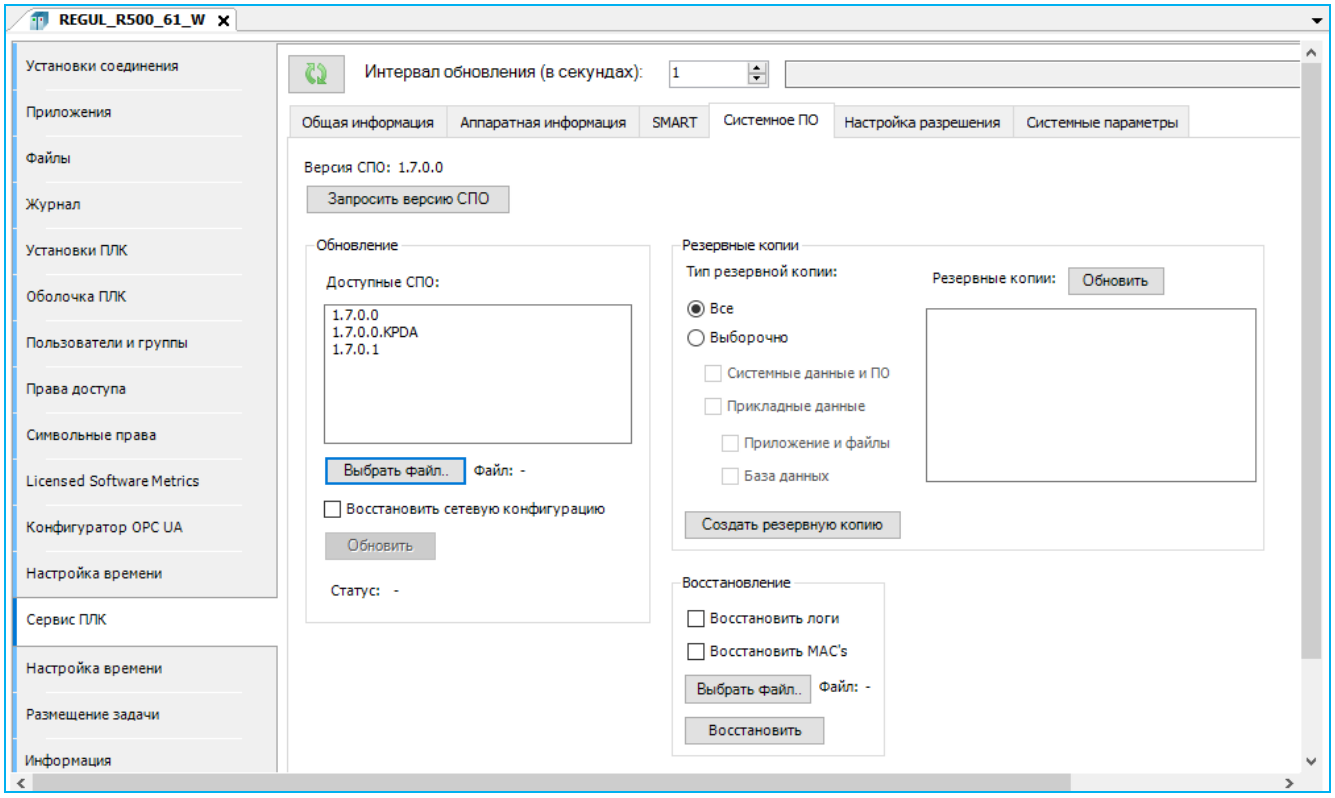


Рисунок 249 – Резервное копирование и восстановление

Если в поле **Резервные копии:** нажать кнопку ***Обновить***, то на экране отобразится список всех резервных копий, имеющихся на контроллере. Для того, чтобы скопировать файл с резервной копией с контроллера на ПК, перейдите на вкладку **Файлы** (Рисунок 250).

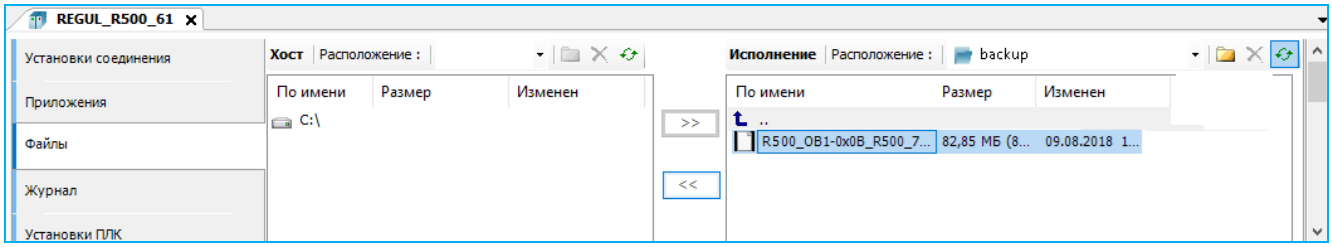


Рисунок 250 – Вкладка «Файлы» для копирования резервной копии с ПЛК на ПК

В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Резервные копии находятся в папке **backup**.

В области **Хост** выберите папку, в которой будет храниться файл с резервной копией. Кнопкой << скопируйте файл из области **Исполнение** в область **Хост**. В правом нижнем углу экрана появится индикатор хода загрузки. Дождитесь окончания загрузки.

Для того, чтобы восстановить состояние системы из резервной копии, установите связь с контроллером. Перейдите на вкладку **Сервис ПЛК**, далее на внутреннюю вкладку **Системное ПО,** блок **Восстановление.** Для того, чтобы не копировать лог-файлы с резервной копии, снимите переключатель с позиции **Восстановить логи.** Аналогично с позицией **Восстановить MAC's**. Нажмите кнопку ***Выберите файл…*** Откроется диалоговое окно выбора файла на компьютере. Выберите файл, нажмите кнопку ***Открыть***. Далее нажмите кнопку ***Восстановить***.

Создание резервной копии загруженного приложения без перезагрузки контроллера

Для создания в контроллере резервной копии приложения, без перезагрузки контроллера (приложение может находиться в любом из состояний RUN/STOP), установите связь с контроллером и перейдите на вкладку **Оболочка ПЛК**. В командной строке введите команду: **backup** и нажмите клавишу ***Enter.***

Появятся сообщения:

```
backup user
     Creating PLC backup
         user - creating a user backup
```

Далее введите команду на создание резервной копии пользователя: **backup user** и нажмите клавишу ***Enter.***

Появится сообщение с названием созданного файла с расширением *tar.gz (резервной копии пользователя) типа:

```
create backup 'PlcName-TargetId-Version-QNX_650-Date.tar.gz'
```

например:

```
create backup '1_061_251-0x08_R500_61-1.7.2.0(rc0r40318)-QNX_650-
240604095510.tar.gz'
```

Все резервные копии находятся в папке **backup**. Для того, чтобы скопировать файл с резервной копией с контроллера на ПК, перейдите на вкладку **Файлы** (Рисунок 250, аналогично описанию выше).

В области **Исполнение** нажмите кнопку  (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере, перейдите в папку **backup**.

Кнопкой  скопируйте файл с расширением *tar.gz с контроллера на ПК (**Исполнение** в область **Хост**). В правом нижнем углу экрана появится индикатор хода загрузки. Дождитесь окончания загрузки.

Обновление ПО контроллера

i	**ИНФОРМАЦИЯ** Начиная с версии СПО 1.7.0.0, доступна конфигурация ПЛК с операционной системой КПДА (ЗОСРВ «Нейтрино»). Требуется уточнение при оформлении заказа

i	**ИНФОРМАЦИЯ** Начиная с версии 1.7.1.0 реализована возможность подсчета контрольной суммы СПО по алгоритму **SHA-256** и **MD5** с помощью функциональных блоков **FirmwareSHA256** и **FirmwareHash** соответственно (см. библиотеку **PsPlcInfo** в приложении К). В оперативный журнал заносятся записи о начале и окончания процедуры расчета, а также о возникших ошибках: – **starting: OK** – успешный запуск вычисления (hash=00000000000000000000000000000000); – **starting: ERR_CALL_AGAIN** – повторный запуск вычисления (hash=00000000000000000000000000000000); – **error: ERR_CALL_AGAIN** – ошибка вычислений из-за повторного запуска процедуры расчета (hash=00000000000000000000000000000000); – **abort: OK** – вычисление прервано (hash=00000000000000000000000000000000); – **done: OK** – успешное вычисление (hash=3E9517E0604070644750195DA1089149)

i	**ИНФОРМАЦИЯ** При обновлении СПО с 1.7.1.1 (и более ранней версии) на 1.7.1.2 (и более поздней версии), происходит деактивация значения функции автозапуска приложения (снимается флажок ☑ в строке **Autostart application**) на контроллере серии R400. Для проверки состояния, активируйте сервисный режим ПЛК на встроенном сенсорном дисплее (см. раздел «Сервисный режим контроллера серии R400»). В остальных случаях значение функции автозапуска приложения при обновлении СПО не меняется

Особенности обновления СПО до 1.7.0.0 и до 1.7.1.0

До версии 1.7.0.0 (переход с Epsilon LD на Astra.IDE)

Для обновления СПО на ПЛК до версии 1.7.0.0 с более ранних версий выполните следующие действия:

– установите пакет 1.6.5.9 в Epsilon LD;
– обновите СПО ПЛК до версии 1.6.5.9;
– подключитесь к контроллеру средой Epsilon LD;

– обновите системное ПО контроллера через сервис, либо через файловую систему, загрузив с ПК на ПЛК файл RegulFw.fwe (путь, например, C:\Program Files\AstraRegul\Astra.IDE 64 1.7.0.0\Astra.IDE\Firmwares\RegulFw 1.7.0.x);

– перезагрузите контроллер. Дождитесь, когда основные индикаторы погаснут и контроллер загрузится в штатном режиме: издаст звуковой сигнал и начнет мигать индикатор PF/PROGF. Подключитесь к ПЛК заново средой Astra.IDE.

С версий 1.7.0.x до версии 1.7.1.0

Для обновления СПО на ПЛК до версии 1.7.1.0 с более ранних версий (1.7.0.x) выполните следующие действия:

– установите два пакета обновления версии 1.7.0.x и 1.7.1.0 (выберите в основном меню Astra IDE **Инструменты ⇨Менеджер пакетов…⇨Откроется окно Менеджер**);

– в проекте должна использоваться версия устройства 3.5.17.30 модуля ЦП. Текущую версию устройства можно определить, перейдя в меню **Проект ⇨ Среда проекта ⇨ Версии устройств** (Рисунок 251);

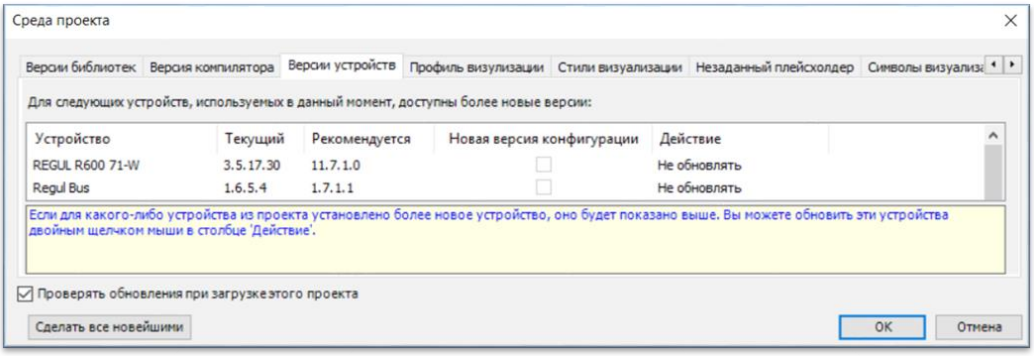


Рисунок 251 – Среда проекта. Версии устройств

Изменить версию устройства можно через обновление: в окне дерева устройств поместите курсор на название контроллера REGUL…, правой кнопкой мыши вызовите контекстное меню, выберите пункт Обновить устройство… (Рисунок 252).

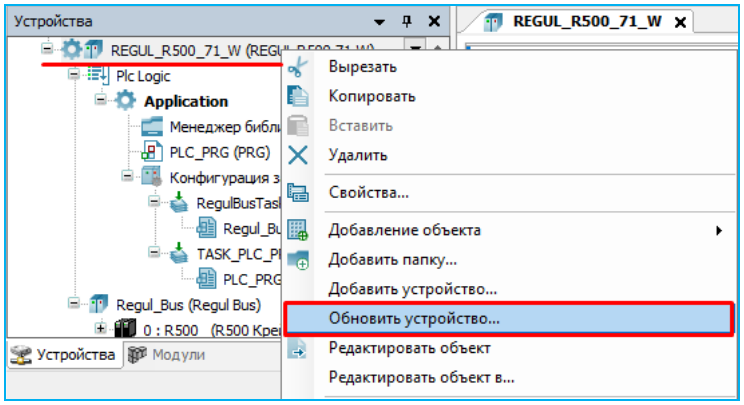


Рисунок 252 – Обновить устройство

Откроется окно **Обновить устройство**, где установите флажок в поле **Отображать все версии (только для экспертов).** В списке модулей будут отображены все модули всех версий, которые поддерживаются в системе. Выберите версию устройства **3.5.17.30**;

- подключитесь к контроллеру средой Astra.IDE;
- для обновления СПО модуля ЦП перейдите на вкладку **Сервис ПЛК**, далее на внутреннюю вкладку **Системное ПО**. Нажмите кнопку ***Запросить версию СПО***. В поле **Версия СПО**: отобразится номер текущей версии. В блоке **Обновление** в поле **Доступные СПО**: отобразятся все возможные файлы с системным ПО. Выберите из списка версию СПО 1.7.1. Нажмите кнопку ***Обновить;***
- подтвердите перезагрузку контроллера. Дождитесь, когда основные индикаторы погаснут и контроллер загрузится в штатном режиме: издаст звуковой сигнал и начнет мигать индикатор PF/PROGF;
- обновите версию устройства в проекте на версию **11.7.1.0**;
- подключитесь заново к ПЛК средой разработки Astra.IDE.

Обновление пакета

Для установки пакета обновления *Prosoft-Systems_Regul_<версия>.package* для Epsilon LD выберите в основном меню **Инструменты** ⇨ **Менеджер пакетов…** Откроется окно **Менеджер пакетов** (Рисунок 253).

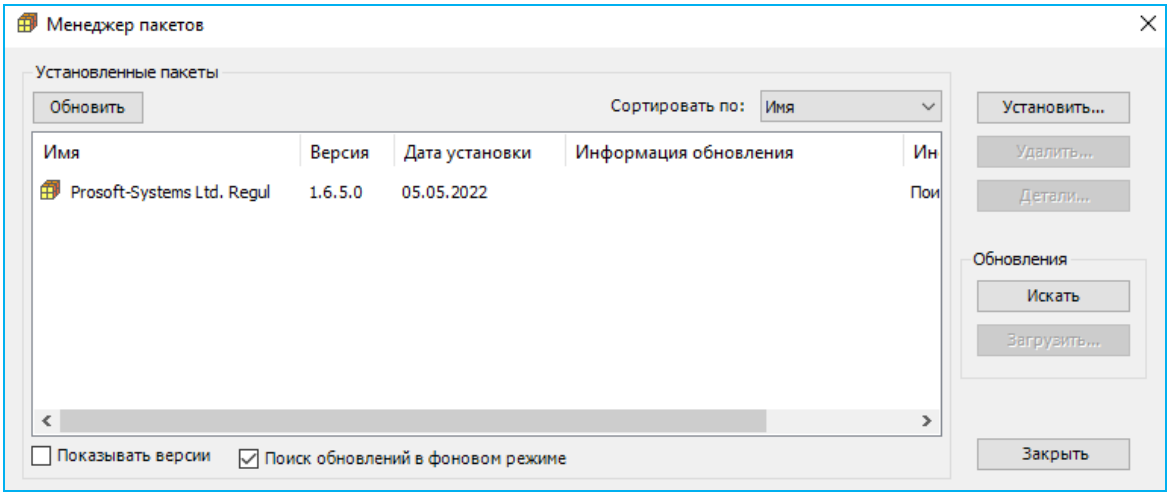


Рисунок 253 – Диалоговое окно «Менеджер пакетов»

Нажмите кнопку ***Обновить***. В блоке **Установленные пакеты:** отобразится список всех пакетов, установленных на данном компьютере. Пакеты могут быть отсортированы по дате установки.

Для установки пакета, отсутствующего в списке, нажмите кнопку ***Установить…*** Откроется окно для выбора файла пакета. Найдите нужный файл с расширением **.package*. Нажмите кнопку ***Открыть***. Подождите несколько секунд.

Откроется окно мастера установки (Рисунок 254).

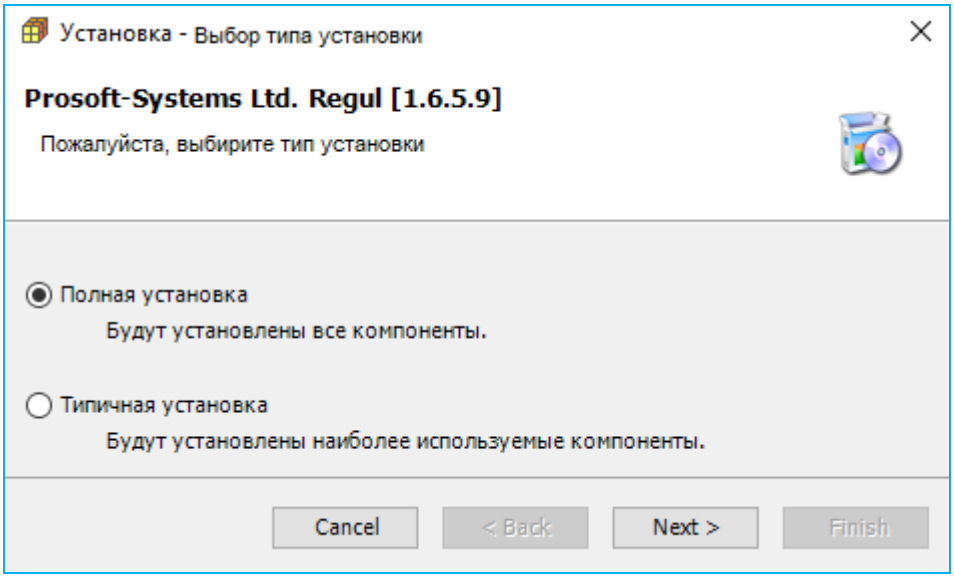


Рисунок 254 – Окно мастера установки обновления ПО

Нажмите кнопку ***Next***. Программа предложит выбрать все версии среды разработки Epsilon LD, которые должны быть обновлены устанавливаемым пакетом. Нажмите кнопку ***Next***.

По завершении установки появляется окно с сообщением: «*Пакет успешно установлен. Нажмите Завершить (Finish), чтобы выйти из установщика, или Далее (Next), чтобы просмотреть результаты. Для вступления в силу некоторых компонентов необходимо перезапустить приложение*».

Нажмите кнопку ***Next***. Появится отчет об установке, в котором можно ознакомиться со списком установленных компонентов (Рисунок 255). Нажмите кнопку ***Finish***.

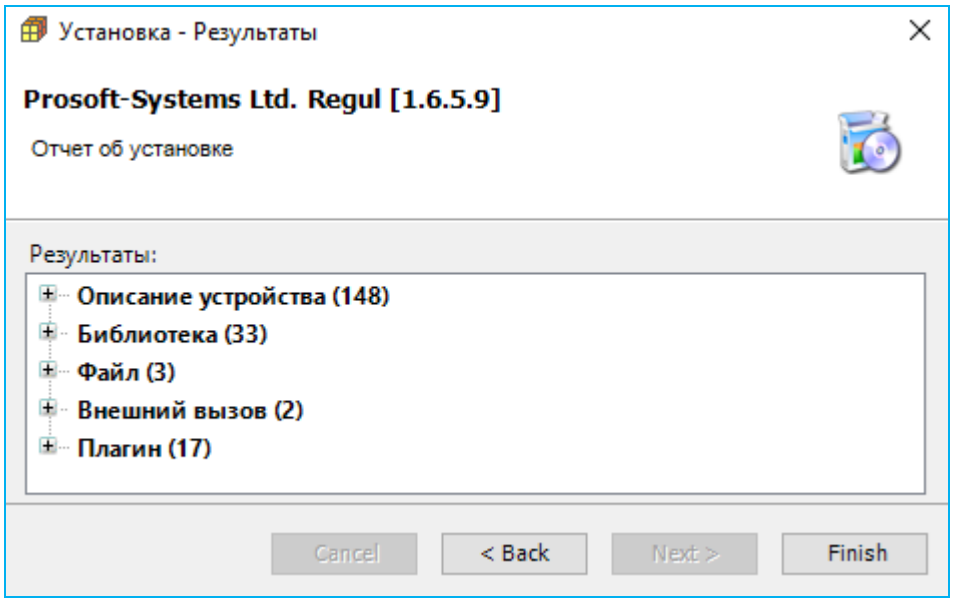


Рисунок 255 – Отчет об установке

В окне **Менеджер пакетов** нажмите кнопку ***Закрыть***. Перезапустите программу Epsilon LD.

Компоненты для Astra.IDE поставляются в виде пакета – файла с именем типа: *RegLab _Ltd_Regul <версия>.package*, например, *RegLab _Ltd_Regul_1.7.0.3*.

Установка пакета обновления Regul Package производится по аналогии со средой Epsilon LD (см. описание выше)

Пакет Regul Package содержит:

- описания программных и аппаратных модулей в составе контроллера;
- набор библиотек и драйверов;
- файл системного ПО контроллера;
- набор подключаемых плагинов для среды разработки Astra.IDE.

Стандартное обновление системного программного обеспечения на ПЛК

При обновлении системного ПО произойдет сброс настроек к заводскому состоянию. Сохранить имеющиеся настройки IP-адресов можно через Сканер сети, функция импорта/экспорта (см. подраздел «Сканер сети. Настройка IP-адресов»).

	**ИНФОРМАЦИЯ** Для получения информации о типе ОС (QNX или KPDA) и настройках ПЛК, выберите в главной вкладке параметров устройства внутреннюю вкладку **Оболочка ПЛК.** В командной строке введите команду plcinfo и нажмите клавишу **Enter**. Появится список с обобщенной информацией о контроллере, в том числе и о типе ОС (например: OS Type = QNX)

Для обновления системного ПО через сервис контроллера перейдите на вкладку **Сервис ПЛК**, далее на внутреннюю вкладку **Системное ПО**. Нажмите кнопку ***Запросить версию СПО***. В поле **Версия СПО:** отобразится номер текущей версии (Рисунки 256, 257).

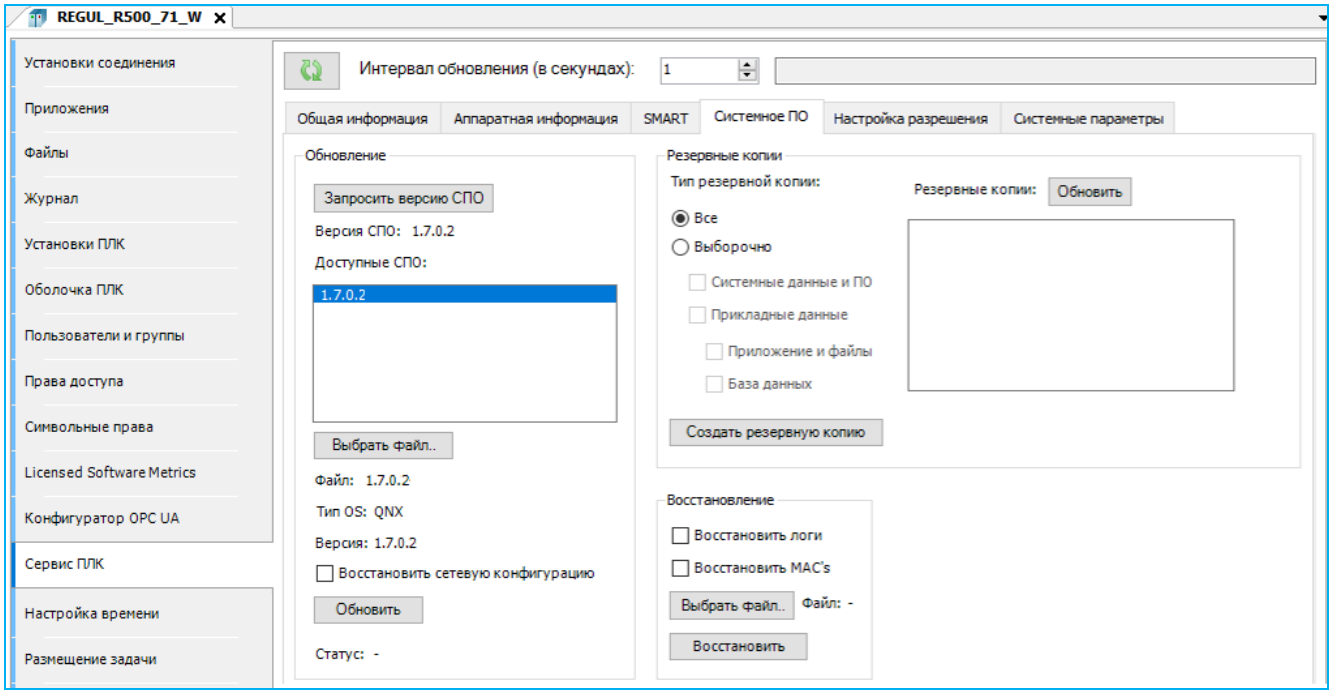


Рисунок 256 – Обновление системного ПО (OS QNX)

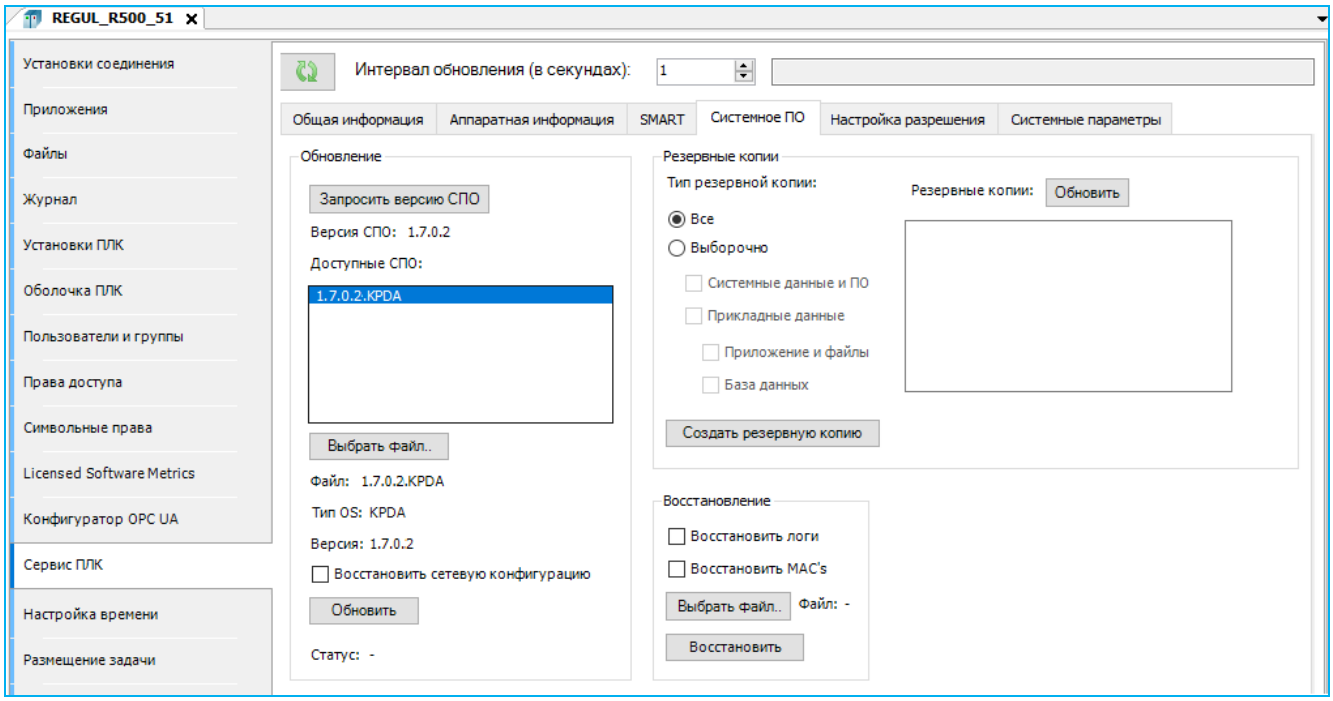


Рисунок 257 – Обновление системного ПО (OS KPDA)

В блоке **Обновление** в поле **Доступные СПО:** отобразятся все возможные файлы с системным ПО, установленные вместе с пакетами поддержки и находящиеся в папке *AstraRegul*. Если в списке есть нужная версия СПО, то выберите эту строку.

В случае, когда файл с системным ПО получен от производителя отдельно, а не в составе пакета поддержки, и находится не в папке установки пакетов *AstraRegul*, а в каком-либо другом месте на компьютере или на переносном носителе, его можно выбрать с помощью функции **Выбрать файл**. Нажмите кнопку ***Выбрать файл…*** Откроется диалоговое окно для выбора

файла системного ПО. Найдите файл *RegulFw.fwe*, либо, для установки версии ОС KPDA - *RegulFw_KPDA.fwe*. В случае, если файл выбран не верно и СПО не предназначено для целевого ПЛК, то появится предупреждающее сообщение (Рисунок 258).

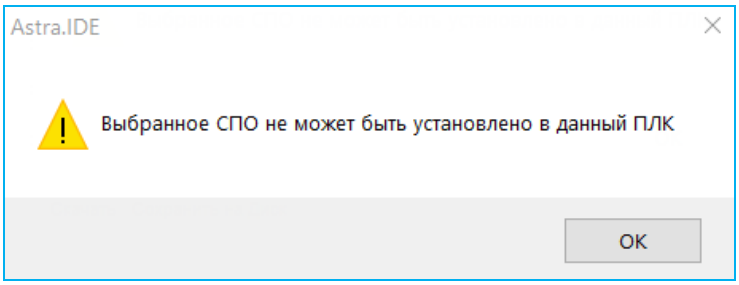


Рисунок 258 – Предупреждение о несоответствии выбранного СПО

Можно воспользоваться опцией восстановления сетевой конфигурации ПЛК (network.cfg) после обновления. Для использования опции, в блоке **Обновление**, установите флажок в поле **Восстановить сетевую конфигурацию**.

Нажмите кнопку ***Обновить***.

Начнется загрузка файла на контроллер, в правом нижнем углу экрана появится индикатор хода процесса загрузки (Рисунок 259), в поле **Статус** появится сообщение: «*Загрузка файла СПО…*».

Рисунок 259 – Индикатор хода процесса загрузки файла на контроллер

При успешной загрузке, в поле **Статус** появится сообщение: «*Файл СПО загружен*», откроется диалоговое окно с сообщением: «*Для обновления СПО контроллера необходима перезагрузка. Перезагрузить контроллер?*» (Рисунок 260).

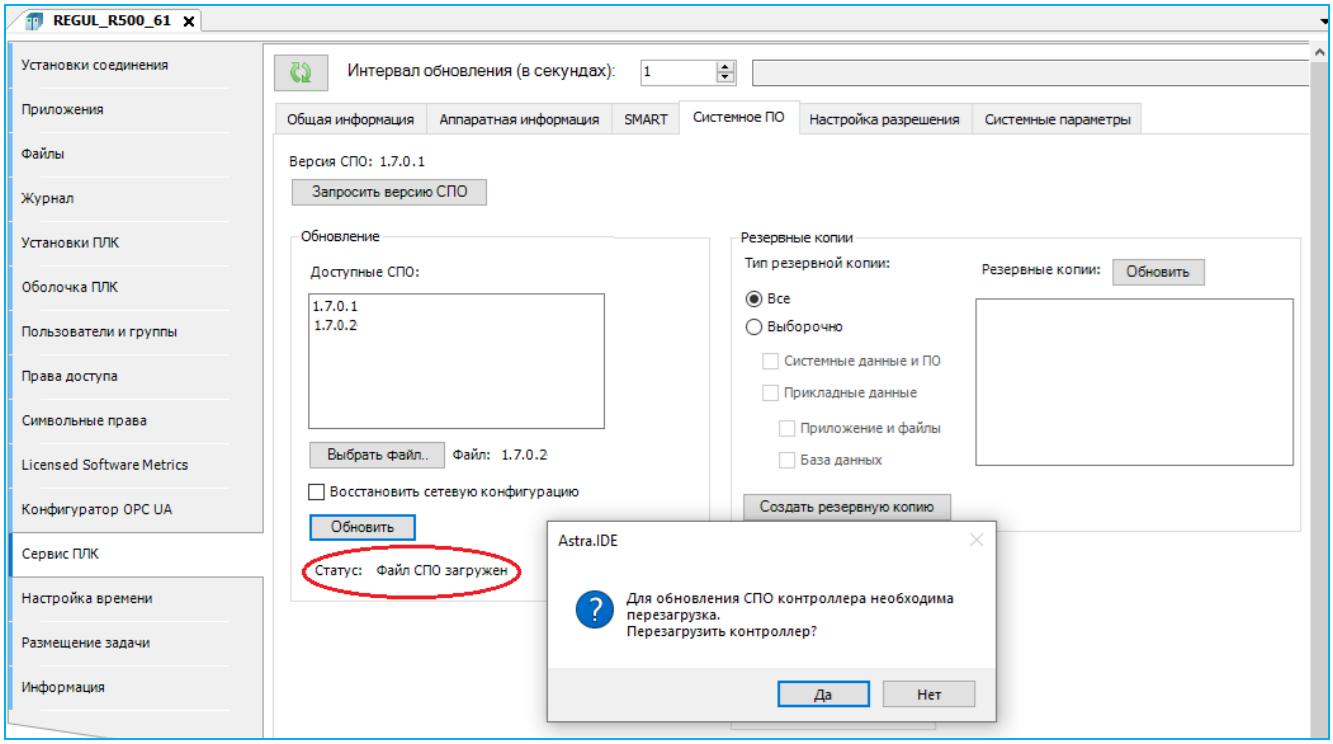


Рисунок 260 – Запрос на перезагрузку контроллер

Нажмите кнопку ***Да***. Откроется информационное окно (Рисунок 261).

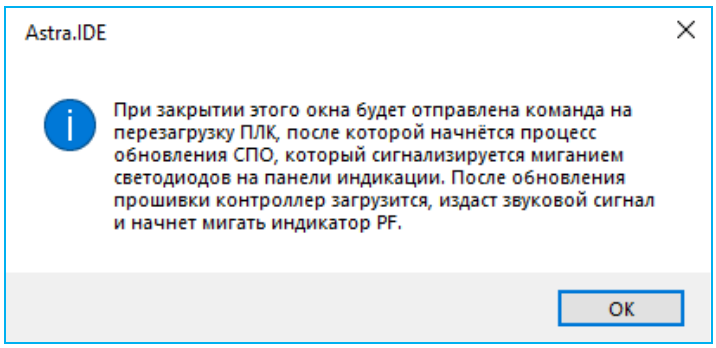


Рисунок 261 – Информационное окно

Нажмите кнопку ***ОК***. В поле **Статус** появится сообщение: «*Контроллер перезагружается…*».

Визуально, процесс обновления сопровождается одновременным миганием основных индикаторов, расположенных на передней панели модуля ЦП:

- серия R500: индикаторы RUN, RDD и LDx,
- серия R200: индикаторы RUN и LDx,
- серия R600: индикаторы RUN, RDD и LEDx.

⚠	**ВНИМАНИЕ!** На выполнение данной процедуры потребуется определенный промежуток времени, поэтому, ни в коем случае не отключайте питание! Дождитесь, когда основные индикаторы погаснут и контроллер загрузится в штатном режиме: издаст звуковой сигнал и начнет мигать индикатор PF/PROGF. В случае возникновения ошибки при обновлении, устройство издаст пятикратный звуковой сигнал

Далее воспользуйтесь сканером сети для подключения к контроллеру вновь. Настроечные параметры будут сброшены: имя контроллера установлено в localhost, а IP-адреса всех сетевых интерфейсов сброшены на 0.0.0.0. Задайте заново настройки (см. подраздел «Сканер сети. Настройка IP-адресов»). В случае, если была отмечена опция восстановления сетевой конфигурации ПЛК, сетевые настройки будут автоматически восстановлены.

Для обновления системного ПО через файловую систему контроллера перейдите на вкладку **Файлы**. В области **Хост** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на компьютере. Найдите файл *RegulFw.fwe*. Такие файлы находятся в папке пакета (Package), который, в свою очередь, обычно устанавливается в папку AstraRegul\…\ Astra.IDE\..., например,

C:\Program Files\AstraRegul\Astra.IDE 64 1.7.0.0\Astra.IDE\Firmwares\RegulFw 1.7.x.x

В области **Исполнение** нажмите кнопку (***Обновить***). В окне отобразится дерево файлов, имеющихся на контроллере. Зайдите в папку **update**.

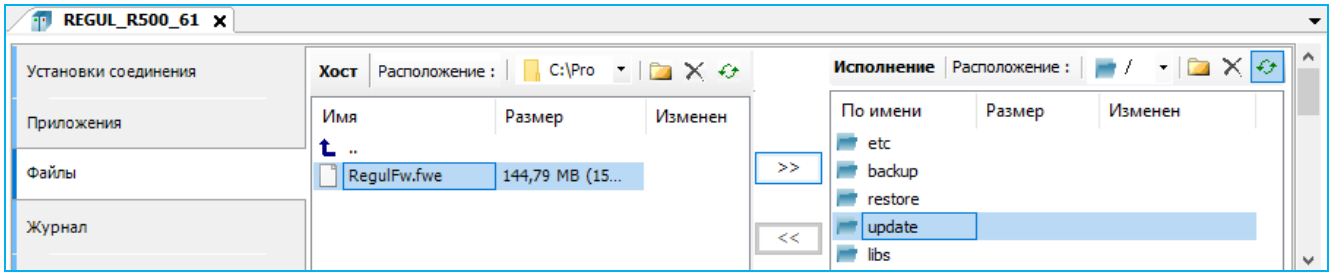


Рисунок 262– Копирование файла системного ПО на контроллер (OS QNX)

Кнопкой >> скопируйте файл RegulFw.fwe с ПК на контроллер в папку **update** (из **Хост** в **Исполнение**). Для версии OS KPDA необходимо выбрать файл *RegulFw_KPDA.fwe*. Начнется загрузка файла системного ПО на контроллер, в правом нижнем углу экрана появится индикатор хода загрузки. Дождитесь окончания загрузки.

После обновления системного ПО через файловую систему потребуется перезагрузить контроллер путем выключения/включения питания либо командой *reboot* на вкладке **Оболочка ПЛК**.

Обновление программного обеспечения модулей ввода/вывода

Для обновления СПО модуля ввода/вывода с помощью специального приложения, зайдите на сайт Производителя: https://reglab.ru/, перейдите на вкладку соответствующего контроллера **RX00** ⇨ выберите необходимый модуль ввода/вывода ⇨ **СКАЧАТЬ** и скачайте пакет обновления.

Инструкция по работе с приложением находится на вкладке **ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ⇨ СЕРВИСНОЕ ПО**.

Начиная с версии СПО модулей ввода/вывода 1.0.25.0 доступно обновление программного обеспечения с помощью среды Astra.IDE. Для этого подключитесь к модулю в режиме онлайн и перейдите на вкладку **Сервисы**. Далее нажмите кнопку ***Обзор***. Откроется диалоговое окно для выбора файла СПО. Найдите необходимый файл. Нажмите кнопку ***Начать*** (Рисунок 263)**.**

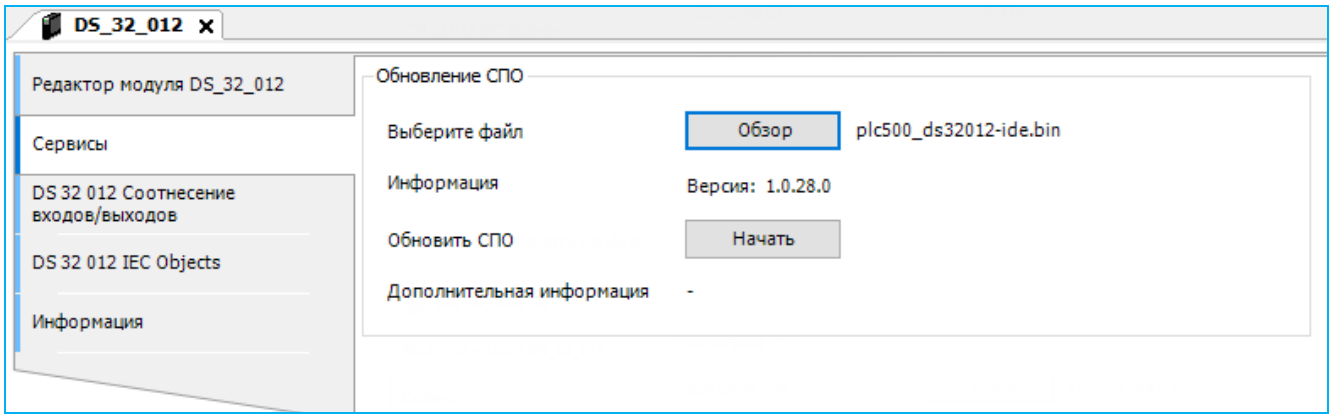


Рисунок 263 – Обновление СПО модуля ввода/вывода

В случае, если файл обновления СПО будет несовместим с выбранным модулем (не соответствует тип модуля и/или модуль с версией СПО до 1.0.25.0), то кнопка ***Начать*** будет не активна и в поле ***Дополнительная информация*** появится «Выбранный файл СПО предназначен для другого устройства», либо «Выбранный файл СПО предназначен для другого устройства. Функция обновления поддерживается на устройствах версией СПО 1.0.25.0 и выше» (Рисунок 264).

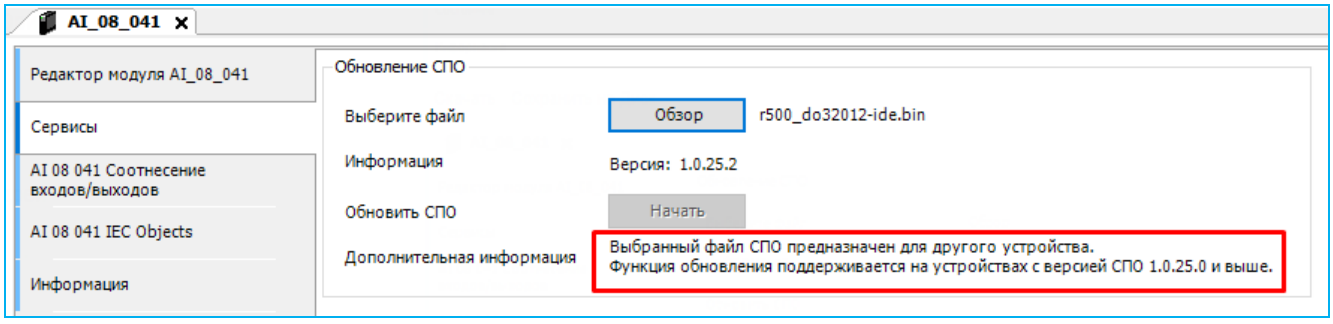


Рисунок 264 – Пример несовместимости СПО модуля ввода/вывода

Об обратной совместимости

Производитель постоянно занимается доработкой контроллеров – выявлением и устранением недостатков, улучшением программной и аппаратной части в соответствии с собственными прогрессивными технологиями и новыми течениями в отрасли в целом. Это неизбежно приводит к наличию разных версий встроенного ПО на модулях контроллера разных выпусков. Чтобы в таких условиях обеспечить полноценное функционирование контроллеров у пользователя, производитель в обязательном порядке прорабатывает вопрос обратной совместимости. Это возможность совместной работы модулей ввода/вывода, имеющих на борту программное обеспечение новейших версий, тогда как СПО модуля центрального процессора относится к более ранней версии; и наоборот – взаимодействие предыдущих версий ПО модулей ввода/вывода с новым СПО модуля центрального процессора.

Каждый пакет обновлений содержит новые файлы-описания устройств. Данные файлы-описания позволяет работать с устройствами, с более ранними версиями СПО. Но если версии СПО, установленные на устройстве, выпущены достаточно давно или при выпуске нового файла-описания, был изменен список переменных, то такие устройства не смогут работать с текущим файлом описания. Информация о таком конфликте автоматически появляется в журнале событий контроллера (log) (см. подраздел «Журнал событий») с описанием ошибки: «*Firmware version is outdated*».

Есть два варианта действий в данной ситуации:

– обновить СПО модуля ввода/вывода (рекомендуется);
– изменить версию файл-описания модуля в среде разработки, если обновление СПО модуля ввода/вывода невозможно/не приемлемо.

Для выбора другой версии файла-описания необходимо выяснить, СПО какой версии установлено на модуле ввода/вывода, после чего сопоставить модулю соответствующий файл-описание. Для этого:

– установите соединение с контроллером, выполните логин, затем старт проекта. Откройте редактор модуля ввода/вывода. В блоке **Общие параметры устройства** в поле **Текущая версия СПО** отображается текущая версия СПО модуля (Рисунок 265). Выпишите этот номер для себя (номер версии СПО 1.0.14.4). Аналогично выпишите для себя значение поля Минимально допустимая версия СПО. Закройте редактор модуля.

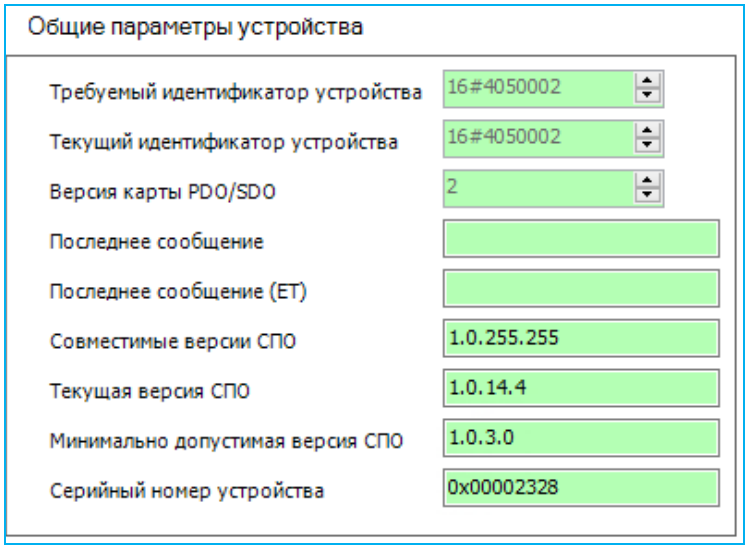


Рисунок 265 – Информация о модуле

– выполните обновление устройства: поместите курсор на название модуля в дереве устройств, правой кнопкой мыши вызовите контекстное меню, выберите пункт **Обновить устройство…** Откроется окно **Обновить устройство**, где установите флажки в поле **Отображать все версии (для экспертов)**. В списке модулей будут отображены все модули всех версий, которые поддерживаются в системе (Рисунок 266);

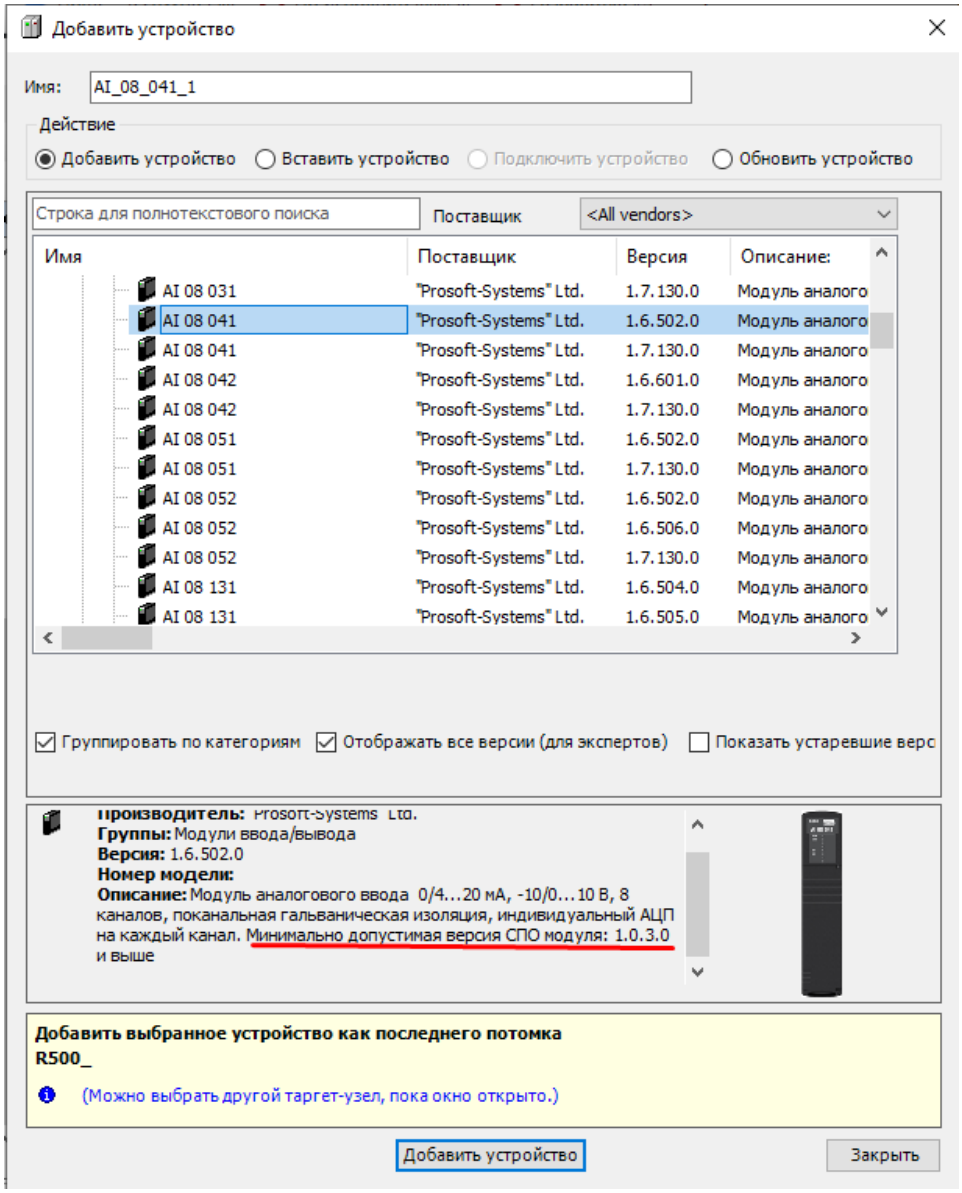


Рисунок 266 – Список всех поддерживаемых версий СПО модуля

– выбирайте в списке нужный модуль и просматривайте раздел **Информация**, где указана требуемая версия СПО модуля. Так, например, модулю с версией СПО 1.0.14.4 соответствует файл-описание версии: 1.6.502.0 с требованием версии СПО модуля 1.0.3.0 и выше (если подходит несколько версий, выбирайте последнюю, то есть самую новую);

– выберите подходящий модуль. Нажмите кнопку ***Обновить устройство***.

Проверка файловой системы на целостность

С целью диагностики повреждений файловой системы ПЛК реализована возможность проверки системных файлов по контрольной сумме, т.е. сравнение его текущей контрольной суммы с ожидаемой. Проверка происходит на этапе включения ПЛК, что может привести к увеличению времени его готовности к работе.

Проверка файлов включается:

- автоматически:
  - после каждого обновления СПО ПЛК и его сброса до заводских настроек;
  - после каждой неудачной проверки;
- вручную пользователем через вкладку **Оболочка ПЛК**.

Проверка файлов выключается:

- автоматически при успешном прохождении проверки;
- вручную пользователем через вкладку **Оболочка ПЛК**.

Для ручного управления необходимо перейти на вкладку **Оболочка ПЛК**. Нажать кнопку […] в нижнем правом углу. Откроется диалоговое окно с перечнем команд. Выбрать команду **firmware** и нажать кнопку ***Вставить***, далее клавишу ***Enter.***

В окне отобразится информация с описанием возможных параметров команды:

```
firmware check [on|off]
Checking the firmware file system status at boot
check - get current status of the file system
check on - enables file system check at boot
check off - disables file system check at boot
```

Необходимо добавить команду на включение/выключение в командную строку и нажать клавишу ***Enter***. В зависимости от выбранной команды будет выведен результат выполнения:

- команда **firmware check** – результат: «*file system check at boot is disabled/enabled*» (проверка файловой системы при загрузке отключена/включена);
- команда **firmware check on** – результат: «file system check at boot is enabled» (проверка файловой системы при загрузке включена). При перезагрузке ПЛК в лог-файле /mnt/user/archive/logs/logger/system.log появится информация о проверке контрольной суммы во время последней загрузки. В случае несовпадения контрольных сумм журналируется ошибка по конкретному файлу и создается файл etc/checkFW;
- команда **firmware check off** – результат: «*file system check at boot is disabled»* (проверка файловой системы при загрузке отключена).

Сервисный режим контроллера

Сервисный режим контроллера предназначен для случаев, когда нет возможности подключения контроллера к Astra.IDE.

Сервисный режим контроллера позволяет осуществлять следующие операции:

- сброс до заводского состояния (***Factory reset***);
- создание резервной копии пользовательских файлов (***User backup***);

- создание резервной копии системных файлов (***System backup***);
- создание полной резервной копии (***All backup***).

Сервисный режим на контроллерах серии R200, R500, R600

С помощью светодиодных индикаторов LDx/LEDx и переключателей RUN/STOP, Key, расположенных на передней панели, пользователь может активировать сервисный режим ПЛК.

Для активации сервисного режима ПЛК необходимо выполнить следующие действия:

- выключить питание ПЛК;
- выключить автозапуск прикладной программы (повернуть ключ KEY в положение *I* и переключатель RUN/STOP в положение *STOP*);
- подать питание на ПЛК;
- когда индикаторы LD1, LD2, LD3 (на контроллерах серии R500, R200) или LED1, LED2, LED3 (на контроллере серии R600) начнут поочередно мигать, необходимо перевести переключатель RUN/STOP в положение *RUN*, что приведет к переходу в сервисный режим. Режим прекратит свою работу, если не перевести переключатель RUN/STOP в течении пяти секунд (пока мигают индикаторы);
- одновременное включение и быстрое мигание следующих индикаторов укажет на успешную активацию сервисного режима:
  - серия R500: индикаторы RUN, RDD, HF и PF,
  - серия R200: индикаторы RUN, HF и PF,
  - серия R600: индикаторы RUN, RDD, HARDF и PROGF.

Работа в сервисном режиме

Выбор и переход по пунктам меню производится путем переключения положений *STOP/RUN*. Переход по пунктам меню осуществляется по кругу (после последнего – переход к первому).

Установите переключатель RUN/STOP в положение *STOP*, затем верните в положение *RUN*. Загорится индикатор LD1/LED1 – выбран первый пункт сервисного меню (сброс до заводского состояния). Для перехода к следующему пункту меню верните переключатель RUN/STOP в положение *STOP*, затем снова установите в положение *RUN*. Загорится индикатор LD2/LED2, показывающий, что выбран второй пункт меню. Для выбора следующих пунктов сервисного меню повторите действие *STOP*↔ *RUN*. Если задержаться на положении переключателя *STOP* больше двух секунд, произойдет возврат к первому пункту меню.

Ниже приведена соответствующая индикация пунктам меню сервисного режима.

Пункт меню сервисного режима	Состояние индикатора LD1/LED1	Состояние индикатора LD2/LED2	Состояние индикатора LD3/LED3	Состояние индикатора LD4/LED4 (резерв)
1 - сброс до заводского состояния	✸	–	–	–
2 - создание резервной копии пользовательских файлов	–	✸	–	–
3 - создание резервной копии системных файлов	✸	✸	–	–
4 - создание полной резервной копии	–	–	✸	–

Для запуска выбранной команды меню поверните ключ KEY в положение *II*. Начнет мигать индикатор соответствующего пункта меню LDx/LEDx и будет мигать зеленым в течении двух секунд. Если в это время повернуть ключ KEY обратно в положение *I*, произойдет отмена запуска команды. Если команду не отменили, через две секунды она запустится, индикатор LDx/LEDx станет постоянно гореть зеленым.

Для выхода из любого пункта меню сервисного режима переведите переключатель RUN/STOP в положение *STOP* и поверните ключ KEY в положение *I*.

Алгоритм сброса к заводским настройкам

Сброс к заводским настройкам осуществляется активацией первого пункта сервисного режима. Для этого требуется:

1. Выключить контроллер.

2. Перевести KEY в положение *I*.

3. Перевести RUN/STOP в положение *STOP*.

4. Включить контроллер.

5. Дождаться, когда поочередно будут мигать индикаторы LD1, LD2, LD3, и перевести RUN/STOP в положение *RUN*, контроллер войдет в сервисный режим (индикаторы RUN, HF, PF начнут мигать).

6. Переключатель RUN/STOP установить в положение *STOP*, затем вернуть в положение *RUN*, загорится индикатор LD1. Переключить ключ KEY в положение *II* – запуск выбранной команды, индикатор LD1 начнет мигать.

7. Дождаться, когда контроллер загрузится, издаст звуковой сигнал и начнет мигать индикатор PF.

8. Проверить сканером сети, что имя контроллера установлено в localhost, а IP-адреса всех сетевых интерфейсов сброшены на 0.0.0.0.

Сервисный режим на контроллере серии R400

Пользователь может активировать сервисный режим ПЛК на встроенном сенсорном дисплее, для этого необходимо выполнить следующие действия:

– включить питание ПЛК. На сенсорном дисплее откроется окно с индикатором хода загрузки, который будет заполняться в течении 5 секунд (Рисунок 267);

Рисунок 267 – Индикатор хода загрузки

– пока индикатор не заполнился до конца, коснуться сенсорной панели в любой области дисплея для открытия сервисного меню, иначе окно будет закрыто. Нажать кнопку ***Service menu*** для открытия окна со списком сервисных команд. Для выхода из сервисного меню нажать кнопку ***Exit*** (Рисунок 268). Для автозапуска приложения установить флажок в строке **Autostart application.**

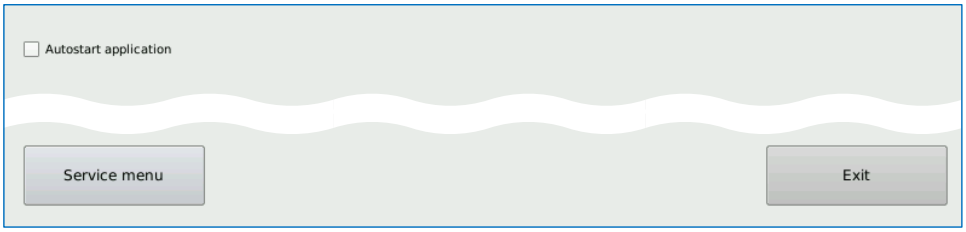


Рисунок 268 – Сервисное меню

– список сервисных команд представлен набором кнопок, соответствующая сервисная команда будет выполнена при нажатии одной из кнопок (Рисунок 269);

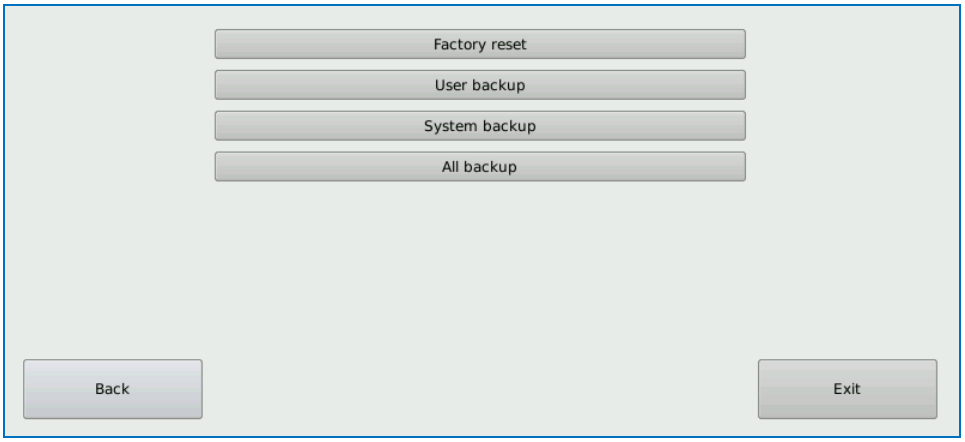


Рисунок 269 – Окно сервисных команд

Отключение входа в сервисный режим на время загрузки контроллера

Предусмотрена возможность отключения входа в сервисный режим на время загрузки контроллера.

Для этого необходимо добавить значение параметра, выполнив следующие действия (Рисунок 270):

– перейдите на вкладку Сервис ПЛК ⇨ Системные параметры ⇨ Экспертный режим. Нажмите на кнопку (*Обновить*);

– выберите название каталога конфигурационный файл **etc/plc.cfg.** Добавьте секцию [Startup] с параметром *ServiceMode* равным Disable;

– нажмите кнопку **Сохранить**.

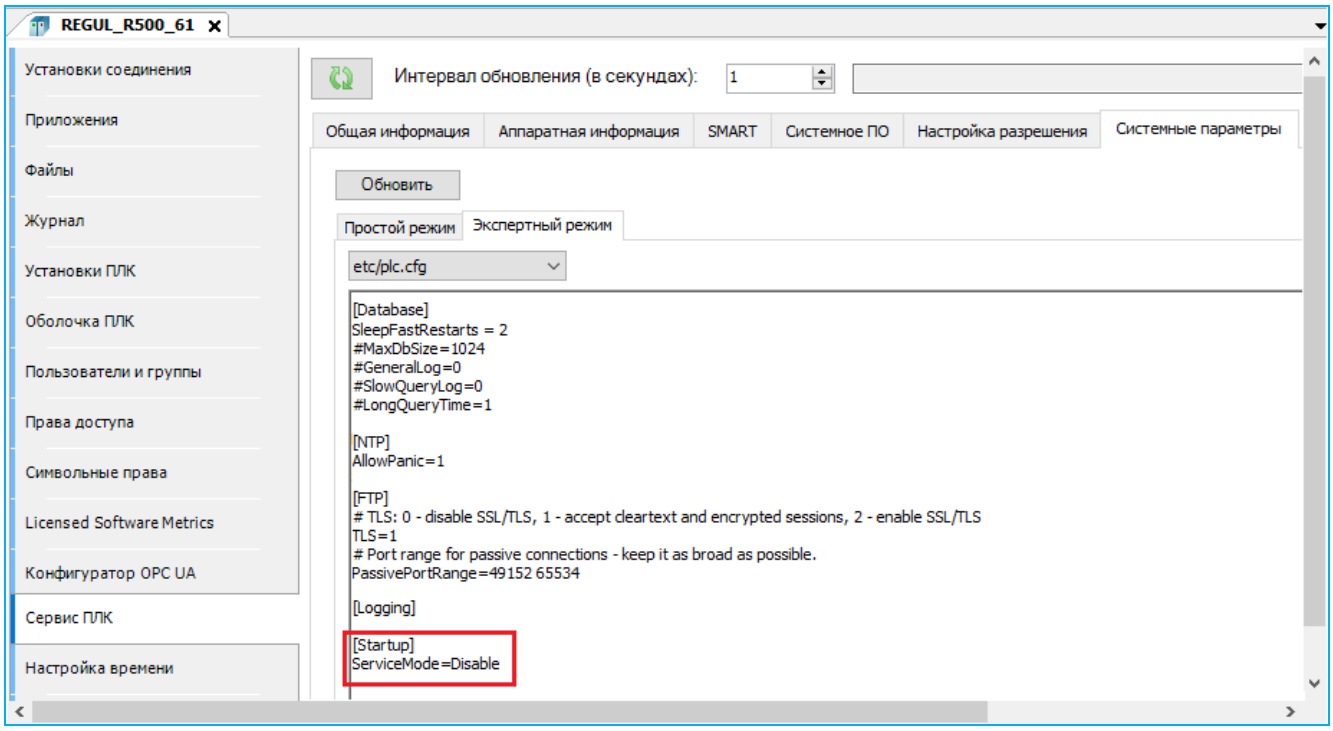


Рисунок 270 – Отключение входа в сервисный режим на время загрузки контроллера

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

Установка пароля или сертификата на файл проекта

⚠	**ВНИМАНИЕ!** По умолчанию канал связи с ПЛК не шифруется. Подробное описание настройки зашифрованного соединения приведено в пункте «Изменение политики соединения с ПЛК»

Последовательность действий:

1. Выберите в основном меню команду **Проект (Project)** ⇨ **Установки проекта (Project Settings)…**В появившемся окне **Установки проекта** выберите категорию **Безопасность (Security)** (Рисунок 271).

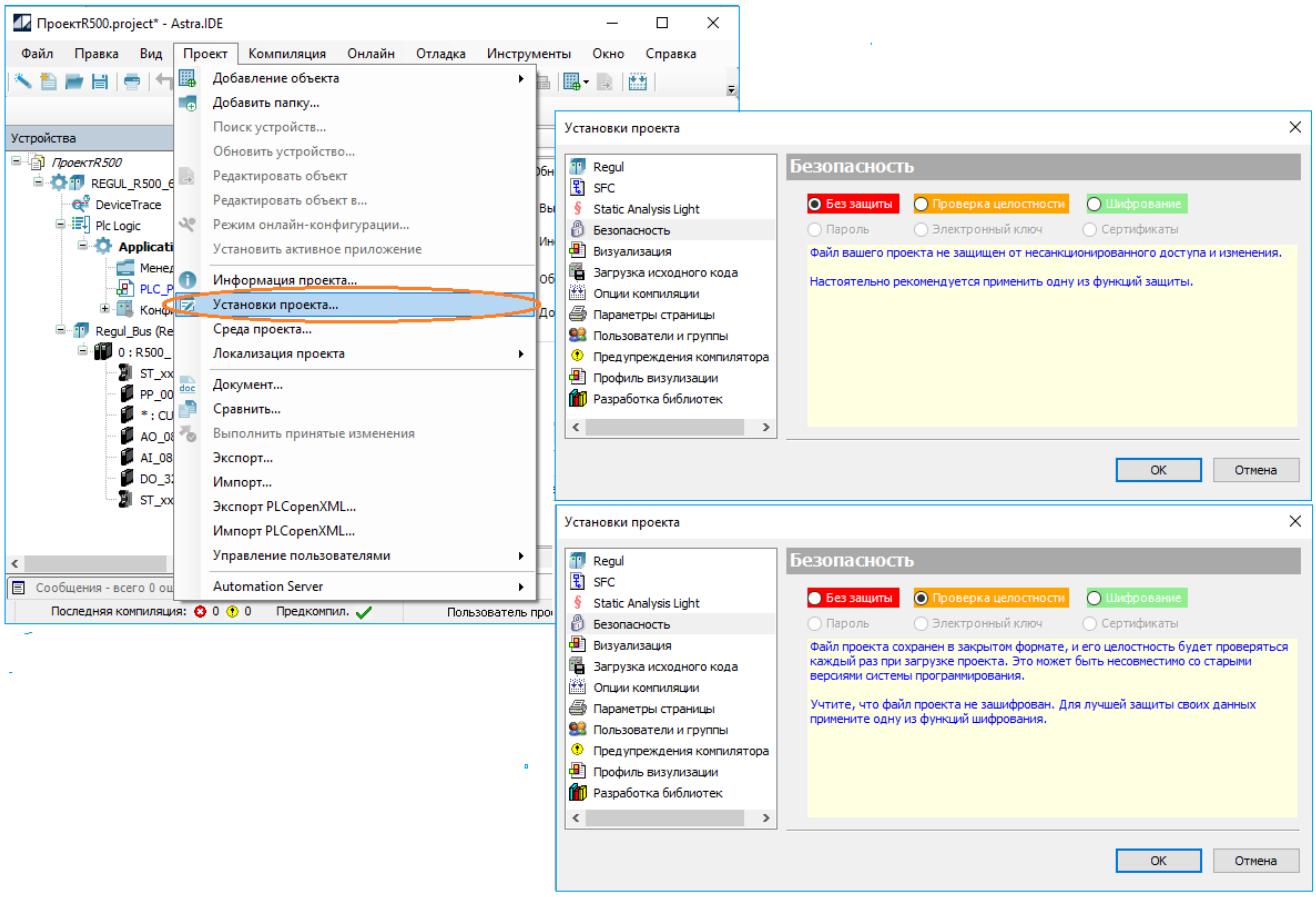


Рисунок 271 – Диалоговое окно установки защиты файл проекта

2. Выберите тип защиты и установите точку в необходимой строке: **Без защиты** (на красном фоне), **Проверка целостности** (на оранжевом фоне) или **Шифрование** (на зеленом фоне).

3. Для лучшей защиты своих данных выберите одну из функций **Шифрования** и установите точку в необходимой строке:**Пароль** или **Сертификаты (**Рисунок 272**)**.

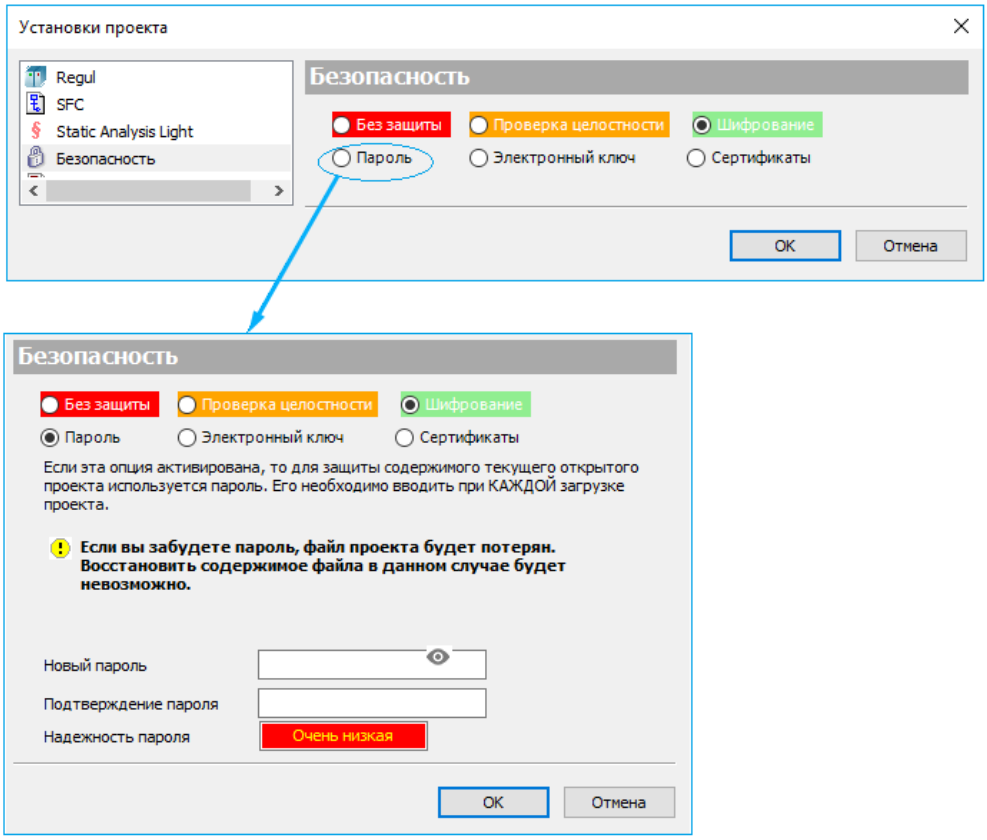


Рисунок 272 – Диалоговые окна активации опций защиты файл проекта (пароль)

4. При использовании пароля, заполните поля **Новый пароль (New password)** желаемый пароль и подтвердите введённый пароль **Подтверждение пароля (Confirm new password).** Заполнив поля нажмите кнопку **ОК**.

5. Сохраните проект (**Файл (File)** ⇨ **Сохранить** (**Save**)). Теперь при каждой последующей попытке запустить проект, на экране будет всплывать диалоговое окно с требованием ввести пароль (Рисунок 273).

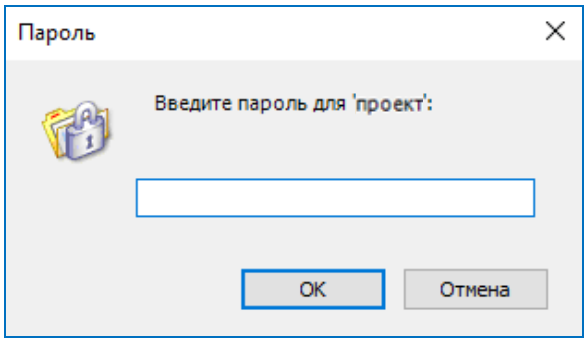


Рисунок 273 – Диалоговое окно ввода пароля при последующем перезапуске проекта

При желании можно сменить пароль, при этом дополнительно к п.5 появится поле **Текущий пароль (Current password**). Заполнив поля нажмите кнопку **ОК**. Для удаления пароля, уберите точку со строки **Пароль (Password),** нажмите кнопку **ОК** и сохраните проект.

Посредством сертификата вы защищаете проект, приложение или онлайн-код. Все доступные сертификаты для шифрования проекта расположены в хранилище сертификатов Windows. Нажмите на кнопку  и откроется окно **Выбор сертификата** (Рисунок 274). В табличном виде будут отображены свойства сертификатов: *Выдан для, Кем выдан, Действителен с, Действителен до, Отпечаток* (печать SHA1).

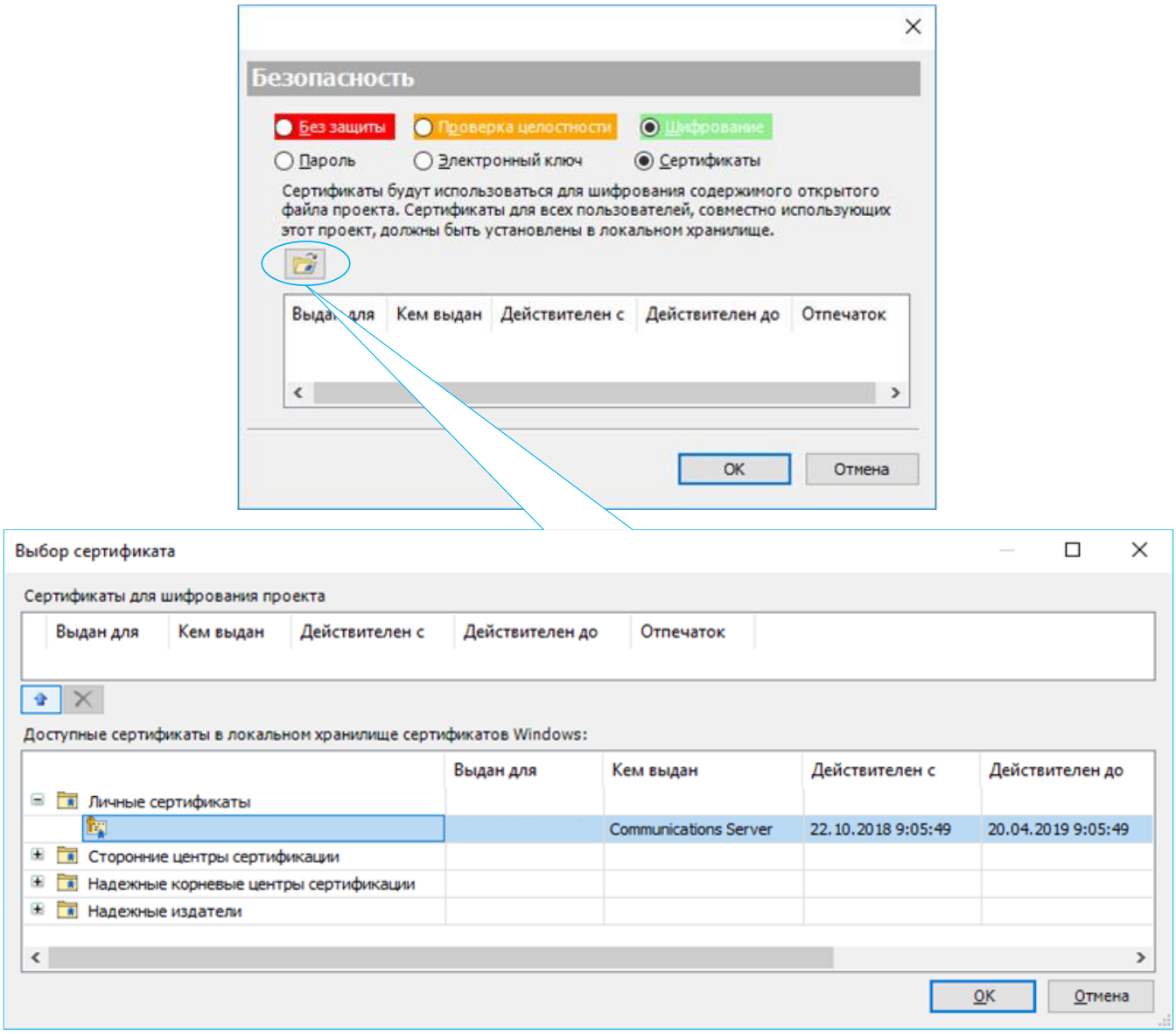


Рисунок 274 – Диалоговое окно выбора сертификата

Выбираем сертификат из списка доступных и с помощью кнопки  добавляем в список сертификатов для шифрования. Для удаления воспользуйтесь кнопкой  . Двойным щелчком левой кнопки мыши по сертификату, приведет к открытию окна с общей информацией о сертификате.

Значки сертификатов:

 – сертификаты;

 – сертификаты с частным ключом;

 – непроверенные сертификаты.

Различие сертификатов:

- Сертификаты с частными ключами:
  - для дешифрования файлов;
  - для цифровых подписей.
- Сертификаты с открытыми ключами (x.509):
  - для шифрования файлов;
  - для проверки цифровых подписей.

Если шифрование выполняется одним ключом, то дешифрование производится другим. Открытые ключи передаются другим пользователям для проверки цифровых подписей и шифрования пересылаемых файлов. Частные ключи не могут быть экспортированы; они используются для создания цифровых подписей и дешифровки файлов. Закрытый ключ известен только владельцу. При шифровании с открытым ключом важна достоверность открытого ключа и стороны, передавшей его, иначе возможна подмена открытого ключа и осуществление несанкционированного доступа к передаваемым зашифрованным файлам. Применение сертификата гарантирует достоверность корреспондента, созданного авторизованным генератором сертификатов. Сертификаты содержат дополнительную информацию, позволяющую идентифицировать владельца личного ключа, соответствующего данному открытому ключу. Сертификат должен быть подписан авторизованным генератором сертификатов.

Настройка прав доступа к ПЛК

Для управления учетными записями перейдите на вкладку устройства (**Device**) **Пользователи и группы (Users and Groups)** нажмите кнопку обновить  (Рисунок 275), после чего в списке **Пользователи (Users)** появится запись *Administrator* (только данный пользователь изначально наделен правами управления пользователями).

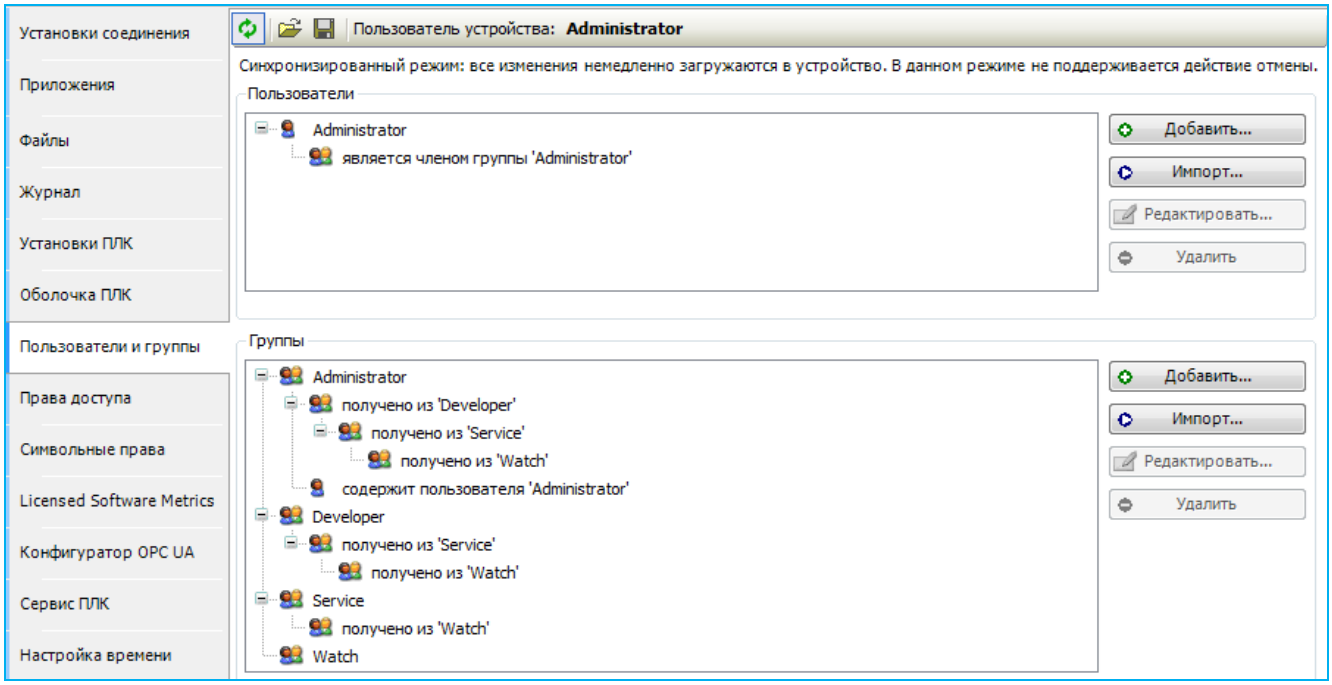


Рисунок 275 – Диалоговое окно пользователи и группы

При повторных подключениях к ПЛК будет всплывать окно авторизации (Рисунок 276) При первичном подключении и успешной авторизации откроется окно с требованием сменить пароль учетной записи (до 1.7.0.0), либо активировать управление пользователями и задать параметры (начиная с 1.7.0.0) (см. подраздел «Установка соединения с контроллером» пункт «Сканирование сети»).

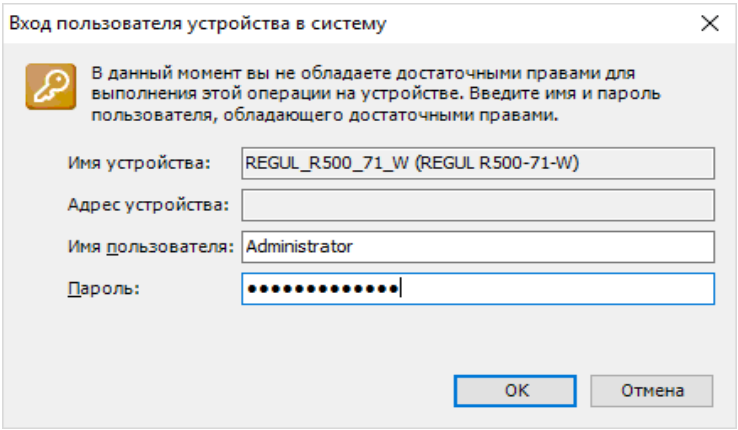


Рисунок 276 – Диалоговое окно авторизации пользователя

Пользователи и группы

Во вкладке **Пользователи и группы (Users and Groups)** можно добавлять, импортировать, редактировать или удалять **Пользователей / Группы** следующим образом:

– напротив списка **Пользователи (Users)**, для создания новой учетной записи (например: <new>), нажмите кнопку Add... (**Добавить…**). В появившемся диалоговом окне **Добавить пользователя (Add User),** заполните поля **Name (Имя),**

Пароль (Password), Подтверждение пароля (Confirm password). В поле **Выберите группу по умолчанию:** с выпадающим списком, выберите к какой группе будет принадлежать новый пользователь. Подтвердите изменения нажатием кнопки **OK** (Рисунок 277);

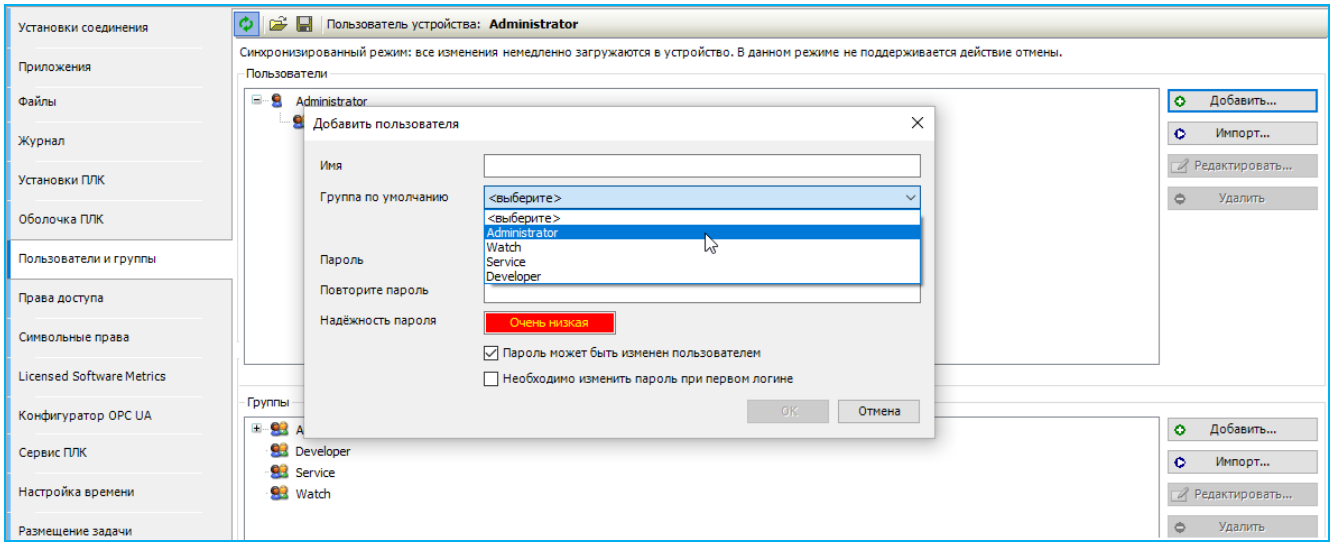


Рисунок 277 – Диалоговое окно добавления нового пользователя

– аналогично напротив списка **Группы (Groups)** можно создать новую группу, нажмите кнопку **Добавить** (Рисунок 278). Далее из списка выберите пользователей, которые должны принадлежать к новой группе;

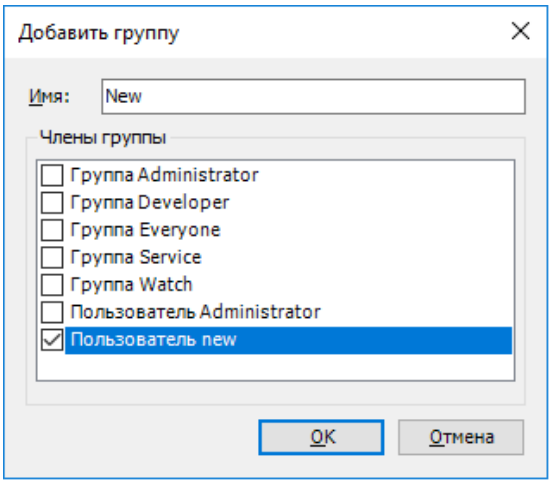


Рисунок 278 – Диалоговое окно добавления новой группы

– для редактирования существующей группы (редактирование), выберите группу из списка и нажмите кнопку 

(Редактировать…), либо нажмите двойным щелчком по ней. Далее в появившемся окне **Редактировать группу** <…> установите / снимите флажок в нужной строке, подтвердите действие нажатием кнопки **OK** (Рисунок 279);

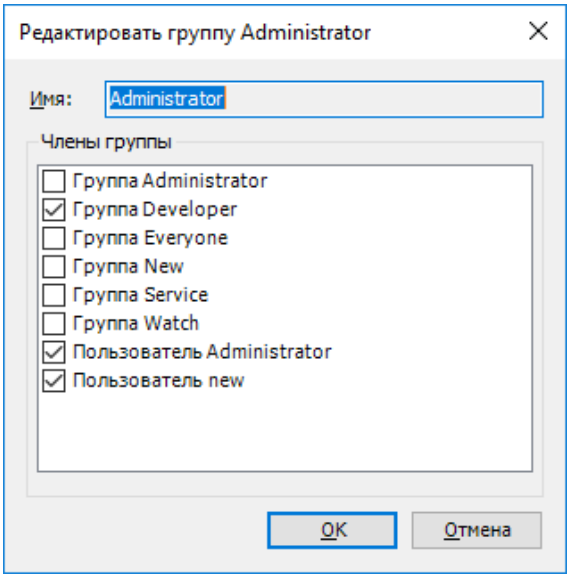


Рисунок 279 – Добавление / удаление пользователя в существующую группу

– для редактирования пользователя, выберите в списке **Пользователи** (**Users**) учетную запись и нажмите на кнопку Редактировать... (**Редактировать…),** либо нажмите двойным щелчком по учетной записи (Рисунок 280).

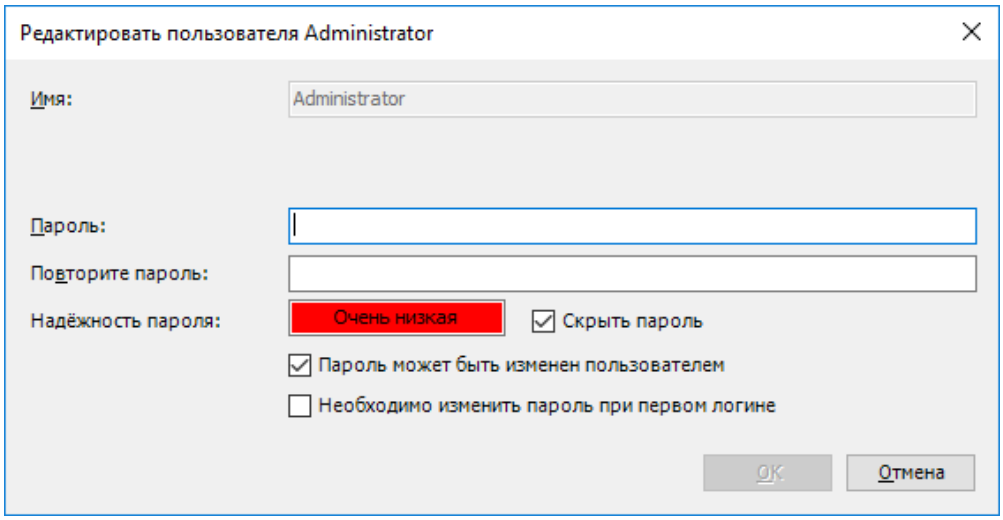


Рисунок 280 – Диалоговое окно редактирования пользователя

– В появившемся диалоговом окне **Редактировать пользователя (Edit User) <…>** заполните поля **Пароль (Password)** и **Повторите пароль (Confirm password)**, подтвердите изменения нажатием кнопки **OK**;

– для импортирования **Пользователей / Группы**, заданных в проекте, нажмите кнопку Импорт... (**Импорт…**, в соответствующем списке), всплывет диалоговое окно **Импорт пользователей / групп** (Рисунок 281). Выберите **Пользователей / Группы** и подтвердите нажатием кнопки **OK.** В целях безопасности все импортируемые учетные записи пользователей переносятся с пустыми паролями.

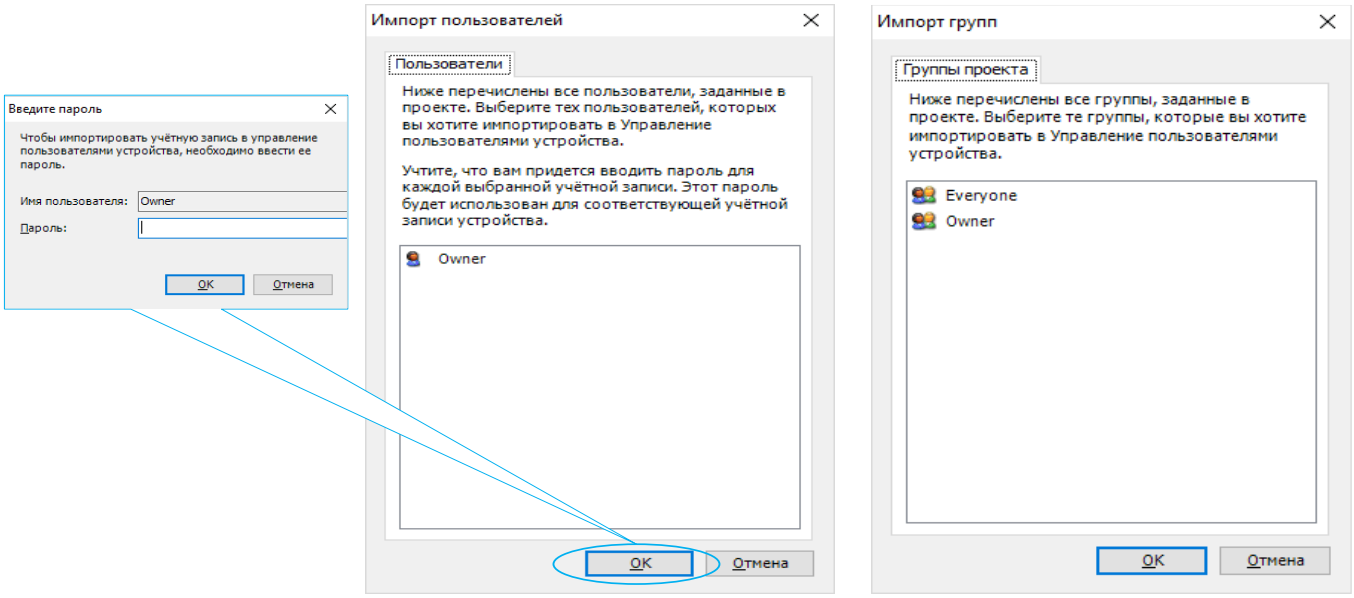


Рисунок 281 – Диалоговые окна импорта пользователей/групп

По завершению на вкладке устройства **Пользователи и группы (Users and Groups)** нажмите кнопку  для загрузки в ПЛК всех изменений.

Права доступа

Предоставьте доступ только уполномоченному пользователю, измените все заданные стандартные пароли при первом вводе в эксплуатацию. Для того, чтобы предоставить или отозвать права на выполнение действий с файлами и объектами в ПЛК группам пользователей (права можно предоставить только группам пользователей), перейдите на вкладку устройства **Права доступа (Access Rights)**. Нажмите кнопку  для загрузки конфигурации с ПЛК (Рисунок 282).

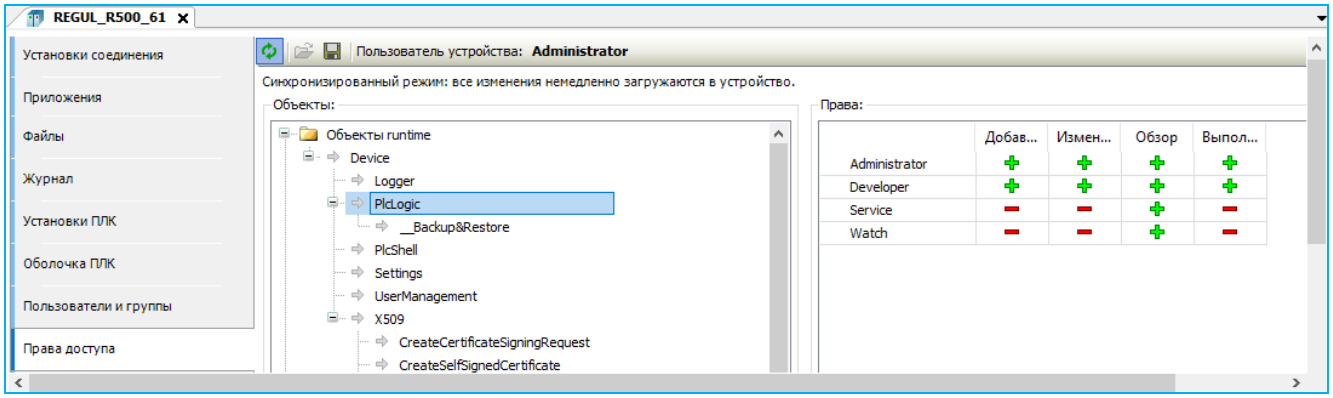


Рисунок 282 – Диалоговое окно прав доступа

В окне **Объекты:** в виде дерева представлены все объекты по категориям, над которыми могут быть выполнены действия. Типы действий, которые могут быть выполнены группой над объектом представлены в окне **Права:**

Представлены четыре типа действий над объектом:

– **добавить / удалить потомка** (возможность пользователю, принадлежащему к группе, добавлять или удалять дочерние объекты);

– **изменить** (возможность изменять объект пользователем);

– **обзор** (возможность мониторинга объекта пользователем);

– **выполнить** (возможность, например, запускать и останавливать приложение).

Для того чтобы переназначить право определенного действия над объектом группе пользователей, выберите ячейку в окне **Права:** и двойным щелчком левой кнопки мыши по ячейке смените разрешение (- предоставлено, - отозвано, - отключено, / - предоставлено / отключено родительским объектом,). Повторяйте процесс пока не установите необходимое разрешение. По завершению нажмите кнопку для загрузки в ПЛК всех изменений.

Текущие конфигурации **Пользователи и группы** / **Права доступа** можно сохранить в xml - файл (***.dum**) / (***.drm**) соответственно, для копирования в другие ПЛК. В шапках вкладок **Пользователи и группы** / **Права доступа** для выгрузки сохраненной конфигурации необходимо нажать кнопку **(Load from Disk),** а для сохранения кнопку **(Save to Disk)**. В появившемся соответствующем диалоговом окне укажите путь выгрузки / сохранения.

Для входа в контроллер под необходимой учётной записью следует отключиться от текущего онлайн пользователя, перейдя на вкладку **Онлайн⇨ Безопасность (Security) ⇨Отключить текущего пользователя устройства,** а затем повторно подключиться к ПЛК (Рисунок 283).

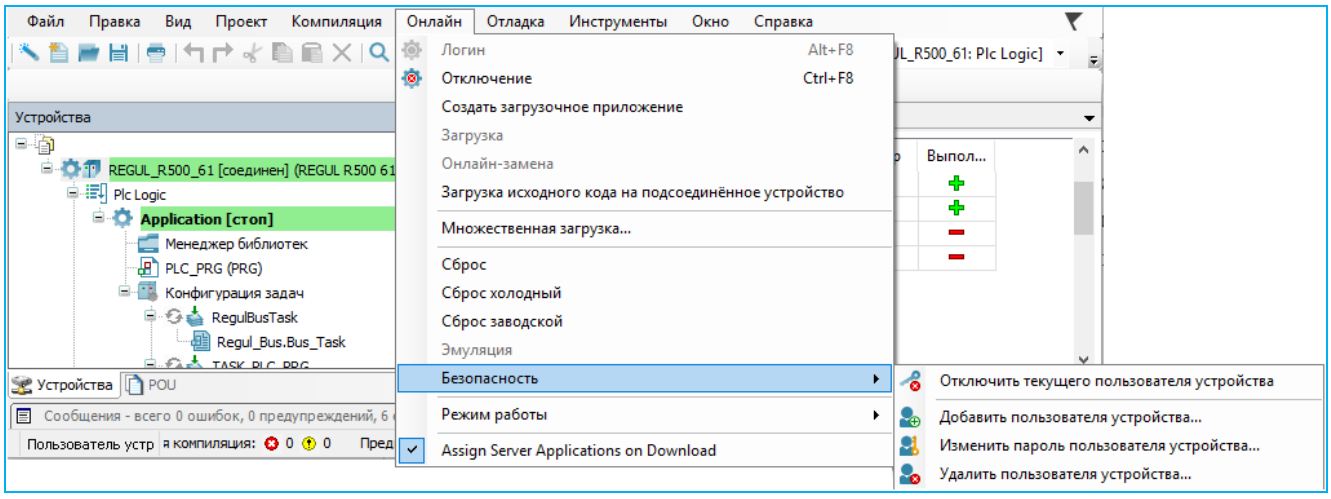


Рисунок 283 – Диалоговое окно управления онлайн - пользователями

Сбросить ограничения доступа к ПЛК до заводского можно двумя способами: через ПО Astra.IDE (см. подраздел «Установка соединения с контроллером» пункт «Сканирование сети») или сервисный режим контроллера (см. подраздел «Сервисный режим контроллера»

пункт «Алгоритм сброса к заводским настройкам»). Сервисный режим используют, когда нет возможности подключения к контроллеру через Astra.IDE.

Настройка прав доступа к объектам проекта среди пользователей

Настройка прав доступа к объектам проекта назначается только группам пользователей, а не отдельным учетным записям. Каждый пользователь должен быть членом какой-либо группы. Прежде чем задать пользователей и группы пользователей, обратите внимание на следующее:

- автоматически создается группа **«Everyone»** (**«Все»**), изначально все заданные пользователи и группы пользователей принадлежат к ней, она всегда есть и оснащена стандартными правами (по умолчанию не имеет прав изменять текущих пользователей, группу и конфигурацию прав). Группу нельзя удалить (а также ее пользователей), но можно переименовать;
- также автоматически создается группа **«Owner»** (**«Владелец»**), содержащая одного пользователя **Owner** (**Владелец**) (первоначально имеет права изменять пользователей, группу и конфигурацию прав в новом проекте, ей предоставлены все права доступа). Группу нельзя удалить (участники могут быть удалены или добавлены, но хотя бы один должен остаться), можно переименовать группу и пользователя.

При запуске / перезапуске системы или проекта никакие изначально пользователи не подключены к проекту (Текущий пользователь: <никто> (Current user:<nobody>)).

Для получения заданного набора прав доступа пользователь может войти в систему с помощью существующей учетной записи и пароля (выбрав в основном меню **Проект** (**Project**) ⇨ **Управление пользователями (User Management)** ⇨ **Подключение пользователя…(User Logon…)** (Рисунок 284).

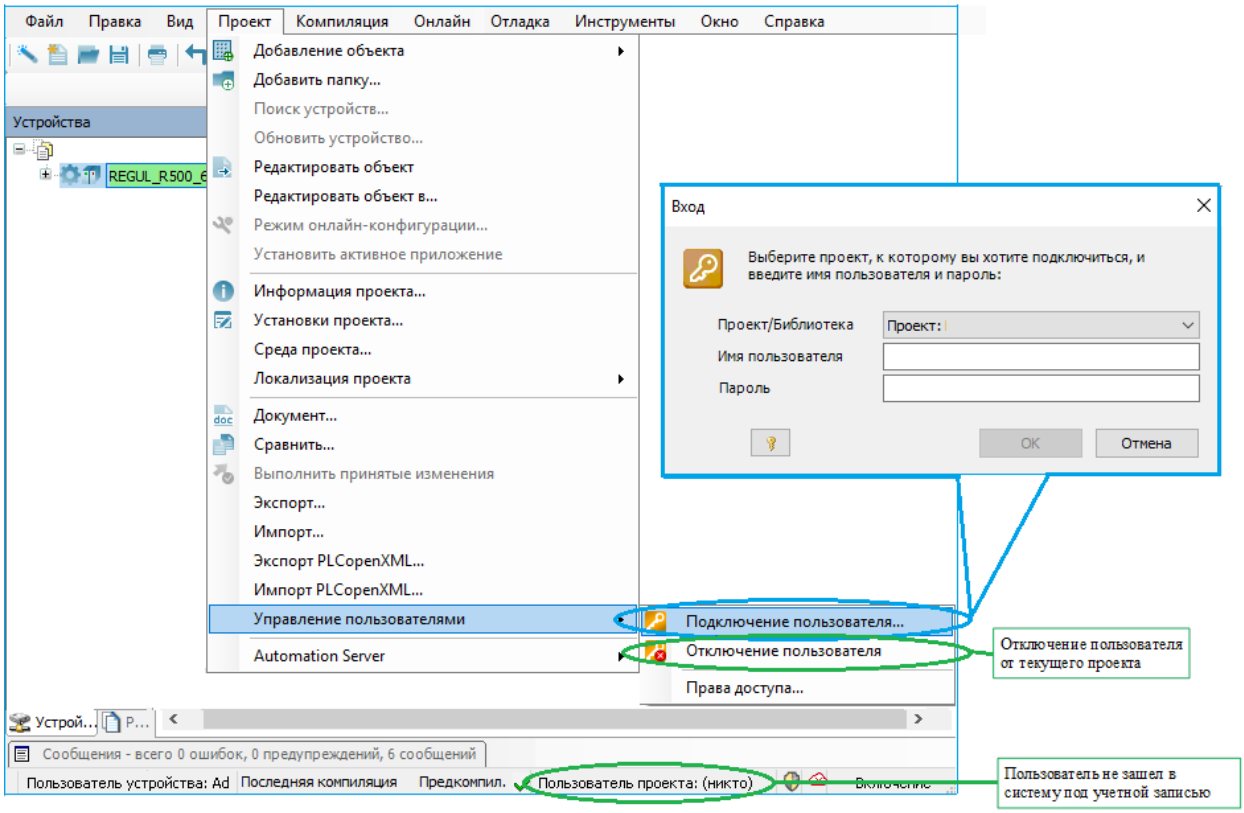


Рисунок 284 – Меню для входа/выхода определенного пользователя

Первоначально можно зайти в систему под учетной записью **Owner** и «пустым» паролем (Текущий пользователь: <Owner> (Current user:<Owner>)). Только **Owner (Владелец)** сначала имеет право изменить текущую конфигурацию пользователя, группы и прав в новом проекте. Следовательно, только **Owner (Владелец)** может назначить это право другой группе.

Пользователи и группы проекта

Последовательность действий:

– выберите в основном меню команду **Проект(Project)** ⇨ **Установки проекта (Project Settings)…** и в появившемся окне категорию **Пользователи и группы (Users and Groups)** (Рисунок 285). Категория разбита на три подкатегории: **Пользователь, Группы** и **Установки;**

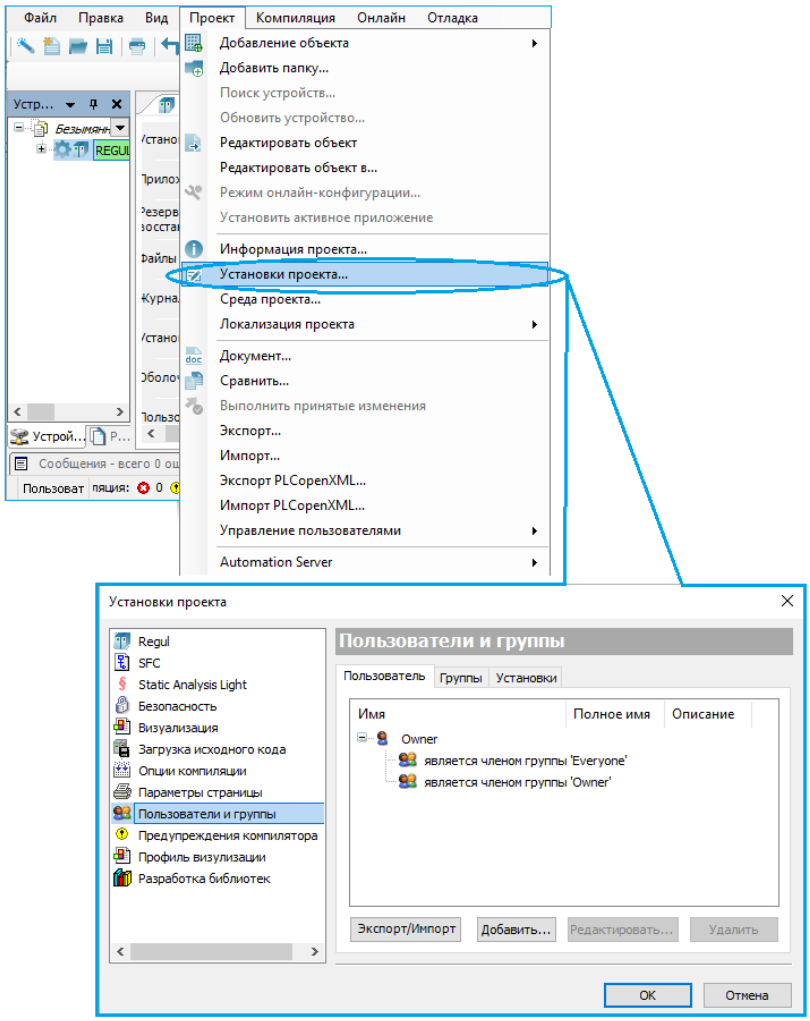


Рисунок 285 – Диалоговое окно установки проекта категории пользователи и группы

– для создания новой учетной записи, в подкатегории Пользователь (Users), нажмите на кнопку **Добавить** откроется диалоговое окно (Рисунок 286).

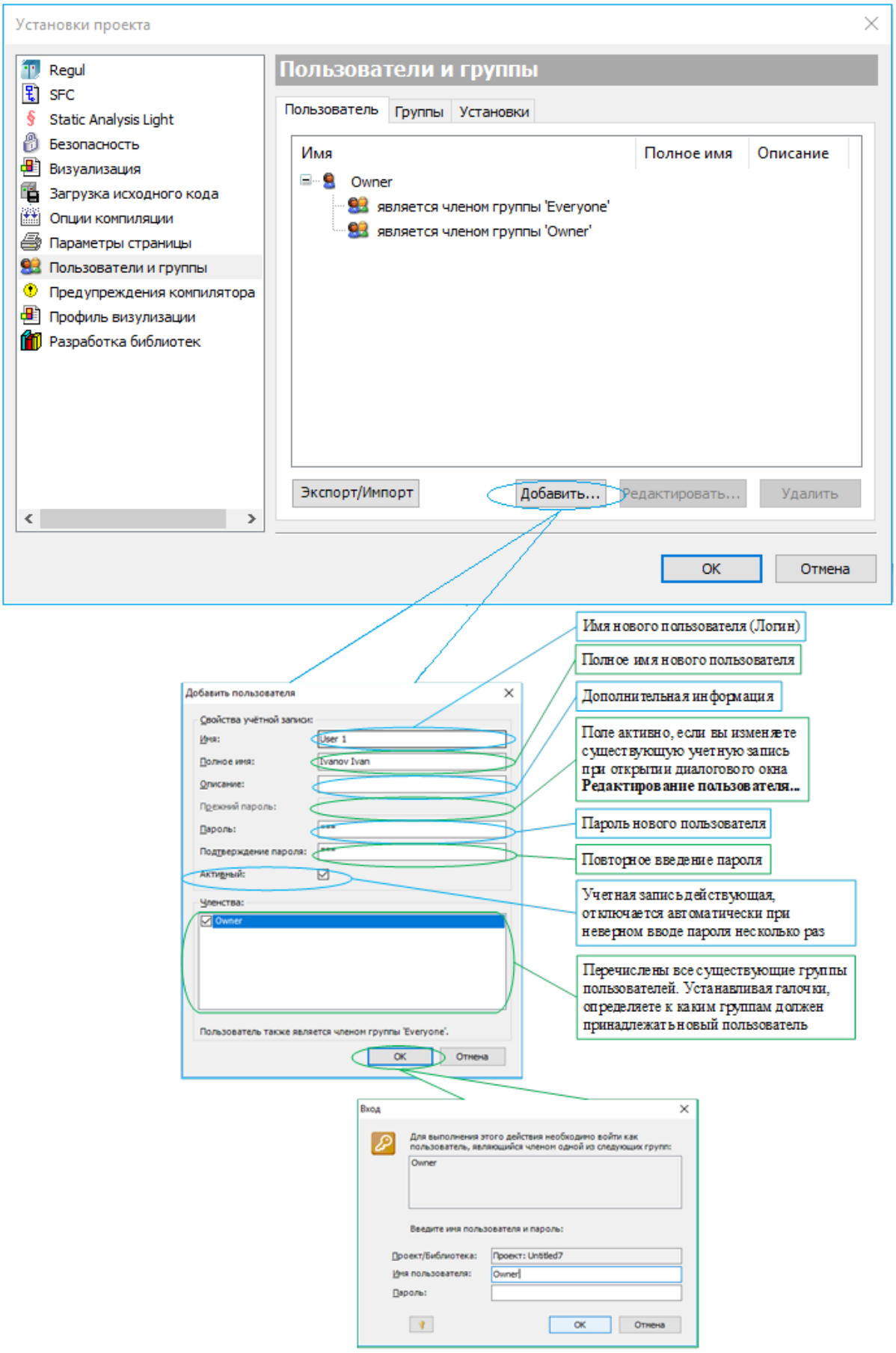


Рисунок 286 – Диалоговое окно добавления пользователя

– Если пользователь не зашел под существующей учетной записью, то при подтверждении на выполнение определенных действии будет всплывать окно авторизации. Введите имя пользователя *Owner*, а строку **Пароль:** не заполняйте (оставьте пустой) и нажмите **OK**;

– чтобы изменить учетную запись, в подкатегории **Пользователь (Users)** выберете нужную и нажмите кнопку **Редактировать… (Edit…)**, всплывет диалоговое окно **Редактировать пользователя…(Edit User)** схожее с диалоговым окном **Добавить пользователя…(Add User)**, только будет активно еще поле **Прежний пароль (Old password)**;

– чтобы удалить учетную запись, выберете нужную и нажмите кнопку **Удалить (Remove).** По крайней мере один пользователь в группе должен остаться;

	**ВНИМАНИЕ!** Если вы забыли пароль пользователя, то сменить пароль и использовать учетную запись уже не получится, т.к. для смены пароля требуется ввод старого пароля. Если забыли пароль пользователя Owner (Владелец), то весь проект станет недоступным

– для создания новой группы, в подкатегории **Группы (Groups)**, нажмите на кнопку **Добавить…** откроется диалоговое окно (Рисунок287). Группу можно изменить и/или удалить, но учтите, что группу **«Everyone» («Все»)** и **«Owner» («Владелец»)** удалить не получится;

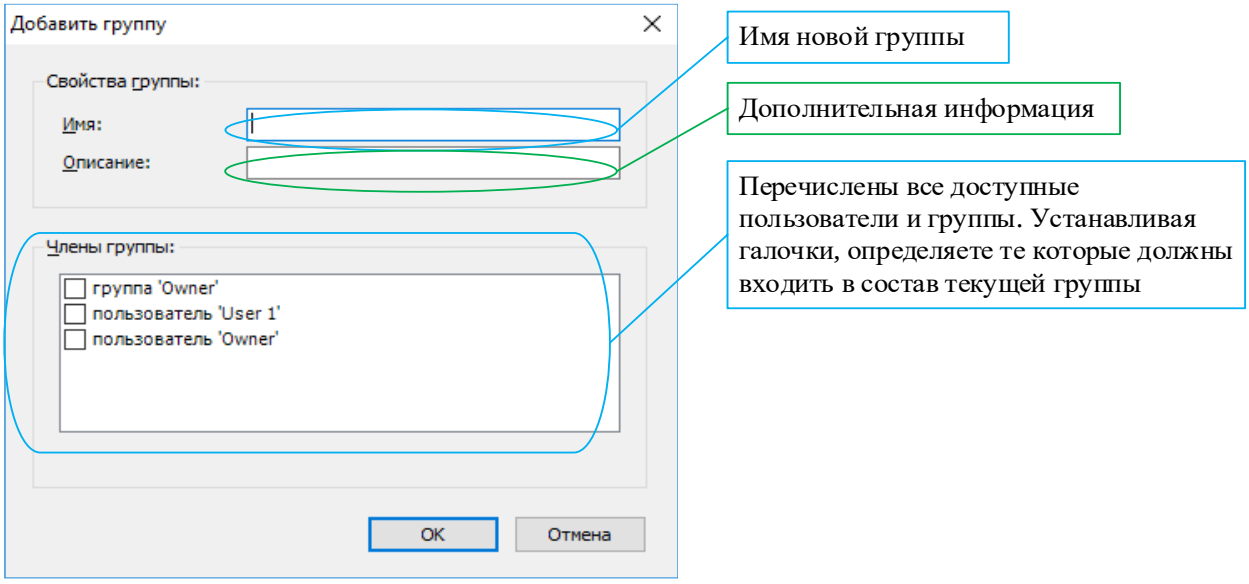


Рисунок 287 – Диалоговое окно добавления группы

– для использования конфигурации управления пользователями из другого проекта нажмите кнопку **Экспорт / Импорт**. Выберите экспорт (тип файла: Users and groups(*.users)) или импорт групп и пользователей;

– в подкатегории **Установки (Settings)** можно задать опции (количество неудачных попыток, период бездействия и «хеширование» пароля), касающиеся учетных записей пользователей (Рисунок 288);

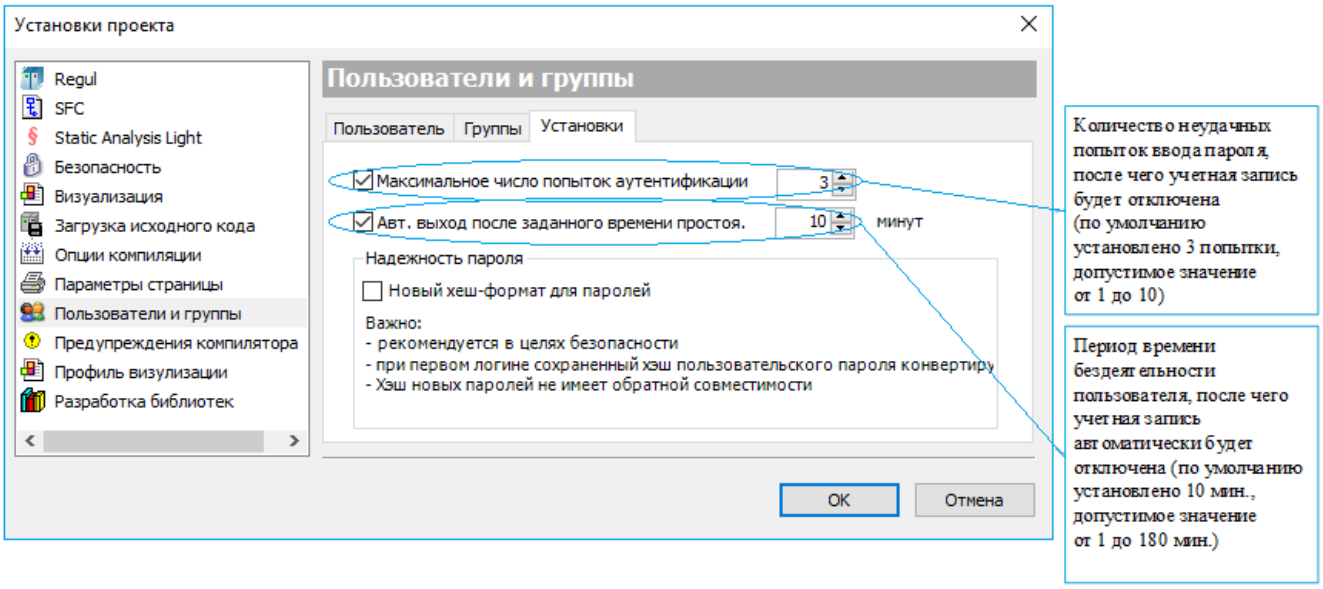


Рисунок 288 – Диалоговое окно для задания опций учетных записей

Чтобы применить новый хеш-формата для паролей, установите флажок в поле **Надежность пароля**. Хеш-формат – это односторонняя функция, при которой вычисляется хеш-код пароля и сохраняется в базе данных. При повторном входе в систему, введенный хеш-код, вычисленный от пароля, сравнивается с хеш-кодом реального пароля.

Каждый проект имеет свое собственное управление пользователями! Поэтому, чтобы получить конкретный набор прав доступа к библиотеке, включенной в проект, пользователь должен отдельно войти в эту библиотеку. Пользователи и группы пользователей, заданные в разных проектах, не идентичны, даже если они имеют одинаковые имена.

Управление пользователями в проекте имеет смысл только в сочетании с назначением прав доступа к объектам в проекте. В новом проекте права не задаются явно, по умолчанию они разрешены. В дальнейшем при работе с проектом каждое право можно явно назначить, отозвать или вернуть в значение по умолчанию.

Права доступа проекта

Чтобы назначить права проекта выберите в основном меню команду **Проект(Project)** ⇨ **Управление пользователями (User Management)** ⇨ **Права доступа…(Permission…)** (Рисунок 289).

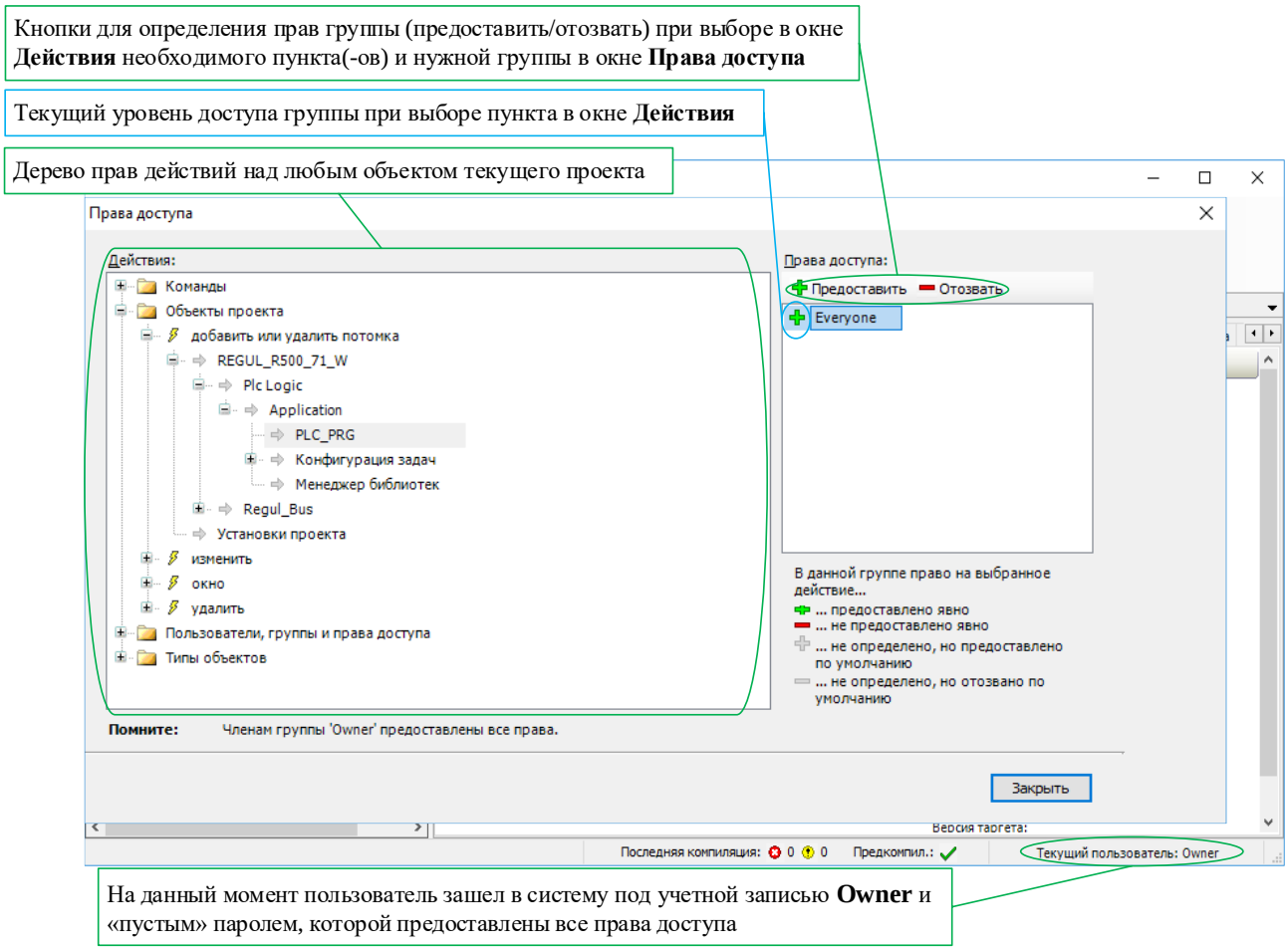


Рисунок 289 – Диалоговое окно прав доступа

Чтобы изменить права доступа к отдельному объекту проекта:

– выберите в окне дерева устройств необходимый объект и нажмите правую кнопку мыши;

– в появившемся контекстном меню выберите пункт **Свойства… (Properties…)**;

– в всплывшем диалоговом окне перейдите на вкладку **Контроль доступа (Access control)** (Рисунок 290). Здесь можно задать права доступа к текущему объекту для разных групп пользователей. Для того чтобы изменить право на конкретное действие конкретной группы, выберите соответствующее поле, щелкните мышкой, чтобы открылся список для выбора нужного права.

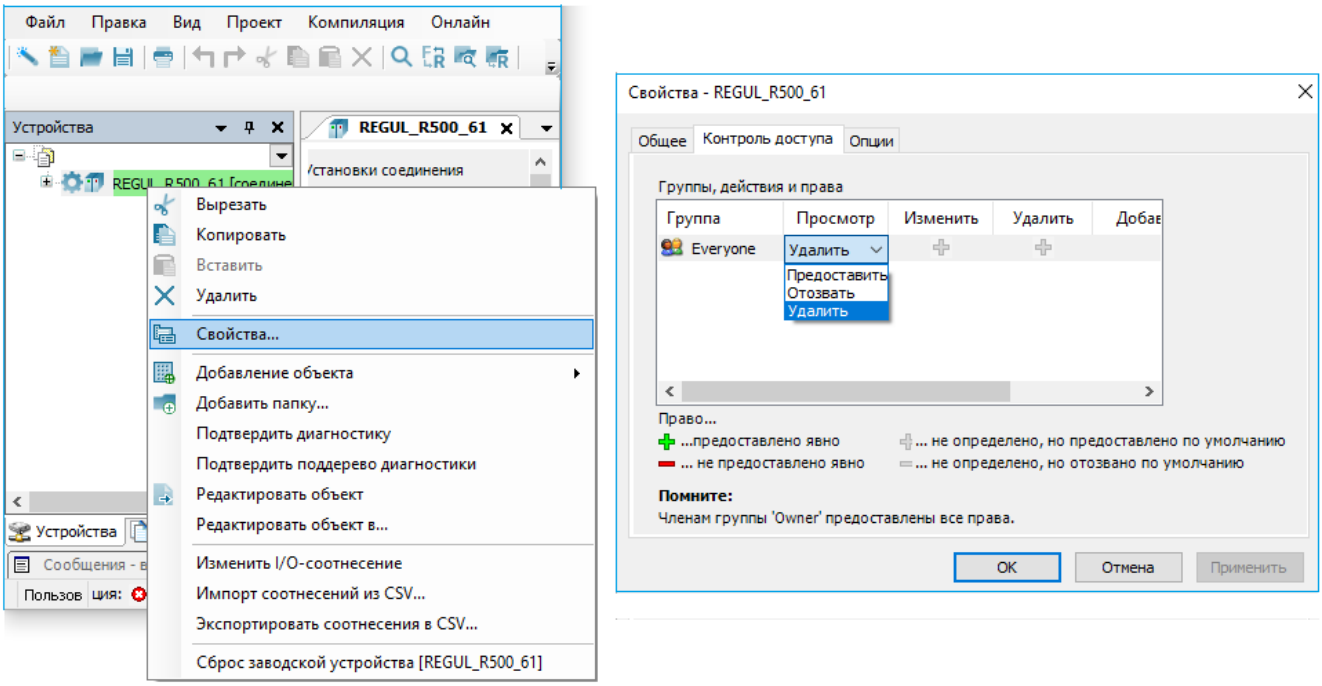


Рисунок 290 – Диалоговое окно контроля доступа

Политика пользователей БД MySQL

В состав программного обеспечения входит система управления базами данных (далее по тексту – БД) MySQL. БД (а также функции) доступна в модулях ЦП I-го/ III-го типа.

По умолчанию возможность удаленного доступа к БД отключена. Для включения доступа перейдите на вкладку **Сервис ПЛК ⇨ Системные параметры** (см. подраздел «Настройка системных параметров»). Нажмите кнопку (***Обновить***) (Рисунок 291).

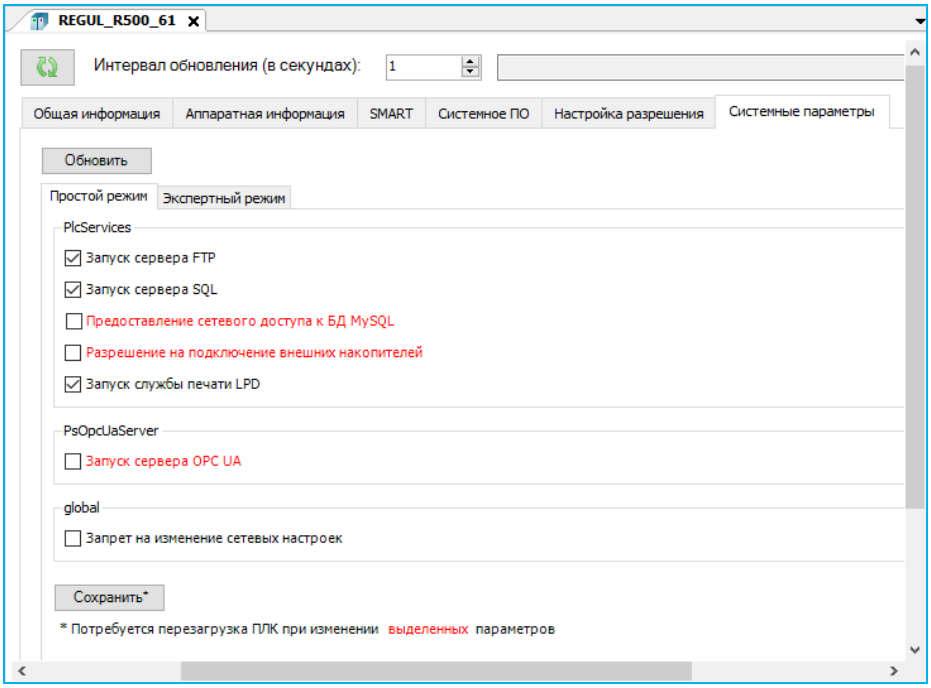


Рисунок 291 – Вкладка системные параметры

Выберите вкладку: **Простой режим** или **Экспертный режим**. Активируйте одним из удобных для вас способов:

- на вкладке **Простой режим** установите флажок ☑ в поле **Предоставление сетевого доступа к БД MySQL;**
- либо, на вкладке **Экспертный режим** в секции [PlcServices] конфигурационного файла *etc/runtime.cfg* добавьте параметр *MySqlNetworkingEnable* равный 1;

```
[PlcServices] MySqlNetworkingEnable=1
```

Нажмите на кнопку **Сохранить**. В обоих случаях, для вступления в силу изменений, перезагрузите контроллер путем выключения/включения питания, либо командой *reboot* на вкладке **Оболочка ПЛК**.

	**ИНФОРМАЦИЯ**
	
	Начиная с версии СПО 1.7.0.0 автоматически запускается обновленная служба сервера SQL (поддержка MariaDB 10.2.44).
	
	Можно деактивировать сервер SQL одним из удобных для Вас способом:
	– на вкладке **Простой режим** снимите флажок ☑ в поле **Запуск сервера SQL;**
	– либо, на вкладке **Экспертный режим** в секции [PlcServices] конфигурационного файла etc/runtime.cfg изменить значение параметра EnableSQL с 1 на 0:
	`[PlcServices]`
	`EnableSQL=0`
	
	Нажмите на кнопку **Сохранить**

Получить доступ к БД MySQL контроллера из приложения можно используя функциональный блок **TMysqlClient** библиотеки **PsMySQLClient**. Задайте настройки подключения для доступа к удалённой базе данных, установив указатель *ConnectionSettings* функционального блока на соответствующую структуру типа *TMysqlSettings*. gg

Доступ к БД контроллера осуществляется через встроенную учетную запись ***user*** со следующими привилегиями:

– локальное и удаленное подключение (локальное подключение производится с использованием пустого пароля). Для удалённого подключения по умолчанию используется пароль: *6ViQgjps*, в последствии пароль можно изменить с помощью **SQL**-запроса:

SET PASSWORD: `SET PASSWORD FOR 'user'@'%' = PASSWORD ('новый пароль');`

– добавлять/удалять/модифицировать таблицы и их содержимое используя SQL-запросы;

– создание/удаление БД производится следующими функциями библиотеки PsMySQL:

- **PsMySQLCreateData** (**'db_name'**, **'CHARACTER SET'**, **'COLLATE'**) - создание БД;
- **PsMySQLDataBase** (**'db_name'**) – удаление БД.

Существует возможность ограничения размера БД. Размер ограничения задается в конфигурационном файле ***plc.cfg*** в секции [Database] параметром MaxDbSize (в Мб). При старте в лог записывается текущий размер файла БД. Если MaxDbSize будет меньше текущего размера, то ограничение будет равным текущему размеру и будет соответствующая запись в логе. Журналирование ошибок происходит в файл: …/logs/db/{$hostname}.log.

Также можно включить журналирование запросов в таблицы mysql.general_log и mysql.slow_log. Размер лога запросов неограничен, поэтому необходимо следить за свободным местом на диске. По умолчанию журналирование отключено (см. «Приложение Е»).

ОБРАЩЕНИЕ В СЛУЖБУ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ

Для обращения в техническую поддержку Пользователю необходимо сформировать запрос на сайте технической поддержки: https://support.prosoftsystems.ru, либо отправить письмо по электронной почте: support@prosoftsystems.ru. В первом случае требуется предварительная регистрация.

Обращение обязательно должно содержать следующие сведения:

- подробное описание сложившейся ситуации;
- наименование объекта и его месторасположение;
- наименование системы автоматизации;
- модель ПЛК;
- серийный номер ПЛК;
- версия пакета обновления для среды разработки Astra.IDE;
- версия СПО-контроллера;
- файл экспорта сетевых настроек контроллера;
- архив с лог-файлами, включающими в себя период времени, когда произошел отказ;
- дата и время возникновения отказа. А также периодичность и устойчивость повторения подобных отказов в случае, если такая информация имеется.

Желательно прислать проект (архив со всеми библиотеками) для Astra.IDE, так как это может значительно упростить и ускорить процесс поиска причины отказа. Пройдите на вкладку **Файл** ⇨**Архив проекта** ⇨ **Сохранить архив** ⇨ в поле **Используемые библиотеки** установите флажок и нажмите кнопку ***Сохранить…***)

Лог-файлы, скопированные на компьютер, желательно поместить в архив. Объем заархивированных текстовых файлов сокращается примерно в 10 раз.

Для того, чтобы узнать версию Astra.IDE, в главном меню выберите **Справка** ⇨ **О программе…** и в открывшемся окне нажмите кнопку ***Информация о версии***.

Для выяснения номера версии СПО и сохранения сетевых настроек контроллера, ознакомьтесь с разделом «Сканер сети. Настройка IP-адресов».

ПРИЛОЖЕНИЕ А

Настройка конфигурационного файла runtime.cfg

Конфигурационный файл находится в каталоге **/etc/runtime.cfg** и содержит некоторые настройки системы, доступные пользователю для редактирования.

Таблица А.1 – Параметры конфигурационного файла

Параметр	Описание	Значение по умолчанию
Основные (прописаны по умолчанию)		
Секция [CmpApp]:		
Bootproject.HardwareSelfDiagFail.Deny	**Запрет** на загрузку приложения в случае, если ПЛК при включении получил ошибку самодигностики. Информация об ошибке журналируется в StdLogger.log в виде сообщений: SysTargetOEM: "Self-diagnostic has failed - system may be in an inconsistent state" SysTargetOEM: "Deny to download boot application: Self-diagnostic has failed" CmpApp: "Bootproject of [<app>Application</app>] denied to load <source>event</source>" Для **разрешения** загрузки приложения, при отрицательном результате самодиагностики, необходимо изменить значение параметра с 1 на 0: Bootproject.HardwareSelfDiagFail.Deny=0	**1**
Секция [PlcServices]:		
EnableSQL	Запуск сервера SQL (1 – запуск, 0 – останов)	**1**
MySqlNetworkingEnable	Предоставление сетевого доступа к БД MySQL	**0**
EnableFTP	Запуск сервера FTP	**0**
EnableLPD	Запуск службы печати LPD	**0**
EnableSNMP	Запуск сервера SNMP	**0**
AutomountStorage	Разрешение на подключение внешних накопителей (USB, MMC/SD)	**0**

<table>
<tr><th colspan="2">Параметр</th><th>Описание</th><th>Значение по умолчанию</th></tr>
<tr><td colspan="4">Секция [PsOpcUaServer]:
В зависимости от версии СПО по умолчанию будет запускаться следующая версия OPC UA сервера:
– начиная с версии СПО 1.7.0.0 – OpcUaServer_OS;
– до версии СПО 1.7.0.0 – OpcUaServer
Разрешается активировать только одну из версий OPC UA! Значения: 0 – отключение, 1 – включение</td></tr>
<tr><td rowspan="2">до версии СПО 1.7.0.0</td><td>Enable</td><td>Запуск сервера OpcUaServer</td><td>0</td></tr>
<tr><td>EnableV2</td><td>Запуск сервера OpcUaServer_OS</td><td>0</td></tr>
<tr><td rowspan="2">Начиная с версии СПО 1.7.0.0</td><td>EnableLegacy</td><td>Запуск сервера OpcUaServer</td><td>0</td></tr>
<tr><td>Enable</td><td>Запуск сервера OpcUaServer_OS</td><td>0</td></tr>
<tr><td colspan="4">Дополнительные (добавляются при необходимости)</td></tr>
<tr><td colspan="4">Секция [PsSysExcept]:
Настройка событий формирования отчетов (logs/dumps.core) при возникновении исключения в прикладном ПО</td></tr>
<tr><td colspan="2">CreateReportCondition</td><td>UnhandledException - необработанные исключения. Отчеты формируются только при возникновении исключения, не обработанного прикладной программой.
NoReport – формирование отчетов отключено
AnyException - любые исключения, как обработанные прикладной программой, так и не обработанные, приводят к формированию отчета.
PostmortemDump – любые исключения, как обработанные прикладной программой, так и не обработанные, приводят к аварийному завершению системы исполнения с формированием дампа памяти процесса</td><td>UnhandledException</td></tr>
<tr><td colspan="4">Секция [PsLed]:</td></tr>
<tr><td colspan="2">RunStopButtonSuperior</td><td>Приоритет переключателя RUN/STOP по сравнению с другими органами управления приложением</td><td>0</td></tr>
<tr><td colspan="4">Секция [PsSysTask]:</td></tr>
<tr><td colspan="2">TaskStatisticLogInterval</td><td>Журналирование статистики задач в лог-файл SysTaskStatistic.log, в секундах. 0 – отключение логирования</td><td>300</td></tr>
<tr><td colspan="2">TaskStatisticLogResetInterval</td><td>Интервал сброса статистики журналирования задач, в секундах. 0 – отключение сброса статистики</td><td>86400</td></tr>
</table>

<table>
<tr><th>Параметр</th><th colspan="2">Описание</th><th>Значение по умолчанию</th></tr>
<tr><td colspan="4">Секция [PsRetain]:</td></tr>
<tr><td>StorageType</td><td colspan="2">Тип хранилища RETAIN переменных (SRAM/File)</td><td>SRAM
(1 Мб)</td></tr>
<tr><td>StorageSize</td><td colspan="2">Размер хранилища RETAIN переменных для типа «File», Кб</td><td>1</td></tr>
<tr><td colspan="4">Секция [CmpChannelMgr]:</td></tr>
<tr><td>ChannelTimeoutMs</td><td colspan="2">Настройка таймаута сессии Astra.IDE, мс</td><td>30000</td></tr>
<tr><td colspan="4">Секция [CmpLog]:
Для вывода информации в оперативный журнал из соответствующего лог-файла. Например, журналирование работы шины RegulBus OS из regul-bus-driver.log (задано по умолчанию)</td></tr>
<tr><td>ExtLogger.N.Name</td><td colspan="2">Имя журнала.
Пример: ExtLogger.0.Name=regul-bus-driver</td><td>ExtLogger.0.Name= regul-bus-driver
#ExtLogger.0.Filter=0 xFFFFFFFF
ExtLogger.1.Name= system</td></tr>
<tr><td rowspan="10">ExtLogger.N.Filter</td><td colspan="2">Уровень протоколирования. Задайте фильтр сообщений для отображения в журнале.
Пример: ExtLogger.0.Filter=0xFFFFFFFF</td><td rowspan="10">0xFFFFFFFF
(отображение всех типов сообщений)</td></tr>
<tr><td>Значение</td><td>Тип записи (описание)</td></tr>
<tr><td>0x1</td><td>LOG_INFO (информаионные сообщения)</td></tr>
<tr><td>0x2</td><td>LOG_WARNING (предупреждение)</td></tr>
<tr><td>0x4</td><td>LOG_ERROR (ошибка)</td></tr>
<tr><td>0x8</td><td>EXCEPTION (исключение)</td></tr>
<tr><td>0x10</td><td>LOG_DEBUG (отладочнные)</td></tr>
<tr><td>0xFFFFFFFF</td><td>LOG_ALL (все)</td></tr>
</table>

ПРИЛОЖЕНИЕ Б

Типы модулей центрального процессора

Таблица Б.1

Тип	Наименование модуля ЦП				
	Модель R200	**Модель R400**	**Модель R500**	**Модель R600**	**Модель R050**
I	—	CU 00 071(-W)	CU 00 051(-W) CU 00 061(-W) CU 00 071(-W) CU 00 052(-W) CU 00 062(-W) CU 00 072(-W) CU 00 082(-W)	CU 00 061(-W) CU 00 071(-W)	
II	CU 00 021(-W) CU 00 031(-W) CU 00 041(-W) CU 00 061(-W)	—	CU 00 021(-W) CU 00 031(-W)	—	CU 00 031(-W) CU 00 051(-W) CU 00 061(-W)
III	—	—	CU 00 151(-W) CU 00 161(-W) CU 00 171(-W) CU 00 181(-W)	—	

ПРИЛОЖЕНИЕ В

Настройка конфигурационного файла network.cfg

Конфигурационный файл находится в каталоге **/etc/network.cfg** и содержит сетевые настройки системы, доступные пользователю для редактирования.

Таблица В.1 – Параметры конфигурационного файла

Параметр	Описание	Значение по умолчанию
Основные (прописаны по умолчанию)		
Секция [**global**]:		
hostname	Символьное сетевое имя контроллера. Максимальная длина имени для ввода -50 символов	-
remote-config	Запрет на изменение сетевых настроек	-
Секция [**ip**]:		
portXX	IP-адрес подключенного порта	-
portXX.N	Установка дополнительного IP-адреса и маски сетевого интерфейса: portXX.N=<ip-адрес> <маска>, где N – номер дополнительно IP-адреса, нумерация начинается с «0»	
Секция [**routing**]:		
gateway	Адрес используемого шлюза	-
Дополнительные (добавляются при необходимости)		
Секция [**mac**]:		
portXX	Mac-адрес для подключения порта	-

ПРИЛОЖЕНИЕ Г

Настройка конфигурационного файла copyjobs.cfg

Конфигурационный файл находится в каталоге **/etc/copyjobs.cfg** и содержит сценарий копирования пользовательских данных, доступный пользователю для редактирования.

Таблица Г.1 – Параметры конфигурационного файла

Параметр	Описание	Значение по умолчанию
[Example]	**Имя сценария**. Задайте имя, которое должно быть уникальным и может содержать: большие и маленькие латинские буквы, цифры, тире, точки	**-**
Source	**Источник**. Укажите источник копирования - файл или директорию. Можно использовать шаблонные имена, используя символ ***** (например, если задать параметр **Source** равный ***.dat**, то из источника будут скопированы все файлы с расширением **.dat**)	**/logs**
Destination	**Место назначения**. Укажите, в какую директорию на контроллере копировать. Если местом назначения является внешний накопитель, то нужно указать директорию /sd0, для контроллера серии R200, или /usb0, для остальных контроллеров	**/sd0**
Schedule	**Расписание копирования**. Задайте частоту копирования в формате **cron**. Например: – *** * * * *** – каждую минуту; – **59 23 31 12 5** – за минуту до конца года, если последний день года – пятница; – **45 17 7 6 *** – каждый год 7-го июня в 17:45; – **0 9 1-7 * 1** – первый понедельник каждого месяца, в 9 утра; – **0 0 1 * *** – в полночь, первого числа, каждый месяц; – **0 0 * * *** – каждый день в полночь; – **0 0 * * 3** – каждую среду в полночь	**"* * * * *"**
Verbose	**Уровень журналирования.** Задайте уровень подробности журналирования. Доступно два уровня: – **0 (либо не задан)** – логируются только ошибки; – **1** – логируются ошибки и информационные сообщения	**1**

ПРИЛОЖЕНИЕ Д

Журналирование системных параметров ПЛК

Диагностические сообщения записываются в лог-файлы, которые находятся в каталоге /logs/stats/sysinfo. Для просмотра информации перейдите в папку ***sysinfo*** на контроллере и скачайте на ПК необходимые лог-файлы.

Таблица Д.1 – Лог-файлы системных параметров

Лог-файл	Описание
cpu.log	Информация о загрузке процессора в процентах. Период записи информации задается в конфигурационном файле plc.cfg. **Пример:** 25.05.2021 11:48:50 2% 1% 25.05.2021 11:48:51 1% 0% 25.05.2021 11:48:52 1% 1% 25.05.2021 11:48:53 1% 1% 25.05.2021 11:48:54 1% 0%
hdd.log	Информация об используемой памяти на разделах файловой системы (в формате «Дата_Время» «раздел»: «занято» Мб/ «общий объем» Мб). Период записи информации задается в конфигурационном файле plc.cfg. **Пример:** hdd.log: **07.05.2020 07:54:09.685** System: 285MB/401MB Archive: 153MB/3424MB App: 153MB/3424MB **07.05.2020 07:55:09.215** System: 284MB/401MB Archive: 153MB/3424MB App: 153MB/3424MB
ram.log (общее), **ram-TOP5.log** (Топ пяти процессов)	Информация о используемой оперативной памяти. Период записи информации задается в конфигурационном файле plc.cfg. **Примеры:** ram.log: **29.08.2020 22:59:54.889:** System RAM: 1942.831MB Total Used: 707.119MB Used Private: 611.490MB Used Shared: 95.629MB **29.08.2020 23:00:54.188:** System RAM: 1942.831MB Total Used: 707.322MB Used Private: 611.693MB Used Shared: 95.629MB ram-TOP5.log:

Лог-файл	Описание
	07.05.2020 07:39:34.637 309932032 118784 49152 309592064 172032 0 4104 devb-ahci 146382848 0 172032 146182144 28672 0 11431998 sysinfo 90292224 5550080 200704 84230144 110592 200704 13049904 mysqld 65171456 49152 32768 102400 40960 64946176 6434832 io-display 53555200 43675648 2293760 5718016 196608 1671168 14725194 devb-ram **07.05.2020 07:40:39.225** 309932032 118784 49152 309592064 172032 0 4104 devb-ahci 158539776 0 172032 158339072 28672 0 11431998 sysinfo 90292224 5550080 200704 84230144 110592 200704 13049904 mysqld 65171456 49152 32768 102400 40960 64946176 6434832 io-display 53555200 43675648 2293760 5718016 196608 1671168 14725194 devb-ram
network.log	Информация о наличии соединения и скорости соединения по всем сетевым интерфейсам (в формате «Дата_Время»: «порт»; «состояние»; «скорость передачи»; «кол-во переданных пакетов»; «принятых пакетов»; «переданных байт»; «принятых байт»; «пакетов с ошибочной CRC»; «с ошибкой выравнивания»). Период записи информации задается в конфигурационном файле **plc.cfg.** **Пример:** **02.04.2021 08:01:00.116:** ecatbus0 link=1 media_rate=100000 rx_bytes=0 rx_pkts=0 tx_bytes=0 rx_pkts=0 fcs_errors=0 align_errors=0 **02.04.2021 08:01:00.324:** port30 link=0 media_rate=0 rx_bytes=0 rx_pkts=0 tx_bytes=0 tx_pkts=0 fcs_errors=0 align_errors=0 port40 link=0 media_rate=0 rx_bytes=0 rx_pkts=0 tx_bytes=0 tx_pkts=0 fcs_errors=0 align_errors=0 port50 link=1 media_rate=1000000 rx_bytes=83611 rx_pkts=1148 tx_bytes=0 tx_pkts=1148 fcs_errors=0 align_errors=0 port60 link=0 media_rate=0 rx_bytes=0 rx_pkts=0 tx_bytes=0 tx_pkts=0 fcs_errors=0 align_errors=0 tap0 link=1 media_rate=100000 rx_bytes=0 rx_pkts=0 tx_bytes=0 tx_pkts=0 fcs_errors=0 align_errors=0
network_events.log	Информация об изменении состояния порта. Запись информации происходит по факту, при изменении состояния и скорости, на любом из портов. **Пример:** **17.05.2021 05:12:32.672**: ecatbus0 link: 0 → 1 ecatbus0 media_rate: 0 → 100000 **17.05.2021 05:12:32.695:** port30 link: 0 → 1 port30 media_rate: 0 → 1000000 port50 link: 0 → 1 port50 media_rate: 0 → 100000 port60 link: 0 → 1

Лог-файл	Описание
	port60 media_rate: 0 → 1000000 tap0 link: 0 → 1 tap0 media_rate: 0 → 100000 **20.05.2021 05:47:21.773**: port60 link: 1 → 0 port60 media_rate: 1000000 → 0
smi.log	Информация о значении счетчика SMI прерываний в системе. Запись информации происходит по факту изменения значения счетчика (в формате «Дата Время»: «Значение счетчика SMI»)

ПРИЛОЖЕНИЕ Е

Настройка конфигурационного файла plc.cfg

Конфигурационный файл находится в каталоге **/etc/plc.cfg** и содержит некоторые настройки системы, доступные пользователю для редактирования.

Таблица Е.1 – Параметры конфигурационного файла

Параметр	Описание	Значение по умолчанию
Основные (прописаны по умолчанию)		
Секция [Display]:		
ScreenSaver	Изменение режима работы подсветки дисплея R400	**0**
Секция [Database]:		
MaxDbSize	Ограничение размера БД (в МБ). Ограничение только для таблиц формата InnoDB и не касается логов запросов. Если MaxDbSize будет меньше текущего размера, то ограничение будет равным текущему размеру и будет соответствующая запись в лог ошибок	**1024**
GeneralLog	Журналирование всех запросов 0 - выключено, 1 – включено	**0**
SlowQueryLog	Журналирование медленных запросов 0 - выключено, 1 – включено	**0**
LongQueryTime	Время (в секундах), дольше которого запрос считается медленным (0 - 10 секунд)	**1**
Секция [NTP]:		
AllowPanic	Остановка службы NTP при старте	**1**
Секция [FTP]:		
TLS	Запуск TLS-соединения	**1**

PassivePortRange	Изменение диапазона портов при работе FTP сервера в пассивном режиме, в формате:
	`PassivePortRange = <Начальный порт> <Конечный порт>,`
	где <**Начальный порт**> - номер порта начала диапазона,
	<**Конечный порт**> - номер порта конца диапазона.
	При условии, что значение диапазона находится в разрешенных границах:
	`1024 ≤ <Начальный порт> ≤ <Конечный порт> ≤ 65534`
		49152 65534

Параметр	Описание	Значение по умолчанию
Секция [GNSS]:		
GNSSparam	Блокирование приема сигнала точного времени от спутников. В зависимости от установленного значения, прием будет осуществляться: – 1 – только от спутников системы GPS; – 2 – только от спутников системы GLONASS; – 3 – от спутников системы GPS и GLONASS	**3**
Секция [Logging]:		
CpuLoad	Период записи информации о загрузке процессора в процентах в лог-файл **cpu.log**, в секундах	**1**
Hdd	Период записи информации о используемой памяти на разделах файловой системы в лог-файл **hdd.log**, в секундах	**60**
Ram	Период записи информации о используемой оперативной памяти в лог-файлы: **ram.log** (общее), **ram-TOP5.log** (топ пяти процессов); в секундах	**60**
Network	Период записи информации о наличии соединения и скорости соединения по всем сетевым интерфейсам в лог-файл **network.log**, в секундах	**60**
DefaultMaxFileSize	Размер одного лог-файла, байт	**5242880**
DefaultMaxFile	Количество лог-файлов для ротации	**5**
Определение глубины логирования различных журналов, с учетом свободного объема памяти (см. в разделе **archive,** рисунок 210). Например, regul-bus-driver, StdLogger		
Logger.N.Name	Имя журнала. Пример: Logger.0.Name=StdLogger	
Logger.N.MaxFiles	Максимальное количество создаваемых файлов для журнала, диапазон: от 1 до 10 файлов. Пример: Logger.0.MaxFiles=10	**5**
Logger.N.MaxFileSize	Максимальный размер одного файла, в байтах, диапазон: от 64КБ до 10МБ Пример: Logger.0.MaxFileSize=65536	**5242880**
Секция [BackgroundImage]:		
File	Определяет путь до пользовательского изображения на ПЛК. Пример: File=background/background_test.png	**-**

Параметр	Описание	Значение по умолчанию
Fill	Определяет повтор изображения по горизонтали(x)/вертикали(y), со следующими возможными значениями (при некорректном вводе, применяется значение по умолчанию): – none – параметр выключен; – repeat-x – повторить изображение по горизонтали; – repeat-y – повторить изображение по вертикали; – repeat-xy – повторить изображение по горизонтали и вертикали. Пример: Fill=repeat-xy	**none**
Align	Определяет место расположения изображения на экране, со следующими возможными значениями (при некорректном вводе, применяется значение по умолчанию): – center – выравнивание по центру; – top – выравнивание по верхнему краю; – bottom – выравнивание по нижнему краю; – left – выравнивание по левому краю; – right – выравнивание по правому краю. Пример: Align=top	**center**
ShowPlcInfo	Включает/выключает отображение информации о ПЛК (модель ЦП и версия СПО) (при некорректном вводе, применяется значение по умолчанию): – yes – включено; – no – выключено. Пример: ShowPlcInfo=no	**yes**
Секция [Syslog]:		
host	Настройка протокола syslog для перенаправления системных сообщений. Конфигурирование отправки сообщений на удаленный сервер осуществляется согласно описанию: Компьютер, принимающий сообщения – **host.** **host** = **hostname** [**protocol**] [**time**] [**severity** [**facilitity**[:**subst**]]*]*, где: – **hostname** – имя или IP-адрес ПК, принимающего сообщения syslog; – **protocol** – протокол передачи сообщений [RFC5424 / RFC3164], по умолчанию RFC3164; – **time** – выбор временного формата для отметки	**-**

Параметр	Описание	Значение по умолчанию
	сообщений [utc / local], по умолчанию local; – **severity** – уровень важности сообщений, отправляются только те сообщения , чья важность меньше указанного в параметре. Значение от 0 до 8 (по умолчанию 7): – ▸ 0 (**Emergency**) – чрезвычайная ситуация, система не может использоваться; – ▸ 1 (**Alert**) – тревога, требуются незамедлительные действия; – ▸ 2 (**Critical**) – критическая ситуация; – ▸ 3 (**Error**) – ошибка; – ▸ 4 (**Warning**) – предупреждение; – ▸ 5 (**Note**) - замечание, нормальная, но важная ситуация (состояние); – ▸ 6 (**Informational**) – информационное сообщение; – ▸ 7 (**Debug**) — отладочное сообщение; – ▸ 8 (**All**) - любые сообщения; – **facility** – источник информации, задается в виде четырех символов: имя или значение от 0000 до 0127. Можно задать несколько групп - **severity** [facility], где **severity** задается для указанного списка **facility**; – **subst** - подмена f**acility**, задается в виде 4х символов: имя или значение от 0000 до 0127. Значение **subst** отделяется от **facility** символом двоеточия ':', и не должно иметь других разделителей. Значение **subst** подменяет заданный в сообщении **facility**. Программы принимающие сообщения могут ограничивать возможное значение **facility**: 24 или 125 значениями. Подмена позволяет переназначать существующее **facility** сообщения. Если после **severity** не задан список **facility**, то данный **severity** применяется для всех не заданных ранее **facility**. Если после hostname не задается ограничений на **severity facility**, то на host будут оправляться все сообщения. Алгоритм оправки сообщения: 1) для текущего сообщения, выбрать его **facility**. Для выбранного **facility**, сравнить **severity** сообщения с указанным в настройках. 2) если **severity** сообщения меньше указанного **severity** для данного **facility**, то заменить в сообщении **facility** на заданное значение **subst** и	

Параметр	Описание	Значение по умолчанию
	отправить сообщение Значения facility, subst: ‣ 0000 (kern) - сообщения ядра; ‣ 0001 (user) - сообщения пользовательского уровня; ‣ 0002 (mail) - почтовая система; ‣ 0003 (daem) - системные службы (демоны); ‣ 0004 (auth) - сообщения, связанные с защитой и предоставлением полномочий; ‣ 0005 (sysl) - внутренние сообщения syslogd; ‣ 0006 (lprn) - подсистема печати (line printer); ‣ 0007 (news) - подсистема сетевых новостей (network news); ‣ 0008 (uucp) - подсистема UUCP; ‣ 0009 (cron) - часы (демон); ‣ 0010 (priv) - сообщения, связанные с защитой и предоставлением полномочий (private); ‣ 0011 (ftpd) - демон FTP; ‣ 0012…0015 - зарезервировано под дополнительные системные сообщения; ‣ 0016…0023 - зарезервировано для пользовательских сообщений;	
Syslog	Включение/выключение отправки сообщения по протоколу syslog (1 – включить(enable), 0/любое другое – отключить (disable)). Отключается при: – ошибках инициализации сокета, – отсутствии host, – отключении всех host (ошибки передачи). Пример: enableSyslog = 1	**0**
Repeater	Ретранслировать сообщения от внешних источников syslog (0 – отключить (disable), 1 – включить(enable)). Пример: enableRepeater = 1	**-**
maxSelectError	Максимальное количество непрерывных ошибок прослушивания сокета, до отключения ретранслятора. При удачном чтение данных счетчик ошибок сбрасывается	**10**
maxRecieveError	Максимальное количество непрерывных ошибок чтения с сокета. Если количество ошибок чтения достигнет maxRecieveError, то счетчик ошибок сбросится в 0, счетчик ошибок прослушки (select) увеличится. При удачном чтение данных счетчики ошибок сбрасывается	**100**

Параметр	Описание	Значение по умолчанию
maxHostError	Максимальное количество непрерывных ошибок инициализации host или посылки данных. Таким образом: – при переполнении количества ошибок, host отключается; – при отключении всех host, отключается передача данных по syslog (enableSyslog=0); – при удачной посылке счетчик ошибок сбрасывается	**300**
delayHostInitError	Задержка междунеудачными попытками инициализировать передачу данных на host, в мс	**10**
maxSyslogInitError	Максимальное количество попыток инициализировать syslog	**30**
delaySyslogInitError	Задержка между неудачными попытками инициализировать syslog, в мс	**100**
Дополнительные (добавляются при необходимости)		
Секция [Startup]:		
ServiceMode	Изменение режима работы сервисного режима на время загрузки контроллера	**Enable**
TouchScreenOnStartup	Изменение режима работы сенсорного экрана на время загрузки контроллера	**Enable**

ПРИЛОЖЕНИЕ Ж

Настройка конфигурационного файла ptp.conf

Конфигурационный файл находится в каталоге **/etc/ptp.conf** и содержит настройки синхронизации времени по протоколу PTP, доступные пользователю для редактирования.

Таблица Ж.1 – Параметры конфигурационного файла

Параметр	Описание	Значение по умолчанию
State	**Состояние синхронизации (включен/выключен)**. Задайте необходимое значение параметра **State**: – для включения синхронизации: State=Enable; – для выключения синхронизации: State=Disable	**Disable**
Mode	**Режим работы ptp на контроллере (Master/Slave)**. Задайте необходимое значение параметра **Mode**: – для ведущего контроллера: Mode=Master; – для ведомого контроллера: Mode=Slave	**Master**
LogLevel	**Уровень протоколирования**. Задайте уровень подробности журнала работы: – **Error** – ошибка; – **Warning** – предупреждение; – **Notice** – пояснение; – **Info** – информационные сообщения; – **Debug** – отладка	**Error**
Port	**Номер порта**. Задайте номер порта, с которого осуществляется подключение к другому контроллеру, для организации канала синхронизации. Например: Port=30	**30**

ПРИЛОЖЕНИЕ 3

Настройка конфигурационного файла netdump.conf

Конфигурационный файл находится в каталоге **/etc/netdump.conf** и содержит настройки управления журналированием сетевого трафика, доступные пользователю для редактирования.

Журналирование работы службы захвата трафика производится в лог-файл netdump.log (расположен в каталоге /logs/logger/user).

Таблица 3.1 – Параметры конфигурационного файла

Параметр	**Описание**	**Значение по умолчанию**
Секция [**config**]:		
portXX	Для включения журналирования без дополнительных фильтров задайте необходимое значение для секции с именем порта (где XX – номер порта): – для включения: portXX=on; – для выключения: portXX=off (любое значение, кроме on)	-
Секция [**portXX**]:		
filename	Имя выходного файла с трафиком. Выходной файл помещается в каталог /logs/netdump/ с расширением *.pcap или *.log	-
output_format	Тип выходного файла. Возможные значения: – text (человеко-читаемый вид, расширение файла *.log); – binary (бинарный вид, расширение файла *.pcap)	-
file_max_size	Ограничение размера выходного файла. При достижении максимального размера, создается новый. Задается в МБ	-
file_count	Ограничение количества файлов. При достижении ограничения, создается новый файл вместо самого старого. Работает только в связке с file_max_size.	-
packet_count	Ограничение количества отфильтрованных пакетов. При достижении ограничения, журналирование завершается. Автоматически выставляется следующее значение в секции: [config] portXX=off	-

Параметр	Описание	Значение по умолчанию
Журналирование трафика с фильтрами		
filter_type	Тип фильтра. Задается протокол для фильтрации. Значения: – **ether** – базовая сетевая технология Ethernet, в фильтре используется аппаратный MAC адрес; – **ip** – протокол IPv4; – **arp** – протокол ARP; – **rarp** – обратный протокол ARP; – **tcp** – протокол TCP; – **udp** – протокол UDP. Например: filter_type=ether	**-**
filter_dir	Направления пакета/кадра. Значения: – src – объект является отправителем; – dst – объект является получателем. При задании filter_dir необходимо задать и filter_arg. Например: filter_dir=src filter_arg=172.29.34.139	**-**
filter_arg	Дополнительные аргументы для фильтрации. Значение **filter_arg**: – если **filter_type=ip** (либо **arp/rarp**) - указать IP адрес для фильтрации по адресу (например: 172.29.34.139), либо MASK/MASKLEN для фильтрации по маске подсети (например: 172.29.34.0/24). Биты в MASK, которые не входят в MASKLEN, обязательно должны быть нулевыми; – если **filter_type=ether -** указать MAC адрес в формате 00:00:00:00:00:00; – если **filter_type= tcp (**либо **udp) –** указать номер порта (например: 20), либо диапазон портов (например: 0-20)	**-**

ПРИЛОЖЕНИЕ И

Обработка расширенных данных и событий модулей

На шине RegulBus

Модули дискретного ввода

Порядок работы с метками времени на модулях дискретного ввода:

1. Выставить опцию «метка времени» у необходимого канала модуля дискретного ввода;
2. Обработать массив событий из кода прикладной программы по примеру:

```
VAR
      ch  : INT;
      e   : INT;
      val : STRING;
      ts  : STRING;
      logMsgStr   : STRING;
      CMP_NAME    : STRING := 'EventHandlerExample';
      CMP_CLASSID : DWORD  := 16#109B2030;
      loggerCmp   : PsLog.Component(CMP_NAME, CMP_CLASSID, 1);
END_VAR
------------------------------------------------------------------------
FOR ch := 0 TO 31 DO
    IF DI_32_011.Events[ch].Count > 0 THEN
        FOR e := 0 TO DI_32_011.Events[ch].Count - 1 DO
            val := TO_STRING(DI_32_011.Events[ch].E[e].xValue);
            ts := PsTime.UTCTIMENS_TO_STRING(DI_32_011.Events[ch].E[e].uliTS,
            1, 1);
            logMsgStr := CONCAT(DI_32_011.Name, ': DiCh=');
            logMsgStr := CONCAT(logMsgStr, TO_STRING(ch));
            logMsgStr := CONCAT(logMsgStr, ' Value=');
            logMsgStr := CONCAT(logMsgStr, val);
            logMsgStr := CONCAT(logMsgStr, ' Ts=');
            logMsgStr := CONCAT(logMsgStr, ts);
            PsLog.stdlog.AddLogEntry(loggerCmp, PsLog.LogClass.LOG_INFO, 0, 0,
            logMsgStr);
        END_FOR
    DI_32_011.Events[ch].Clear();
    END_IF
END_FOR
```

Модули аналогового ввода R500 AI 08 242/ R500 AI 08 342

Объект модуля AI 08 242 содержит массив **channelData**, который состоит из объектов **PsRegulBusCore.FcmDataQueue**. Число элементов в массиве равно числу измерительных каналов модуля, то есть каждому измерительную каналу соответствует свой объект структуры **PsRegulBusCore.TFcmChannelData**. Данный объект представляет собой очередь, в которую со стороны модуля помещаются данные. Проверка на наличие данных в очереди производится при помощи свойства **isEmpty** (**FALSE** – очередь содержит данные, **TRUE** – очередь пустая). Извлечение данных из очереди производится методом **Get**, в который передается ссылка на структуру **PsRegulBusCore.TFcmChannelData** (в эту структуру метод **Get** поместит данные, с которыми в последующем можно работать). Данная структура содержит массив (буфер)

отчетов **samplesData** в кодах АЦП. Количество отчетов в массиве **samplesData** указывается в переменной **samplesCount** данной структуры. Максимально возможное количество отсчетов в буфере указывается в переменной **GVL_FCM.MAX_SAMPLES_NUM**.

Так же каждый буфер отсчетов имеет свою метку времени (переменная структуры **TimeStamp**). Два последовательно считанные буфера из одной очереди имеют разные метки времени.

Если не вычитывать данные из очереди, то со временем она переполнится и выставит свойство **IsOverflowed = TRUE.** Максимальный размер очереди 100 буферов. При помещении очередного буфера со стороны модуля в переполненную очередь приведет к удалению самого старого буфера из очереди на тот момент.

Ниже приведен пример кода для получения данных с модулей:

```
VAR CONSTANT
    CH_NUM : INT := 8;
END_VAR

VAR
    channelData : ARRAY [0..CH_NUM - 1] OF
PsIoDrvRegulBus.PsRegulBusCore.TFcmChannelData;
    CH : INT;
END_VAR

FOR ch := 0 TO CH_NUM - 1 DO
    WHILE NOT AI_08_242_1.channelsData[ch].IsEmpty DO
        IF AI_08_242_1.channelsData[ch].Get(channelData[ch]) THEN
            //-->Тут работать с данными структуры channelData
        END_IF
    END_WHILE
END_FOR
```

В примере производится вычитывание всех буферов со всех очередей измерительных каналов модуля ***AI_08_242_1*** (***AI_08_242_1*** – это объект модуля в дереве устройств). В цикле **FOR** последовательно перебираются все каналы модуля. В цикле **WHILE** производится вычитывание накопленных буферов из очереди при помощи метода **Get**, до тех пор, пока очередь не окажется пустой (**isEmpty = TRUE**). После метода **Get** вы можете разместить ваш код, отвечающий за обработку очередного считанного буфера из очереди канала.

Пользователь, помимо самого массива данных, может получать усредненное на массиве значение, вычисляющийся как среднее арифметическое. Количество значений, на котором производится усреднение, зависит от частоты дискретизации: время преобразования фиксировано и равно 10 мс, а частота изменяется от 1 до 10 кГц, поэтому количество значений в массиве изменяется от 10 до 100.

На шине RegulBus_OS

В драйвере шины RegulBus_OS получение событий от модулей AI, DI, DA реализовано через функцию обратного вызова – **callback**-функция. Все события, поступающие от любого из модулей, сохраняются во внутреннем буфере драйвера и поступают на обработку последовательно путем вызова **callback** – метода по каждому из событий.

Событие от модуля **DI** формируются по изменению состояния любого дискретного входа (переход из «0» в «1» или из «1» в «0») при условии, что у данного канала установлен параметр «Метка времени» и он не замаскирован.

Событие от модуля **AI** (AI 08 242 и AI 08 342) формируется для любого незамаскированного канала после того, как модуль сформировал буфер или часть буфера с отсчетами. При этом может возникнуть ситуация, когда несколько друг за другом идущих подряд событий для одного измерительного канала содержат буферы отсчетов с одинаковой меткой времени. В этом случае, данные с этих буферов необходимо рассматривать как данные с одного буфера отсчетов (если есть необходимость, то можно самостоятельно объединить данные с этих событий в буфер с одной меткой времени в той последовательности, в которой были сформированы данные события для конкретного измерительного канала).

Событие от модулей **DA** формируются только при работе в режиме **СИКН**. Генерация событий происходит для каждого частотного входа по изменению состояния любого дискретного входа при условии, что у него установлен параметр «Формирование события дискретного входа N» («по фронту» – формирование события происходит при переходе дискретного входа из «0» в «1», «по спаду» – формирование события происходит при переходе дискретного входа из «1» в «0») и выставлен соответствующий бит в параметре «Канал для проверки».

Уведомление приложения о доступности данных для обработки производится с использованием программного события (Event).

	**ВНИМАНИЕ!** Необходимо использовать примитивы синхронизации над данными приложения, в которые производится выгрузка из обработчика, т.к. обработчик вызывается асинхронно задачам приложения

Callback-функция (метод) принимает содержимое буфера как входной параметр. Callback-функции добавлены по каждому типу событий на всю шину сразу (Рисунок И.1).

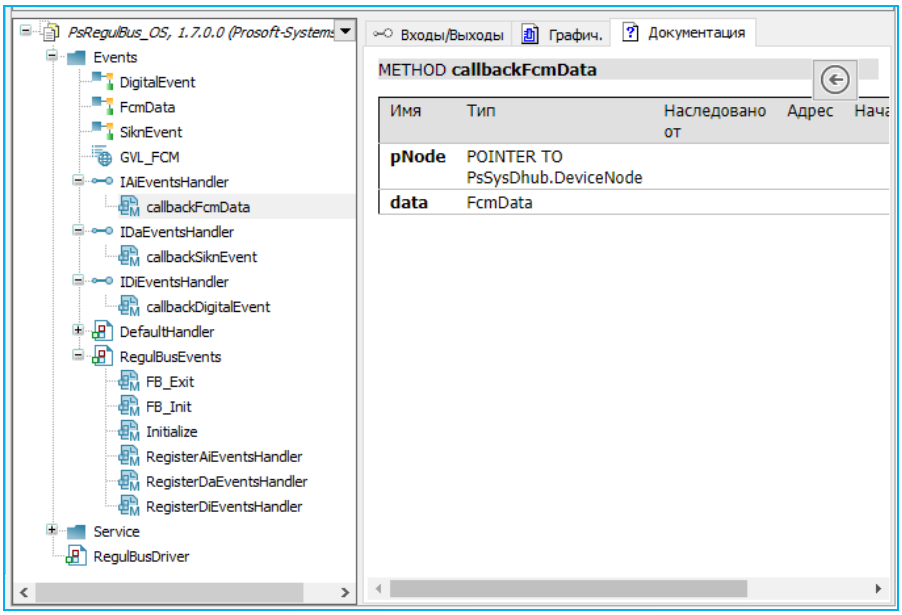


Рисунок И.1 – Callback-функции по каждому из событий

В обработчик поступает указатель на базовый класс для модулей и данные. Для применения обработчиков необходимо реализовать соответствующий интерфейс и зарегистрировать его в ФБ RegulBusEvents.

Для добавления интерфейса используйте контекстное меню, вызываемое нажатием правой клавишей мыши по **Application**, далее пройдите по пунктам: **Добавить POU⇨ Функциональный блок.** Установите флажок в поле **Implements** и нажмите на кнопку ... . Выберите необходимые интерфейсы, нажмите кнопку ***ОК.*** Задайте имя ФБ (в поле **Имя**) и проверьте **Язык реализации**… *Структурированный текст(ST),* нажмите кнопку ***Добавить*** (Рисунок И.2).

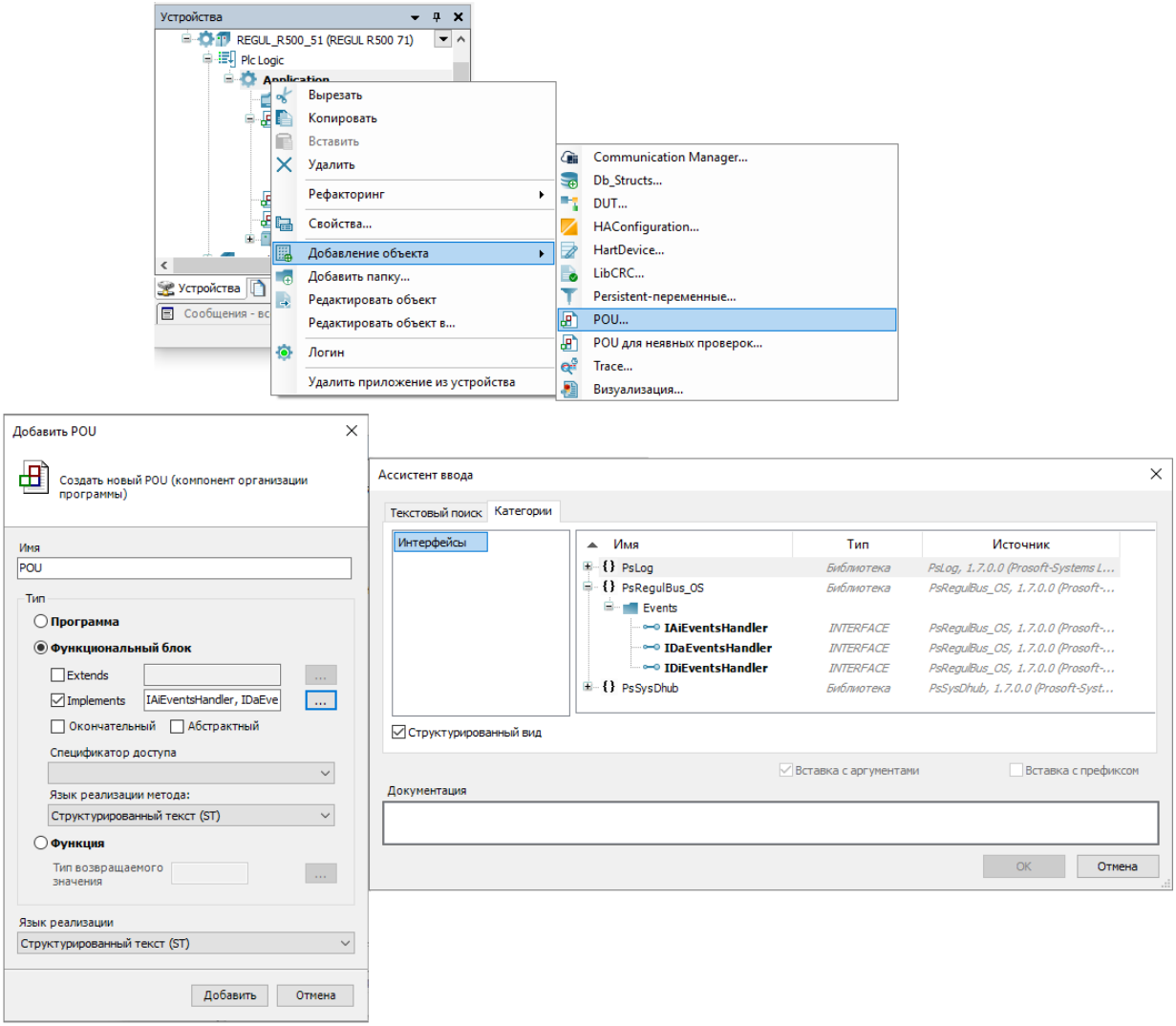


Рисунок И.2– Добавление интерфейсов

Пример кода ФБ пользовательского обработчика событий

Создайте список глобальных переменных

```
VAR_GLOBAL
      //Для логирования сообщений в callback-методах
      CMP_NAME    : STRING := 'EventHandlerExample';
      CMP_CLASSID : DWORD := 16#109B2030;
      loggerCmp   : PsLog.Component(CMP_NAME, CMP_CLASSID, 1);
END_VAR
```

Создайте функциональный блок обработчика событий

```
FUNCTION_BLOCK EventsHandler IMPLEMENTS IDiEventsHandler, IDaEventsHandler,
IAiEventsHandler
-------------------------------------------------------------------------
//Метод для обработки событий от модулей DI
METHOD callbackDigitalEvent
VAR_INPUT
      pNode : POINTER TO PsRegulBus_OS.PsSysDhub.DeviceNode;
      event : PsRegulBus_OS.DigitalEvent;
END_VAR
VAR
      logMsgStr : STRING;
```

```
END_VAR
-------------------------------------------------------------------------
logMsgStr := CONCAT(pNode^.Name, ': DiCh=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.usiChannel));
logMsgStr := CONCAT(logMsgStr, ' Value=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.xValue));
logMsgStr := CONCAT(logMsgStr, ' Ts=');
logMsgStr := CONCAT(logMsgStr, PsTime.UTCTIMENS_TO_STRING(event.uliTs, 1, 1));
PsLog.stdlog.AddLogEntry(loggerCmp, PsLog.LogClass.LOG_INFO, 0, 0, logMsgStr);
-------------------------------------------------------------------------

//Метод для обработки событий от модулей AI-модулей (AI08242, AI08342)
METHOD callbackFcmData
VAR_INPUT
      pNode : POINTER TO PsRegulBus_OS.PsSysDhub.DeviceNode;
      data  : PsRegulBus_OS.FcmData;
END_VAR
VAR
      logMsgStr : STRING;
END_VAR
-------------------------------------------------------------------------
logMsgStr := CONCAT(pNode^.Name, ': Ch=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(data.usiChannel));
logMsgStr := CONCAT(logMsgStr, ' Ts=');
logMsgStr := CONCAT(logMsgStr, PsTime.UTCTIMENS_TO_STRING(data.uliTs, 1, 1));
logMsgStr := CONCAT(logMsgStr, ' SamplesCnt=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(data.uiSamplesCount));
PsLog.stdlog.AddLogEntry(loggerCmp, PsLog.LogClass.LOG_INFO, 0, 0, logMsgStr);
-------------------------------------------------------------------------

//Метод для обработки событий от модулей DA
METHOD callbackSiknEvent
VAR_INPUT
      pNode : POINTER TO PsSysDhub.DeviceNode;
      event : SiknEvent;
END_VAR
VAR
      logMsgStr : STRING;
END_VAR
-------------------------------------------------------------------------
logMsgStr := CONCAT(pNode^.Name, ': FreqCh=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.byFreqChan));
logMsgStr := CONCAT(logMsgStr, ' DiCh=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.byDiChan));
logMsgStr := CONCAT(logMsgStr, ' Cnt=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.udiCount));
logMsgStr := CONCAT(logMsgStr, ' T=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.udiT));
logMsgStr := CONCAT(logMsgStr, ' Tau=');
logMsgStr := CONCAT(logMsgStr, TO_STRING(event.udiTau));
PsLog.stdlog.AddLogEntry(loggerCmp, PsLog.LogClass.LOG_INFO, 0, 0, logMsgStr);
```

Создайте программу

```
PROGRAM PLC_PRG
VAR
      xInit   : BOOL := FALSE;
      handler : EventsHandler;
END_VAR
-------------------------------------------------------------------

IF NOT xInit THEN
```

```
      //Регистрация пользовательских обработчиков событий
      Regul_Bus_OS.regulbus_events.RegisterAiEventsHandler(handler);
      Regul_Bus_OS.regulbus_events.RegisterDiEventsHandler(handler);
      Regul_Bus_OS.regulbus_events.RegisterDaEventsHandler(handler);
      xInit := TRUE;
END_IF
```

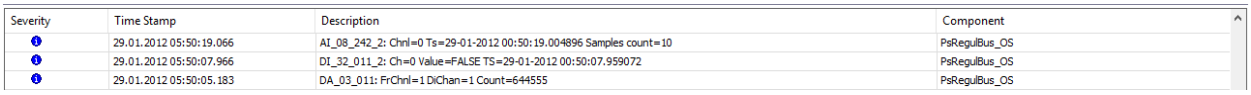
В примере указан универсальный функциональный блок для обработки событий всех трех типов модулей, где при генерации соответствующего события от модуля в callback-методе происходит запись информации в оперативный журнал.

Для каждого типа сообщений возможно создать отдельный функциональный блок обработки событий (но только один). Это говорит о том, что сообщения одного типа для разных модулей обрабатываются callback-методом одного обработчика. Поэтому, если необходимо идентифицировать сообщения от конкретного модуля, то в callback-методе воспользуйтесь конструкцией следующего вида (на примере модулей DI):

```
IF pNode = ADR(DI_32_011_1) THEN
      //Обработка события для модуля DI_32_011_1
ELSE IF pNode = ADR(DI_32_011_2) then
      //Обработка события для модуля DI_32_011_2
ELSE IF pNode = ADR(DI_32_011_N)
      //Обработка события для модуля DI_32_011_N
END_IF
```

Где DI_32_011_1, DI_32_011_2, DI_32_011_3 - это объекты модулей DI в дереве устройств.

Если пользователь не зарегистрировал обработчик событий для какого-нибудь типа модулей, а данные модули присутствуют в системе и генерируют события, то их обработка будет производиться обработчиками, назначенными по умолчанию (данные обработчики принимают события и формируют сообщения с информацией о них в оперативный журнал). Примеры сообщений из оперативного журнала от обработчиков событий приведены на рисунке И.3.



Severity	Time Stamp	Description	Component
ⓘ	29.01.2012 05:50:19.066	AI_08_242_2: Chnl=0 Ts=29-01-2012 00:50:19.004896 Samples count=10	PsRegulBus_OS
ⓘ	29.01.2012 05:50:07.966	DI_32_011_2: Ch=0 Value=FALSE TS=29-01-2012 00:50:07.959072	PsRegulBus_OS
ⓘ	29.01.2012 05:50:05.183	DA_03_011: FrChnl=1 DiChan=1 Count=644555	PsRegulBus_OS

Рисунок И.3– Сообщения от обработчиков событий (по умолчанию)

ВНИМАНИЕ!

При отладке в режиме **Онлайн** в callback –методах не работают точки останова. Для отладки кода внутри callback- метода следует использовать другие приемы отладки (например, запись в лог или промежуточные переменные)

ПРИЛОЖЕНИЕ К

Перечень модулей, поддерживаемых шиной RegulBus и/или RegulBus OS.

Таблица К.1

Обозначение серии/модуля	**Поддержка модулей по шине**	
	RegulBus	**RegulBus OS**
Все модули серии: – R600/R400/R200/R000 – R050	 Да Нет	 Да* Да
Все модули серии R500, кроме: – R500 CU 00 1X1 (-W, поддержка WEB-визуализации) – R500 EU 04 021 – R500 EU 04 031 – R500 CP 01 031 – R500 CP 04 041 – R500 AI 16 012 – R500 DI 32 012 – R500 DO 32 013	Да Нет Нет Нет Нет Нет Нет Нет Нет	Да* Да Да Да Да Да Да Да Да
* В зависимости от версии СПО модуля. Шина RegulBus OS не поддерживает модули с версией СПО ниже, чем: – **1.0.20.1** для модулей: R200 AO 02 011, R200 DO 04 021, R200 DO 08 011, R500 AO 08 011, R500 AO 08 021, R500 AO 08 031, R500 AS 08 011, R500 DO 16 021, R500 DO 32 011, R500 DS 32 011, R600 AO 08 011, R600 DO 32 011 – **1.0.31.0** для модулей: R200 AO 02 031, R500 DO 32 012, R500 DO 32 041, R500 DS 32 012		

ПРИЛОЖЕНИЕ Л

Библиотеки

ИНФОРМАЦИЯ

Начиная с СПО 1.7.2.0 всем EXTERNAL-функциям добавлен префикс компонента (например, TimeStampNow ⇨**PsLog**TimeStampNow)

PsJsonConvertions (PS_JSON до 1.6.0.0)

Компонент для JSON сериализации МЭК структур.

▸ Функциональные блоки

➢JsonReader

Функциональный блок для разбора строки в формате JSON.

Пример:

```
json_reader: JsonReader;
```

Метод cd

Переход к вложенному объекту/массиву.

Входной аргумент:

– относительный путь *rel_path* типа STRING.

Пример:

```
json_reader : JsonReader;
xResult : BOOL;


xResult := json_reader.ParseJson(ADR('{"name":"nested
bj","params":{"coords":{"x":10,"y":20}}}'));
IF xResult THEN
json_reader.cd('params');
json_reader.cd('coords');
json_reader.cd('y');
json_reader.cd('') // Перейти к корню
json_reader.cd('params.coords.x')
END_IF
```

Метод cdArr

Переход к элементу в массиве.

Входной аргумент:

– индекс внутри массива arr_ind типа UINT.

Пример:

```
json_reader : JsonReader;
```

```
xResult : BOOL;

xResult :=
json_reader.ParseJson(ADR('{"name":"Cars","CarArr":["Ford","BMW","Fiat"]}'));
IF xResult THEN
json_reader.cd('CarArr');
json_reader.cdArr(1);// "BMW"
END_IF
```

Метод cdUp

Переход к объекту/массиву уровнем выше.

Входной аргумент:

– отсутствует.

Пример:

```
json_reader : JsonReader;
xResult : BOOL;


xResult := json_reader.ParseJson(ADR('{"name":"nested
obj","params":{"coords":{"x":10,"y":20}}}'));
IF xResult THEN
json_reader.cd('params.coords.x');
json_reader.cdUp();
json_reader.cd('y');
END_IF
```

Метод Destroy

Запрос на освобождение внутренних данных.

Входной аргумент:

– отсутствует.

Пример:

```
json_reader : JsonReader;
xResult : BOOL;

xResult := json_reader.Destroy();
```

Метод Get_BOOL

Запрос на получение переменной типа BOOL.

Входные аргументы:

– путь к JSON переменной *path* типа STRING;

– ссылка на переменную dest типа REFERENCE TO BOOL.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
```

```
val : BOOL;

xRes := json_reader.Get_BOOL('path.val', val);
```

Метод Get_BYTE

Запрос на получение переменной типа BYTE.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO BYTE.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : BYTE;

xRes := json_reader.Get_BYTE('path.val', val);
```

Метод Get_DINT

Запрос на получение переменной типа DINT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO DINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : DINT;

xRes := json_reader.Get_DINT('path.val', val);
```

Метод Get_DWORD

Запрос на получение переменной типа DWORD.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO DWORD.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : DWORD;

xRes := json_reader.Get_DWORD('path.val', val);
```

Метод Get_INT

Запрос на получение переменной типа INT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO INT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : INT;

xRes := json_reader.Get_INT('path.val', val);
```

Метод Get_LINT

Запрос на получение переменной типа LINT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO LINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : LINT;

xRes := json_reader.Get_LINT('path.val', val);
```

Метод Get_LREAL

Запрос на получение переменной типа LREAL.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO LREAL.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : LREAL;

xRes := json_reader.Get_LREAL('path.val', val);
```

Метод Get_LWORD

Запрос на получение переменной типа LWORD.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO LWORD.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : LWORD;

xRes := json_reader.Get_LWORD('path.val', val);
```

Метод Get_REAL

Запрос на получение переменной типа REAL.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO REAL.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : REAL;

xRes := json_reader.Get_REAL('path.val', val);
```

Метод Get_SINT

Запрос на получение переменной типа SINT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO SINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : SINT;

xRes := json_reader.Get_SINT('path.val', val);
```

Метод Get_STRING

Запрос на получение переменной типа STRING.

Реализован для строк длиной не более 80 символов.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;

– ссылка на переменную *dest* типа REFERENCE TO STRING.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : STRING;

xRes := json_reader.Get_STRING('path.val', val);
```

Метод Get_STRING2

Запрос на получение переменной типа STRING определенной длины.

Реализован для строк произвольной длины.

Размер выходного буфера принимается в байтах, а длина строки в символах.

Входные аргументы:

– путь к JSON переменной *path* типа STRING;

– ссылка на переменную *dest* типа POINTER TO STRING;

– размер выходного буфера *size* типа DINT.

Возвращаемое значение:

– "-1", если произошла ошибка;

– "0", если пустая строка;

– ≥1, длина полученной строки типа INT.

Пример:

```
json_reader : JsonReader;
xRes : INT;
val : STRING(20);

xRes := json_reader.Get_STRING2('path.val', ADR(val), SIZEOF(val));
```

Метод Get_UDINT

Запрос на получение переменной типа UDINT.

Входные аргументы:

– путь к JSON переменной *path* типа STRING;

– ссылка на переменную *dest* типа REFERENCE TO UDINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : UDINT;

xRes := json_reader.Get_UDINT('path.val', val);
```

Метод Get_UINT

Запрос на получение переменной типа UINT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO UINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : UINT;

xRes := json_reader.Get_UINT('path.val', val);
```

Метод Get_ULINT

Запрос на получение переменной типа ULINT.

Входныеаргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO ULINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : ULINT;

xRes := json_reader.Get_ULINT('path.val', val);
```

Метод Get_USINT

Запрос на получение переменной типа USINT.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO USINT.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : USINT;

xRes := json_reader.Get_USINT('path.val', val);
```

Метод Get_WORD

Запрос на получение переменной типа WORD.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO WORD.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : WORD;

xRes := json_reader.Get_WORD('path.val', val);
```

Метод Get_WSTRING

Запрос на получение переменной типа WSTRING.

Реализован для строк длиной не более 80 символов.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- ссылка на переменную *dest* типа REFERENCE TO WSTRING.

Пример:

```
json_reader : JsonReader;
xRes : BOOL;
val : WSTRING;

xRes := json_reader.Get_WSTRING('path.val', val);
```

Метод Get_WSTRING2

Запрос на получение переменной типа WSTRING определенной длины.

Реализован для строк произвольной длины.

Размер выходного буфера принимается в байтах, а длина строки в символах.

Входные аргументы:

- путь к JSON переменной *path* типа STRING;
- указатель на переменную *dest* типа REFERENCE TO WSTRING;
- размер выходного буфера *size* типа DINT.

Возвращаемое значение:

- "-1", если произошла ошибка;
- "0", если пустая строка;
- ≥1, длина полученной строки типа INT.

Пример:

```
json_reader : JsonReader;
xRes : INT;
val : WSTRING(20);

xRes := json_reader.Get_STRING2('path.val', ADR(val), SIZEOF(val));
```

Метод ParseJson

Разбор строки формата JSON.

Входной аргумент:

– ссылка на строку *pJsonStr* типа POINTER TO STRING.

Пример:

```
json_reader : JsonReader;
xResult : BOOL;


xResult :=
json_reader.ParseJson(ADR('{"name":"Cars","CarArr":["Ford","BMW","Fiat"]}'));
```

➢JsonWriter

Функциональный блок для формирования строки в формате JSON.

Пример:

```
json_writer : JsonWriter;
xRes : BOOL;

xRes := json_writer.Clear();
```

Метод EndArray

Завершить формирование массива.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;


json_writer.StartArray();
json_writer.Key( 'lint_test1' );
json_writer.Val_LINT( 1 );
json_writer.Key( 'lint_test2' );
json_writer.Val_LINT( 2 );
json_writer.EndArray();
```

Метод EndObject

Завершить формирование объекта.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;


json_writer.StartObject();
json_writer.Key( 'lint_test' );
json_writer.Val_LINT( 3 );
json_writer.EndObject();
```

Метод GetResultJsonString

Получить сформированную JSON строку.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;
res : POINTER TO STRING;

res := json_writer.GetResultJsonString();
```

Метод GetResultSize

Получить размер сформированной JSON строки.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;
res : DINT;

res := json_writer.GetResultSize();
```

Метод Key

Добавить ключ к строке.

Входной аргумент:

– значение ключа *key_name* типа STRING.

Пример:

```
json_writer: JsonWriter;
```

```
json_writer.StartObject();
json_writer.Key( 'lint_test' );
json_writer.Val_LINT( 3 );
json_writer.EndObject();
```

Метод StartArray

Начать формирование массива.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;


json_writer.StartArray();
json_writer.Key( 'lint_test1' );
json_writer.Val_LINT( 1 );
json_writer.Key( 'lint_test2' );
json_writer.Val_LINT( 2 );
json_writer.EndArray();
```

Метод StartObject

Начать формирование объекта.

Входной аргумент:

– отсутствует.

Пример:

```
json_writer: JsonWriter;


json_writer.StartObject();
json_writer.Key( 'lint_test' );
json_writer.Val_LINT( 3 );
json_writer.EndObject();
```

Метод Val_BOOL

Добавить значение переменной типа BOOL к строке.

Входной аргумент:

– значение переменной *val* типа BOOL.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : BOOL;

json_writer.Key( 'test' );
```

```
xRes := json_writer.Val_BOOL(val);
```

Метод Val_BYTE

Добавить значение переменной типа BYTE к строке.

Входной аргумент:

– значение переменной *val* типа BYTE.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : BYTE;

json_writer.Key( 'test' );
xRes := json_writer.Val_BYTE(val);
```

Метод Val_DINT

Добавить значение переменной типа DINT к строке.

Входной аргумент:

– значение переменной *val* типа DINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : DINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_DINT(val);
```

Метод Val_DWORD

Добавить значение переменной типа DWORD к строке.

Входной аргумент:

– значение переменной *val* типа DWORD.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : DWORD;

json_writer.Key( 'test' );
xRes := json_writer.Val_DWORD(val);
```

Метод Val_INT

Добавить значение переменной типа INT к строке.

Входной аргумент:

– значение переменной *val* типа INT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : INT;

json_writer.Key( 'test' );
xRes := json_writer.Val_INT(val);
```

Метод Val_LINT

Добавить значение переменной типа LINT к строке.

Входной аргумент:

– значение переменной *val* типа LINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : LINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_LINT(val);
```

Метод Val_LREAL

Добавить значение переменной типа LREAL к строке.

Входной аргумент:

– значение переменной *val* типа LREAL.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : LREAL;

json_writer.Key( 'test' );
xRes := json_writer.Val_LREAL(val);
```

Метод Val_LWORD

Добавить значение переменной типа LWORD к строке.

Входной аргумент:

– значение переменной *val* типа LWORD.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
```

```
val : LWORD;

json_writer.Key( 'test' );
xRes := json_writer.Val_LWORD(val);
```

Метод Val_REAL

Добавить значение переменной типа REAL к строке.

Входной аргумент:

– значение переменной *val* типа REAL.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : REAL;

json_writer.Key( 'test' );
xRes := json_writer.Val_REAL(val);
```

Метод Val_SINT

Добавить значение переменной типа SINT к строке.

Входной аргумент:

– значение переменной *val* типа SINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : SINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_SINT(val);
```

Метод Val_STRING

Добавить значение переменной типа STRING к строке.

Входной аргумент:

– значение переменной *val* типа STRING.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : STRING;

json_writer.Key( 'test' );
xRes := json_writer.Val_STRING(val);
```

Метод Val_UDINT

Добавить значение переменной типа UDINT к строке.

Входной аргумент:

– значение переменной *val* типа UDINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : UDINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_UDINT(val);
```

Метод Val_UINT

Добавить значение переменной типа UINT к строке.

Входной аргумент:

– значение переменной *val* типа UINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : UINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_UINT(val);
```

Метод Val_ULINT

Добавить значение переменной типа ULINT к строке.

Входной аргумент:

– значение переменной *val* типа ULINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : ULINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_ULINT(val);
```

Метод Val_USINT

Добавить значение переменной типа USINT к строке.

Входной аргумент:

– значение переменной *val* типа USINT.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
```

```
val : USINT;

json_writer.Key( 'test' );
xRes := json_writer.Val_USINT(val);
```

Метод Val_WORD

Добавить значение переменной типа WORD к строке.

Входной аргумент:

– значение переменной *val* типа WORD.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : WORD;

json_writer.Key( 'test' );
xRes := json_writer.Val_WORD(val);
```

Метод Val_WSTRING

Добавить значение переменной типа WSTRING к строке.

Входной аргумент:

– значение переменной *val* типа WSTRING.

Пример:

```
json_writer: JsonWriter;
xRes : BOOL;
val : WSTRING;

json_writer.Key( 'test' );
xRes := json_writer.Val_WSTRING(val);
```

PsLed (PS_LED)

Компонент для управления индикацией и звуковой сигнализацией, получения сервисной информации о состоянии переключателей модуля центрального процессора контроллера.

Перечисления

➢ENUM_LED_MODE

Перечисление ENUM_LED_MODE типа BYTE содержит режимы работы индикатора, указанные в таблице Л.1.

Таблица Л.1 – Режимы работы индикатора

Режим	Значение	Описание режима
OFF	0	Индикатор выключен
ON	1	Индикатор включен
FLASH	2	Индикатор мигает
FAST_FLASH	3	Индикатор быстро мигает
FLASH_REVERSE	4	Индикатор мигает в противофазе индикатору FLASH
FAST_FLASH_REVERSE	5	Индикатор мигает в противофазе индикатору FAST_FLASH

➢**ENUM_LEDS**

Перечисление ENUM_LEDS типа BYTE содержит следующие виды индикаторов, указанные в таблице Л.2.

Таблица Л.2 – Виды индикаторов

Режим	Значение
RUN	0
RDD	1
HARDF	2
PROGF	3
USR1_GREEN	4
USR1_RED	5
USR2_GREEN	6
USR2_RED	7
USR3_GREEN	8
USR3_RED	9
Только для ПЛК Regul R500:	
B1_GREEN	10
B1_RED	11
B2_GREEN	12
B2_RED	13
MB1_GREEN	14
MB1_RED	15
MB2_GREEN	16
MB2_RED	17

Режим	Значение
USRLED4_GREEN	18

➢**ENUM_MBS**

Перечисление ENUM_LEDS типа USINT содержит положения ключа MBS, указанные в таблице Л.3. Используется только для ПЛК Regul R500.

Таблица Л.3 – Положения ключа MBS

Положение ключа MBS	Значение	Описание положения ключа MBS
UNKNOWN	0	Неизвестно
MBS1	1	Положение 1
MBS2	2	Положение 2
MBS3	3	Положение 3

Функции

➢ **PsLedBeep (Beep до 1.7.2.0)**

Воспроизводит единичный звуковой сигнал.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат *PsLedBeep* типа UDINT.

Пример:

```
xBeepEvent : BOOL;
      ...

IF xBeepEvent THEN
      xBeepEvent := FALSE;
      PsLedBeep();
END_IF
```

➢**PsLedGetKeyAutoStartApp (getKeyPosition до 1.7.2.0)**

Получение положения переключателя KEY.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "-1", если переключатель KEY не поддерживается;

– "1", если переключатель KEY находится в положении I;

– "2", если переключатель KEY находится в положении II;

– "3", если переключатель KEY находится в положении III.

Пример:

```
Key : SINT;
...
Key = PsLedGetKeyAutoStartApp()
```

➢**PsLedGetMBSPosition (getMBSPosition до 1.7.2.0)**

Получение положения переключателя MBS.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "-1", если переключатель MBS не поддерживается;

– "1", если переключатель MBS находится в положении I;

– "2", если переключатель MBS находится в положении II;

– "3", если переключатель MBS находится в положении III.

Пример:

```
Mbs : SINT;
...
Mbs = PsLedGetMBSPosition()
```

➢ **PsLedGetRun (getRun до 1.7.2.0)**

Получение положения переключателя RUN/STOP.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "-1", если переключатель не поддерживается;

– "0", переключатель находится в положении STOP;

– "1", переключатель находится в положении RUN.

Пример:

```
Run : SINT;
...
Run = PsLedGetRun()
```

➢**Get_SWKEYD (устаревшая)**

Получение положения переключателя KEY.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "TRUE", если переключатель KEY находится в положении II;

– "FALSE", если переключатель KEY находится в другом положении.

Пример:

```
Key : BOOL := PsLed.Get_SWKEYD();
```

➢Get_SWMB (устаревшая)

Получение положения переключателя MBS.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "TRUE", если переключатель MBS находится в положении II;

– "FALSE", если переключатель MBS находится в другом положении.

Пример:

```
isMbs2 : BOOL := PsLed.Get_SWMB();
```

➢Get_SWMB2 (устаревшая)

Получение положения переключателя MBS.

Входной аргумент:

– отсутствует.

Пример:

```
Mbs : ENUM_MBS := PsLed.Get_SWMB2();
```

➢Get_SWRUN (устаревшая)

Получение статуса положения переключателя RUN/STOP.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "TRUE", если переключатель находится в положении RUN;

– "FALSE", если переключатель находится в положении STOP.

Пример:

```
isRun : BOOL := PsLed.Get_SWRUN();
```

➢ **PsLedControl (LedControl до 1.7.2.0)**

Управление режимом работы индикаторов. Для управления доступны только пользовательские индикаторы с префиксом USR. Наличие и поддержку различных цветов индикаторов следует уточнить в документации используемого контроллера.

Входные аргументы:

– индикатор *led* из перечисления ENUM_LEDS;

– режим работы индикатора *mode* из перечисления ENUM_LED_MODE.

Возвращаемое значение:

– "0", если запрос выполнен успешно.

Примеры:

Зажечь индикатор LD1 зеленым:

```
PsLedControl(PsLed.ENUM_LEDS.USR1_GREEN, PsLed.ENUM_LED_MODE.ON);
PsLedControl(PsLed.ENUM_LEDS.USR1_RED, PsLed.ENUM_LED_MODE.OFF);
```

Зажечь индикатор LD2 красным:

```
PsLedControl(ENUM_LEDS.USR2_GREEN, ENUM_LED_MODE.OFF);
PsLedControl(ENUM_LEDS.USR2_RED, ENUM_LED_MODE.ON);
```

Быстро мигать желтым индикатором LD3:

```
PsLedControl(ENUM_LEDS.USR3_GREEN, ENUM_LED_MODE.FAST_FLASH);
PsLedControl(ENUM_LEDS.USR3_RED, ENUM_LED_MODE.FAST_FLASH);
```

➢ **PsLedSetBacklight (**setBacklight до 1.7.2.0)

Установка яркости подсветки сенсорного экрана контроллера R400.

Входной аргумент:

– значение яркости подсветки от 0 до 100% *backlight* типа INT.

Возвращаемое значение:

– "TRUE" – команда успешно выполнена;

– "FALSE" – устройство не поддерживает данную команду.

Установка яркости подсветки в 50% - PsLedSetBacklight(50), пример:

```
xBacklight : BOOL;
xResult : BOOL;
      ...

IF xBacklight THEN
      xBacklight := FALSE;
```

```
      xResult := PsLedSetBacklight(50);
END_IF
```

PsLog (PS_Log)

Компонент для ведения журнала работы контроллера.

Интерфейсы

ILogger

Интерфейс функционального блока журналирования.

Пример:

```
my_logger : PsLog.Logger('my_logger');
sup_logger : PsLog.LoggerDelegateDuplicateSuppressor(my_logger);
logger : PsLog.ILogger;
logger_type : INT := 1;
//
IF logger_type = 0 THEN
logger := my_logger;
ELSIF logger_type = 1 THEN
logger := sup_logger;
ELSE
logger := PsLog.stdlog;
END_IF
```

Метод AddLogEntry

Добавление сообщения в журнал.

Функциональные блоки

Component

Логический компонент системы журналирования.

Входные аргументы:

- уникальное имя компонента для текущего идентификатора *sCmpName* типа STRING;
- уникальный идентификатор компонента *udiCmpId* в диапазоне 16#109B3000 - 16#109B3FFF типа UDINT;
- версия компонента *udiCmpVersion* типа UDINT.

Пример:

```
cmp_test1 : PsLog.Component('test1_cmp', 16#109B3001, 1);
```

Метод FB_Init

Инициализация функционального блока Component.

Logger

Функциональный блок журналирования.

Входной аргумент:

– уникальное имя журнала *sLoggerName* типа STRING.

Пример:

```
Logger : PsLog.Logger('test_logger');
```

Метод AddLogEntry

Добавление сообщения в журнал.

Входные аргументы:

– ссылка на компонент *refCmp* типа REFERENCE TO Component;
– идентификатор типа сообщения (см. LogClass) *udiClassId* типа UDINT;
– идентификатор ошибки (см. CmpErrors) *udiErrorId* типа UDINT;
– идентификатор текста сообщения (см. CmpLog.LogAdd) *udiInfoId* типа UDINT;
– текст сообщения *sText* типа STRING(255).

Пример:

```
cmp_test1 : Component('test1_cmp', 16#109B3001, 1);
logger : Logger('test_logger');
logger.AddLogEntry(cmp_test1, PsLog.LogClass.LOG_INFO, 0, 0, 'Test info message');
```

Метод FB_Init

Инициализация функционального блока Logger.

➢LoggerDelegateDuplicateSuppressor

Функциональный блок для подавления дублирования.

Входной аргумент:

– объект наследованный от **ILogger**.

Используется для игнорирования запросов на запись дублированных сообщений в журнал. Максимальное количество одинаковых сообщений подряд: 1.

Пример:

```
Logger : PsLog.Logger('test_logger');
logger_sup : PsLog.LoggerDelegateDuplicateSuppressor(logger);
```

Метод AddLogEntry

Добавление сообщения в журнал.

Сообщение не будет записано в журнал, если оно дублируется второй раз.

Входные аргументы:

- ссылка на компонент *refCmp* типа REFERENCE TO Component;
- идентификатор типа сообщения (см. LogClass) *udiClassId* типа UDINT;
- идентификатор ошибки (см. CmpErrors) *udiErrorId* типа UDINT;
- идентификатор текста сообщения (см. CmpLog.LogAdd) *udiInfoId* типа UDINT;
- текст сообщения *sText* типа STRING(255).

Пример:

```
cmp_test1 : PsLog.Component('test1_cmp', 16#109B3001, 1);
logger : PsLog.Logger('test_logger');
logger_sup : PsLog.LoggerDelegateDuplicateSuppressor(logger);
logger_sup.AddLogEntry(cmp_test1, PsLog.LogClass.LOG_INFO, 0, 0, 'Test info
message');
```

Метод FB_Init

Инициализация функционального блока **LoggerDelegateDuplicateSuppressor**.

➢ **LoggerStd**

Функциональный блок для журналирования в стандартный журнал (StdLogger).

Входной аргумент:

- отсутствует.

Пример:

```
Logger : LoggerStd;
```

Метод AddLogEntry

Добавление сообщения в журнал.

Входные аргументы:

- ссылка на компонент *refCmp* типа REFERENCE TO Component;
- идентификатор типа сообщения (см. LogClass) *udiClassId* типа UDINT;
- идентификатор ошибки (см. CmpErrors) *udiErrorId* типа UDINT;
- идентификатор текста сообщения (см. CmpLog.LogAdd) *udiInfoId* типа UDINT;
- текст сообщения *sText* типа STRING(255).

Пример:

```
cmp_test1 : PsLog.Component('test1_cmp', 16#109B3001, 1);
stdlog.AddLogEntry(cmp_test1, PsLog.LogClass.LOG_INFO, 0, 0, 'Test info message');
```

Функции

➢ **PsLogLoggerAddLogTsEntry (LoggerAddLogTsEntry до 1.7.2.0)**

Добавление сообщения в журнал с заданной меткой времени.

Входные аргументы:

- функция обработки на открытый журнал (см.CmpLog.LogCreate) *hLog* типа RTS_IEC_HANDLE;
- метка времени в сотнях наносекунд *uliTs* типа ULINT;
- ID компонента *udiCmpId* типа UDINT;
- идентификатор типа сообщения (см. LogClass) *udiClassId* типа UDINT;
- идентификатор ошибки (см. CmpErrors) *udiErrorId* типа UDINT;
- идентификатор текста сообщения (см. CmpLog.LogAdd) *udiInfoId* типа UDINT;
- текст сообщения *sText* типа STRING(255).

Возвращаемое значение:

- результат добавления сообщения *PsLogLoggerAddLogTsEntry* типа RTS_IEC_RESULT.

Пример:

```
PsLogLoggerAddLogTsEntry(0, PsLog.TimeStampNow(), 16#109B3001,
PsLog.LogClass.LOG_INFO, 0, 0, 'Test info message');
```

➢ PsLogTimeStampNow (**TimeStampNow до 1.7.2.0)**

Получение текущей метки времени. Для перевода метки времени используйте константы (см. Ps_LogGlobal).

Возвращаемое значение:

- метка времени в сотнях наносекунд *PsLogTimeStampNow* типа ULINT.

Пример:

```
uliTs       : ULINT;
// ...
uliTs := PsLogTimeStampNow();
```

➢PS_LogGlobal

Константы и глобальный экземпляр стандартного журнала stdlog (LoggerStd).

Константы:

- количество отсчетов метки времени в секунде *S_TO_TS* типа ULINT (10000000);
- количество отсчетов метки времени в миллисекунде *MS_TO_TS* типа ULINT (10000);
- количество отсчетов метки времени в микросекунде US_TO_TS типа ULINT (10);
- количество наносекунд в 1 отсчете метки времени TS_TO_NS типа ULINT (100).

PsMySQLClient (PS_MysqlClient)

Универсальный клиент MySQL.

Структуры

➢TMysqlSettings

Структура с настройками MySQL клиента.

Содержит переменные, указанные в таблице Л.4.

Таблица Л.4 – Переменные TMysqlSettings

Переменная	Тип	Описание
username	STRING	Имя пользователя
password	STRING	Пароль пользователя
host	STRING	Адрес хоста
port	UDINT	Номер порта
store_result	BOOL	Параметр указывает на процесс обработки запроса: – **FALSE** – результат запроса SQL будет подготовлен, но не будет сохраняться, доступ к результатам путем вызова функции Result(); – **TRUE** – результат запроса SQL будет подготовлен и полностью сохранен в SQLResult

Функции

➢EscapeString

Функция используется для создания допустимой SQL-строки в формате ASCII, которую можно использовать в команде SQL.

Входные аргументы:

– исходная строка *from_str* типа POINTER TO STRING;

– возвращаемая строка *to_str* типа POINTER TO STRING;

– размер строки *to_size* типа INT.

Возвращаемое значение:

– > 0 – длина кодированной строки в INT;

– < 0 – требуемая длина (если to_size меньше необходимой длины);

– = 0 – ошибка кодирования.

Пример:

```
Test := EscapeString_test(origin_str, return_str, size_str);
```

➢EscapeWString

Функция используется для создания допустимой SQL-строки в формате Unicode, которую можно использовать в команде SQL.

Входные аргументы:

– исходная строка *from_str* типа POINTER TO STRING;

– возвращаемая строка *to_str* типа POINTER TO STRING;

– размер строки *to_size* типа INT.

Возвращаемое значение:

– > 0 – длина кодированной строки в INT;

– < 0 – требуемая длина (если to_size меньше необходимой длины);

– = 0 – ошибка кодирования.

Пример:

```
Test := EscapeWString_test(origin_wstr, return_wstr, size_str);
```

▸ Функциональные блоки

➢TMysqlClient

Предоставляет возможность работы MySQL клиента с MySQL сервером.

Пример:

```
VAR
test_MysqlClient : TMysqlClient;
return_str : POINTER TO STRING;
error_test : BOOL;
done_test : BOOL;
mysql_test : PsSqlQuery.MySql;
END_VAR
...
Test_MysqlClient(xExecute := TRUE, strSQLQuery := return_str, xError =>
error_test, xDone => done_test, mysql => mysql_test);
```

где:

– xExecute – разрешение выполнения;

– strSQLQuery – SQL – запрос;

– xError – возвращаемое значение наличия ошибки;

– xDone – возвращаемое значение успешного выполнения;

– mysql – база данных MySql.

Метод FetchNextRow

Метод FetchNextRow вызывается для того, чтобы получить следующую строку.

Пример:

```
VAR
test_mysql : TMysqlClient;
test_row : POINTER TO POINTER TO STRING;
END_VAR
...
test_row := test_mysql.FetchNextRow();
```

Свойство ConnectionSettings

Настройки соединения.

Пример:

```
VAR
test_TMysqlClient : TMysqlClient;
test_Settings : TMysqlSettings;
END_VAR

...

test_Settings.username := 'username';
test_Settings.password := 'password';
test_Settings.host := '192.168.1.1';
test_Settings.port := 0;
test_MysqlClient.ConnectionSettings := ADR(test_Settings);
```

Свойство ErrorCode

Получение кода ошибки.

Пример:

```
VAR
test_TMysqlClient : TMysqlClient;
test_ErrorCode : DINT;
END_VAR
...
test_ErrorCode := test_TMysqlClient.ErrorCode;
```

Свойство NumFields

Получение количества полей в результирующем наборе.

Пример:

```
VAR
test_TMysqlClient : TMysqlClient;
test_NumFields : DINT;
END_VAR
...
test_NumFields := test_TMysqlClient.NumFields;
```

PsPlcInfo (PS_PlcInfo)

Компонент доступа к системной информации ПЛК.

Структуры

➢hdd_info_t (устаревшая)

Структура с информацией об использовании HDD.

Содержит переменные, указанные в таблице Л.5.

Таблица Л.5 – Переменные hdd_info_t (hdd_info2_t, hdd_info3_t, hdd_info4_t)

Переменная	Тип		Описание
	hdd_info_t	**hdd_info2_t, hdd_info3_t, hdd_info4_t**	
mountpoint	STRING(79)	STRING(79)	Точка монтирования
used_mb	LREAL	REAL	Использовано мегабайт
total_mb	LREAL	REAL	Всего мегабайт

➢hdd_info2_t (устаревшая)

Структура с информацией об использовании HDD.

Содержит переменные, указанные в таблице Л.5.

➢hdd_info3_t (устаревшая)

Структура с информацией об использовании HDD.

Содержит переменные, указанные в таблице Л.5.

➢hdd_info4_t

Структура с информацией об использовании HDD.

Содержит переменные, указанные в таблице Л.5.

➢interface_info_t (устаревшая)

Структура с информацией о сетевых интерфейсах.

Содержит переменные, указанные в таблице Л.6.

Таблица Л.6 – Переменные interface_info_t (interface_info3_t, interface_info4_t)

Переменная	Тип	Описание
name	STRING(29)	Имя интерфейса
rx_pkts	ULINT	Количество принятых пакетов
rx_bytes	ULINT	Количество принятых байт

Переменная	Тип	Описание
tx_pkts	ULINT	Количество отправленных пакетов
tx_bytes	ULINT	Количество отправленных байт
align_errors	UDINT	Ошибки выравнивания
fcs_errors	UDINT	Ошибки CRC
link	BOOL	Состояние link

➢**interface_info3_t (устаревшая)**

Структура с информацией о сетевых интерфейсах.

Содержит переменные, указанные в таблице Л.6.

➢**interface_info4_t**

Структура с информацией о сетевых интерфейсах.

Содержит переменные, указанные в таблице Л.6.

➢**lemt_dynamic_info_t (устаревшая)**

Структура с динамической информацией о процессоре.

Содержит переменные, указанные в таблице Л.7

Таблица Л.7 – Переменные lemt_dynamic_info_t

Переменная	Тип	Описание
WDT_remain	UINT	Обратный счетчик WDT (секунд)
Initial_timeout_WDT	UINT	Начальное значение WDT (секунд)
TotalUpTimeMinutes	UINT	Общее время наработки (минут)
SincePowerUpSeconds	UINT	Текущее время работы (секунд)
SMCFlags	BYTE	Флаги
CPUTemp	lemt_temp_stat_t	Структура с статистикой по температуре процессора
BoardTemp	lemt_temp_stat_t	Структура с статистикой по температуре платы
Voltages	voltage_t	Структура с информацией о напряжении
MainCurrent_mA	REAL	Общее потребление

➢**lemt_static_info_t (устаревшая)**

Структура с общей информацией о процессоре.

Содержит переменные, указанные в таблице Л.8.

Таблица Л.8 – Переменные lemt_static_info_t

Переменная	Тип	Описание
PowerCycles	UDINT	Счетчик включений
BootReason	STRING[19]	Причина последнего перезапуска
BoardPartNumber	STRING[19]	Номер партии
BoardSerialNumber	STRING[19]	Серийный номер
BIOSVersion	STRING[19]	Версия BIOS
BoardTestDate	STRING[19]	Дата тестирования

➢**lemt_temp_stat_t (устаревшая)**

Структура с статистикой по температуре.

Содержит переменные, указанные в таблице Л.9.

Таблица Л.9 – Переменные lemt_temp_stat_t

Переменная	Тип	Описание
current	REAL	Текущее значение
startup	SINT	Значение в момент запуска
minimal	SINT	Минимальное зарегистрированное значение
maximal	SINT	Максимальное зарегистрированное значение

➢**ram_info_t (устаревшая)**

Структура с информацией о RAM.

Содержит переменные, указанные в таблице Л.10.

Таблица Л.10 – Переменные ram_info_t (ram_info3_t)

Переменная	Тип	Описание
used_mb	LREAL	Использовано мегабайт
total_mb	LREAL	Всего мегабайт

➢**ram_info3_t (устаревшая)**

Структура с информацией о RAM.

Содержит переменные, указанные в таблице Л.10.

➢**ram_info4_t**

Структура с информацией о RAM.

Содержит переменные, указанные в таблице Л.10.

➢sys_info_t (устаревшая)

Структура с обобщенной информацией о контроллере.

Содержит переменные, указанные в таблице Л.11.

Таблица Л.11 – Переменные sys_info_t

Переменная	Тип	Описание
version	STRING(49)	Версия прошивки
cpu_load	LREAL	Загрузка ЦП
ram	ram_info_t	Информация о RAM
hdd_cnt	DINT	Количество HDD
hdd_info	ARRAY [0..1] OF hdd_info_t	Информация о HDD
net_itf_cnt	DINT	Количество сетевых интерфейсов
net_itf_info	ARRAY [0..5] OF interface_info_t	Информация о сетевых интерфейсах

➢sys_info2_t (устаревшая)

Структура с обобщенной информацией о контроллере.

Содержит переменные, указанные в таблице Л.12.

Таблица Л.12 – Переменные sys_info2_t

Переменная	Тип	Описание
version	STRING(49)	Версия прошивки
cpu_cnt	UDINT	Количество ядер CPU
cpu_load	ARRAY [0..31] OF UDINT	Загрузка ЦП
ram	ram_info_t	Информация о RAM
hdd_cnt	UDINT	Количество HDD
hdd_info	ARRAY [0..2] OF hdd_info2_t	Информация о HDD
net_itf_cnt	UDINT	Количество сетевых интерфейсов
net_itf_info	ARRAY [0..5] OF interface_info_t	Информация о сетевых интерфейсах

➢sys_info3_t (устаревшая)

Структура с обобщенной информацией о контроллере.

Содержит переменные, указанные в таблице Л.13.

Таблица Л.13 – Переменные sys_info3_t

Переменная	Тип	Описание
version	STRING(49)	Версия прошивки
cpu_cnt	UDINT	Количество ядер CPU
cpu_load	ARRAY [0..31] OF UDINT	Загрузка ЦП
ram	ram_info3_t	Информация о RAM
hdd_cnt	UDINT	Количество HDD
hdd_info	ARRAY [0..2] OF hdd_info3_t	Информация о HDD
net_itf_cnt	UDINT	Количество сетевых интерфейсов
net_itf_info	ARRAY [0..5] OF interface_info3_t	Информация о сетевых интерфейсах

➢**sys_info4_t**

Структура с обобщенной информацией о контроллере.

Содержит переменные, указанные в таблице Л.14.

Таблица Л.14 – Переменные sys_info4_t

Переменная	Тип	Описание
version	STRING(49)	Версия прошивки
cpu_cnt	UDINT	Количество ядер CPU
cpu_load	ARRAY [0..31] OF UDINT	Загрузка ЦП
ram	ram_info4_t	Информация о RAM
hdd_cnt	UDINT	Количество HDD
hdd_info	ARRAY [0..2] OF hdd_info4_t	Информация о HDD
net_itf_cnt	UDINT	Количество сетевых интерфейсов
net_itf_info	ARRAY [0..12] OF interface_info4_t	Информация о сетевых интерфейсах

➢**voltage_t (устаревшая)**

Структура с информацией о напряжении.

Содержит переменные, указанные в таблице Л.15.

Таблица Л.15 – Переменные voltage_t

Переменная	Тип	Описание
cpu_core_V	REAL	CPU Core
cpu_graphic_V	REAL	CPU Graphic
chipset_core_V1_22	REAL	Chipset Core (1.22V)
V1_05	REAL	1.0V
V5	REAL	5.0V
V3_3	REAL	3.3V
V1_8	REAL	1.8V

➢**MD5**

Содержит переменную типа *Array[0..15] of BYTE* со значением результата вычисления по алгоритму *MD5.*

➢SHA256

Содержит переменную типа *Array[0..31] of BYTE* со значением результата вычисления по алгоритму *SHA256.*

Функции

➢ **PsPlcInfoGetFileInfo (GetFileInfo до 1.7.2.0)**

Запрос на получение информации о файле.

Входные аргументы:

– имя файла *FileName* типа STRING;

– контрольное значение *Crc2* типа UDINT;

– размер файла *FileSize* типа UDINT.

Функция принимает на вход имя файла, и если результат запроса успешен, то после исполнения будут заполнены crc и size запрашиваемого файла.

Пример:

```
Res : RTS_IEC_RESULT;
FileName : STRING := 'test.txt';
Crc2 : UDINT;
FileSize: UDINT;
```

```
...
Res := PsPlcInfoGetFileInfo(FileName, Crc2, FileSize);
```

➢ **PsPlcInfoGetFileHashMD5 (GetFileHashMD5 до 1.7.2.0)**

Вычисление хэш-функции файла по алгоритму MD5.

Входные аргументы:

– имя файла *FileName* типа REFERENCE TO STRING;

– значение хэш-функции *hash* типа REFERENCE TO MD5.

Возвращаемое значение (код ошибки):

– ERR_FAILED - ошибка при обработке файла;

– ERR_NO_OBJECT - файл не найден;

– ERR_NO_ACCESS_RIGHTS - отсутствует разрешение на чтение файла;

– ERR_FILE_ERROR - ошибка открытия файла.

Функция принимает на вход имя файла, и при успешном завершении заполняет *hash*.

Пример:

```
Res      : RTS_IEC_RESULT;
FileName : STRING := 'test.txt';
hash     : MD5;

Res := PsPlcInfoGetFileHashMD5(FileName, hash);
```

➢ **PsPlcInfoGetFileHashSHA256 (GetFileHashSHA256 до 1.7.2.0)**

Вычисление хэш-функции файла по алгоритму SHA-256.

Входные аргументы:

– имя файла *FileName* типа REFERENCE TO STRING;

– значение хэш-функции *hash* типа REFERENCE TO SHA256.

Возвращаемое значение (код ошибки):

– ERR_FAILED - ошибка при обработке файла;

– ERR_NO_OBJECT - файл не найден;

– ERR_NO_ACCESS_RIGHTS - отсутствует разрешение на чтение файла;

– ERR_FILE_ERROR - ошибка открытия файла.

Функция принимает на вход имя файла, и при успешном завершении заполняет *hash*.

Пример:

```
Res      : RTS_IEC_RESULT;
```

```
FileName : STRING := 'test.txt';
hash     : SHA256;

Res := PsPlcInfoGetFileHashSHA256(FileName, hash);
```

➢**GetHwPPSInfo (устаревшая)**

Запрос на получение полной информации о процессоре.

Входной аргумент:

– структура *info* типа ppshw_info_t с полной информацией о процессоре.

Пример:

```
Res : RTS_IEC_RESULT;
Info : REGUL_PLC_INFO.ppshw_info_t;
...
Res := REGUL_PLC_INFO.GetHwPPSInfo(Info);
```

➢**GetLemtDynamicInfo (устаревшая)**

Запрос на получение динамической информации о процессоре.

Входной аргумент:

– структура с динамической информацией о процессоре *DynInfo* типа lemt_dynamic_info_t.

Пример:

```
Res : RTS_IEC_RESULT;
DynInfo : REGUL_PLC_INFO.lemt_dynamic_info_t;
...
Res := REGUL_PLC_INFO.GetLemtDynamicInfo(DynInfo);
```

➢**GetLemtSerialNumber (устаревшая)**

Запрос на получение серийного номера процессора.

Входной аргумент:

– серийный номер процессора *SerialNumber* типа LINT.

Пример:

```
Res : RTS_IEC_RESULT;
SerialNumber : LINT;
...
Res := REGUL_PLC_INFO.GetLemtSerialNumber(SerialNumber);
```

➢**GetLemtSerialNumber2 (устаревшая)**

Запрос на получение серийного номера процессора.

Входной аргумент:

– серийный номер процессора *SerialNumber* типа STRING.

Пример:

```
Res : RTS_IEC_RESULT;
SerialNumber : STRING;
...
Res := REGUL_PLC_INFO.GetLemtSerialNumber2(SerialNumber);
```

➢**GetLemtStatInfo (устаревшая)**

Запрос на получение общей информации о процессоре.

Входной аргумент:

– структура с общей информацией о процессоре *StatInfo* типа lemt_static_info_t.

Пример:

```
Res : RTS_IEC_RESULT;
StatInfo : REGUL_PLC_INFO.lemt_static_info_t;
...
Res := REGUL_PLC_INFO.GetLemtStatInfo(StatInfo);
```

➢ **PsPlcInfoGetNetIntVal (GetNetIntVal до 1.7.2.0)**

Запрос целочисленного значения параметра сетевого интерфейса.

Входной аргумент:

– индекс сетевого интерфейса *netIndex* типа *UDINT*, от 0 до (sys_info3_t.net_itf_cnt - 1);

– имя параметра *paramName* типа STRING (align_errors, fcs_errors, link, media_rate, roc_errors, ruc_errors, rx_bytes, rx_percent_rate, rx_pkts, rx_rate, sqe_errors, sym_errors, tnc_errors, tx_bytes, tx_percent_rate, tx_pkts, tx_rate, txd_errors);

– ссылка на возвращаемый параметр *outValue* типа *REFERENCE TO UDINT*.

Возвращаемое значение:

– **ERR_OK**, если запрос успешен, то *outValue* содержит значение запрашиваемого параметра;

– **ERR_TYPE_MISMATCH**, если запрос успешен, но значение параметра не является числом (*outValue* не изменен);

– **ERR_FAILED**, ошибка при запросе параметра (*outValue* не изменен).

Пример:

```
res : RTS_IEC_RESULT;
netIndex : BYTE;
paramName : STRING;
value : UDINT;
...
netIndex := 0;
paramName := 'media_rate';
res := PsPlcInfoGetNetIntVal(netIndex, paramName, value);
```

➢ **PsPlcInfoGetNetStringVal (GetNetStringVal до 1.7.2.0)**

Запрос строкового значения параметра сетевого интерфейса.

Входной аргумент:

- индекс сетевого интерфейса *netIndex* типа *UDINT*, от 0 до (sys_info3_t.net_itf_cnt - 1);
- имя параметра *paramName* типа STRING (align_errors, fcs_errors, link, media_rate, name, roc_errors, ruc_errors, rx_bytes, rx_percent_rate, rx_pkts, rx_rate, sqe_errors, sym_errors, tnc_errors, tx_bytes, tx_percent_rate, tx_pkts, tx_rate, txd_errors);
- возвращаемый параметр *outValue* типа *REFERENCE TO STRING;*
- максимальная длина возвращаемой строки *maxLen* типа *UDINT*.

Возвращаемое значение:

- **ERR_OK**, если запрос успешен, то *outValue* содержит значение запрашиваемого параметра;
- **ERR_FAILED**, ошибка при запросе параметра (outValue не изменен).

Пример:

```
res : RTS_IEC_RESULT;
netIndex : BYTE;
paramName : STRING;
value : STRING;
...
netIndex := 0;
paramName := 'name';
res := PsPlcInfoGetNetStringVal(netIndex, paramName, value, sizeof(value));
```

➢GetRtsName

Получение имени используемой среды исполнения.

Входной аргумент:

- отсутствует.

Возвращаемое значение:

- имя используемой среды исполнения типа STRING(80).

Пример:

```
RtsName : STRING(80) := REGUL_PLC_INFO.GetRtsName();
```

➢GetRtsVersion

Получение версии используемой среды исполнения.

Входной аргумент:

- отсутствует.

Возвращаемое значение:

- имя среды исполнения вида XXXXXXXX типа DWORD.

Пример:

```
RtsVersion : DWORD := REGUL_PLC_INFO.GetRtsVersion();
```

➢**GetRtsVersion2**

Получение версии используемой среды исполнения.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– имя среды исполнения вида XX.XX.XX.XX типа STRING(80).

Пример:

```
RtsVersion : STRING(80) := REGUL_PLC_INFO.GetRtsVersion2();
```

➢ **GetSysInfo (устаревшая)**

Запрос обобщенной информации о контроллере.

Входной аргумент:

– структура с обобщенной информацией о контроллере *Sysinfo* типа sys_info_t.

Пример:

```
Res : RTS_IEC_RESULT;
Sysinfo : REGUL_PLC_INFO.sys_info_t;
...
Res := REGUL_PLC_INFO.GetSysInfo(Sysinfo);
```

➢ **GetSysInfo2 (устаревшая)**

Запрос обобщенной информации о контроллере.

Входной аргумент:

– структура с обобщенной информацией о контроллере *Sysinfo* типа sys_info2_t.

Пример:

```
Res : RTS_IEC_RESULT;
Sysinfo : REGUL_PLC_INFO.sys_info2_t;
...
Res := REGUL_PLC_INFO.GetSysInfo2(Sysinfo);
```

➢**GetSysInfo3 (устаревшая)**

Запрос обобщенной информации о контроллере.

Входной аргумент:

– структура с обобщенной информацией о контроллере *Sysinfo* типа sys_info3_t.

Пример:

```
Res : RTS_IEC_RESULT;
Sysinfo : REGUL_PLC_INFO.sys_info3_t;
...
Res := REGUL_PLC_INFO.GetSysInfo3(Sysinfo);
```

➢PsPlcInfoGetSysInfo4 (GetSysInfo4 до 1.7.2.0)

Запрос обобщенной информации о контроллере.

Входной аргумент:

– структура с обобщенной информацией о контроллере *Sysinfo* типа sys_info4_t.

Пример:

```
Res : RTS_IEC_RESULT;
Sysinfo : REGUL_PLC_INFO.sys_info4_t;
...
Res := PsPlcInfoGetSysInfo4(Sysinfo);
```

➢ PsPlcInfoHwmonGetFloatVal (HwmonGetFloatVal до 1.7.2.0)

Запрос параметра с типом REAL.

Входные аргументы:

– имя параметра *ValName* типа STRING;

– ссылка на возвращаемый параметр *Val* типа REFERENCE TO REAL.

Возвращаемое значение:

– TRUE, если запрос успешен, *outVal* содержит значение запрашиваемого параметра;

– FALSE, если произошла ошибка, *outVal* не изменен.

Пример:

```
Res : BOOL;
ValName : STRING := 'VAR_NAME';
Val : REAL;
...
Res := PsPlcInfoHwmonGetFloatVal(ValName, Val);
```

➢ PsPlcInfoHwmonGetIntVal (HwmonGetIntVal до 1.7.2.0)

Запрос параметра с типом UDINT.

Входной аргумент:

– имя параметра *ValName* типа STRING;

– ссылка на возвращаемый параметр *Val* типа REFERENCE TO UDINT.

Возвращаемое значение:

– TRUE, если запрос успешен, *outVal* содержит значение запрашиваемого параметра;

– FALSE, если произошла ошибка, *outVal* не изменен.

Пример:

```
Res : BOOL;
ValName : STRING := 'VAR_NAME';
Val : UDINT;
...
Res := PsPlcInfoHwmonGetIntVal(ValName, Val);
```

➢ **PsPlcInfoHwmonGetParamNamesList (HwmonGetParamNamesList до 1.7.2.0)**

Получение списка параметров.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– указатель на указатель на список параметров типа POINTER TO POINTER TO STRING.

Пример (получение значений всех параметров):

```
resB      : BOOL;
ptrParams : POINTER TO POINTER TO STRING;
strVal    : STRING;

ptrParams := PsPlcInfoHwmonGetParamNamesList();
WHILE ptrParams <> 0 AND ptrParams^ <> 0 DO
resB := PsPlcInfoHwmonGetStringVal(ptrParams^^, strVal);
      ptrParams := ptrParams + SIZEOF(POINTER TO STRING);
END_WHILE
```

➢ **PsPlcInfoHwmonGetStringVal (HwmonGetStringVal до 1.7.2.0)**

Запрос параметра с типом STRING.

Входные аргументы:

– имя параметра *ValName* типа STRING;

– ссылка на возвращаемый параметр *Val* типа REFERENCE TO STRING.

Возвращаемое значение:

– TRUE, если запрос успешен, *outVal* содержит значение запрашиваемого параметра;

– FALSE, если произошла ошибка, *outVal* не изменен.

Пример:

```
Res : BOOL;
ValName : STRING := 'VAR_NAME';
Val : STRING;
...
Res := RPsPlcInfoHwmonGetStringVal(ValName, Val);
```

➢ **PsPlcInfoHwmonSetWatchdog (HwmonSetWatchdog до 1.7.2.0)**

Настройка сторожевого таймера.

Входной аргумент:

– период срабатывания (в секундах) *wdt_value* типа UINT.

Функция предназначена как для первоначального запуска сторожевого таймера, так и для его сброса. Для отключения сторожевого таймера необходимо вызвать с параметром *wdt_value* равным 0.

Возвращаемое значение:

– TRUE, если сторожевой таймер успешно запущен;

– FALSE, если произошла ошибка, таймер не запущен.

Пример:

```
Res : BOOL;
StopWD : BOOL := FALSE;
...
Res := PsPlcInfoHwmonSetWatchdog(20);

IF StopWD THEN
      StopWD := FALSE;
      Res := PsPlcInfoHwmonSetWatchdog(0);
END_IF
```

➢ **PsPlcInfoMD5ToString**

Преобразовать MD5 в шестнадцатеричную запись.

Входной аргумент:

– значение *value* типа MD5.

Возвращаемое значение:

– шестнадцатеричная запись MD5 *PsPlcInfoMD5ToString* типа STRING(32).

➢ **PsPlcInfoSHA256ToString**

Преобразовать SHA256 в шестнадцатеричную запись.

Входной аргумент:

– значение *value* типа SHA256.

Возвращаемое значение:

– шестнадцатеричная запись SHA256 *PsPlcInfoSHA256ToString* типа STRING.

➢**SetLemtWatchdog (устаревшая)**

Настройка сторожевого таймера.

Входной аргумент:

– период срабатывания (в секундах) *timeout* типа UINT.

Функция предназначена как для первоначального запуска сторожевого таймера, так и для его сброса. Для отключения сторожевого таймера необходимо вызвать с параметром *timeout* равным 0.

Пример:

```
Res : RTS_IEC_RESULT;
StopWD : BOOL := FALSE;
...
Res := REGUL_PLC_INFO.SetLemtWatchdog(20);

IF StopWD THEN
      StopWD := FALSE;
      Res := REGUL_PLC_INFO.SetLemtWatchdog(0);
END_IF
```

➢**SysComExtGetRxTx (устаревшая)**

Запрос количества принятых/переданных байт COM-портом.

Входные аргументы:

– дескриптор COM-порта *Hcom*, полученный с помощью функции SysComOpen;

– счетчик байтов, переданных портом *Tx* типа ULINT;

– счетчик байтов, принятых портом *Rx* типа ULINT.

Пример:

```
ComRes : RTS_IEC_RESULT;
Tx : ULINT;
Rx : ULINT;
Hcom : RTS_IEC_HANDLE := SysCom.SysComOpen( ... );
...
ComRes := REGUL_PLC_INFO.SysComExtGetRxTx(Hcom,Tx, Rx);
```

Функциональные блоки

➢FirmwareSHA256

Функциональный блок для вычисления хэш-суммы (контрольной суммы) СПО по алгоритму SHA-256

Входной аргумент:

– состояние запуска вычисления хэш-суммы *xExecute* типа BOOL:

  o FALSE - сбросить FB в начальное состояние, остановить ранее запущенный процесс вычисления хэш,

  o TRUE - запустить вычисление хэш-суммы, получить состояние вычисления. После завершения вычисления xDone = TRUE, udiResult - результат вычисления;

Выходные аргументы:

- статус успешного завершения вычисления хэш-суммы *xDone* типа BOOL:
  - FALSE - вычисление не завершено успешно,
  - TRUE - вычисление завершено успешно;
- статус ошибки вычисления *xError* типа BOOL:
  - FALSE - нет ошибки
  - TRUE - вычисление завершено с ошибкой;
- результат вычисления хэш-суммы udiResult:
  - ERR_OK - нет ошибки,
  - ERR_CALL_AGAIN - Если xError=FALSE, то идет процесс вычисления и требуется повторный запуск FirmwareSHA256() для получения хэш-суммы. Если xError=TRUE, то не смог запустить вычисление, т.к. вычисление запущено другим ФБ,
  - ERR_FAILED - внутренняя ошибка посылки команды,
  - ERR_CRC_FAILED - вычисление хэш-суммы было прервано;
- значение хэш-суммы *HashSha256* типа SHA256.

Пример:

```
PROGRAM PlcInfoExample
VAR
    xGetSHA256: BOOL;
    hash: FirmwareSHA256;
END_VAR

IF NOT xGetSHA256 THEN
    // получить значение SHA256
    hash( xExecute := TRUE );
    IF hash.xDone THEN
        // вычисление завершено успешно
        hash.HashSha256; // содержит хэш-сумму системных файлов

        //очистить хэш
        hash( xExecute := FALSE );
        xGetSHA256 := TRUE;
    ELSIF hash.xError THEN
        // вычисление завершено с ошибкой
        hash.udiResult; // см. код ошибки в описании

        //очистить хэш
        hash( xExecute := FALSE );
        xGetSHA256 := TRUE;
    END_IF
END_IF
```

➢FirmwareHash

Функциональный блок для вычисления хэш-суммы (контрольной суммы) СПО по алгоритму MD5

Входной аргумент:

- состояние запуска вычисления хэш-суммы *xExecute* типа BOOL:
  - FALSE - сбросить FB в начальное состояние, остановить ранее запущенный процесс вычисления хэш,
  - TRUE - запустить вычисление хэш-суммы, получить состояние вычисления. После завершения вычисления xDone = TRUE, udiResult - результат вычисления;

Выходные аргументы:

- статус успешного завершения вычисления хэш-суммы *xDone* типа BOOL:
  - FALSE - вычисление не завершено успешно,
  - TRUE - вычисление завершено успешно;
- статус ошибки вычисления *xError* типа BOOL:
  - FALSE - нет ошибки
  - TRUE - вычисление завершено с ошибкой;
- результат вычисления хэш-суммы udiResult:
  - ERR_OK - нет ошибки,
  - ERR_CALL_AGAIN - Если xError=FALSE, то идет процесс вычисления и требуется повторный запуск FirmwareSHA256() для получения хэш-суммы. Если xError=TRUE, то не смог запустить вычисление, т.к. вычисление запущено другим ФБ,
  - ERR_FAILED - внутренняя ошибка посылки команды,
  - ERR_CRC_FAILED - вычисление хэш-суммы было прервано;
- значение хэш-суммы *Hash* типа MD5.

Пример:

```
PROGRAM PlcInfoExample
VAR
    xGetHash: BOOL;
    hash: FirmwareHash;
END_VAR

IF NOT xGetHash THEN
    // получить значение Hash
    hash( xExecute := TRUE );
    IF hash.xDone THEN
        // вычисление завершено успешно
        hash.Hash; // содержит хэш-сумму системных файлов

        //очистить хэш
        hash( xExecute := FALSE );
        xGetHash := TRUE;
    ELSIF hash.xError THEN
        // вычисление завершено с ошибкой
        hash.udiResult; // см. код ошибки в описании

        //очистить хэш
        hash( xExecute := FALSE );
```

```
        xGetHash := TRUE;
    END_IF
END_IF
```

PsPrint (PS_Print)

Реализует программную поддержку печати.

Интерфейсы

IHtmlPrinter

Интерфейс функционального блока принтера, печатающего формат html.

Пример:

```
res := Printer.Print();
```

Метод Print

Печать документа на принтере.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "0", если ошибка отсутствует;
– "1", если наименование принтера некорректное (INVALID);
– "2", если некорректный (INVALID) файл;
– "3", если произошла ошибка службы печати;
– "4", если произошла ошибка файловой системы;
– "5", если произошла внутренняя ошибка.

Пример:

```
Printer.Print();
```

Свойство Landscape

Ориентация страницы (альбомная или портретная/книжная).

Принимает одно из следующих значений:

– “FALSE” – книжная;

– “TRUE” – альбомная.

Пример:

```
Printer.Landscape := FALSE;
```

Свойство PageMargins

Размер полей в миллиметрах, в формате “l,t,r,b”:

– l – левое поле;

– t – верхнее поле;

– r – правое поле;

– b – нижнее поле.

Допускается указание не всех полей. Например, “25,15”:

– левое поле – 25 мм;

– верхнее поле – 15 мм.

Пример:

```
Printer.PageMargins := '20,20,20,20';
```

Свойство PageSize

Размер страницы (формат).

Принимает одно из следующих значений: A4, B5, Letter, Executive, A0, A1, A2, A3, A5, A6, A7, A8, A9, B0, B1, B2, B3, B4, B6, B7, B8, B9, B10, C5E, Comm10E, DLE, Folio, Ladger, Tabloid.

При указании неизвестного формата или пустой строки, используется значение по умолчанию (A4).

Пример:

```
Printer.PageSize := 'A4';
```

➢**IPostscriptPrinter**

Интерфейс функционального блока принтера, печатающего формат postscript.

Пример:

```
res := Printer.Print('printerName');
```

Метод Print

Печать документа на принтере.

Входной аргумент:

– уникальный идентификатор принтера из файла plc.cfg *printerName* типа STRING.

Возвращаемое значение:

– "0", если ошибка отсутствует;
– "1", если наименование принтера некорректное (INVALID);
– "2", если некорректный (INVALID) файл;
– "3", если произошла ошибка службы печати;
– "4", если произошла ошибка файловой системы;
– "5", если произошла внутренняя ошибка.

Пример:

```
Printer.Print('printername');
```

▸ Структуры

➢TPageFormat

Структура с информацией о формате страницы.

Содержит переменные, указанные в таблице Л.16.

Таблица Л.16 – Переменные TPageFormat

Переменная	Тип	Описание
PageWidthPt	UINT	Ширина
PageHeightPt	UINT	Высота

➢TPageMargins

Структура с информацией о размере полей в миллиметрах.

Содержит переменные, указанные в таблице Л.17

Таблица Л.17 – Переменные TPageMargins

Переменная	Тип	Описание
MarginTopMm	UINT	Отступ сверху, мм (по умолчанию 25 мм)
MarginLeftMm	UINT	Отступ слева, мм (по умолчанию 25 мм)
MarginRightMm	UINT	Отступ справа, мм (по умолчанию 25 мм)
MarginBottomMm	UINT	Отступ снизу, мм (по умолчанию 25 мм)

▸ Функции

➢PsPrintHtml (PrintHtml до 1.7.2.0)

Отправляет настройки и Html-данные на принтер.

Входные аргументы:

- имя принтера *printerName* типа STRING;
- указатель на строку *pHtmlData* типа POINTER TO STRING;
- семейство шрифтов *FontFamily* типа STRING;
- размер шрифта *FontSize* типа UINT;
- размер страницы *PageSize* типа STRING;
- поля *Margins* типа STRING;
- ориентация (альбомная/книжная) *Landscape* типа BOOL;
- кодировка *Encoding* типа STRING.

Возвращаемое значение:

- "0", если ошибка отсутствует;
- "1", если наименование принтера некорректное (INVALID);
- "2", если некорректный (INVALID) файл;
- "3", если произошла ошибка службы печати;
- "4", если произошла ошибка файловой системы;
- "5", если произошла внутренняя ошибка.

Пример:

```
Print := PsPrintHtml( mPrinterName, mBuffer.GetStringPtr(), mFontFamily,
mFontSize, mPageSize, mPageMargins, mLandscape, mEncoding );
```

➢PsPrintPostscript (PrintPostscript до 1.7.2.0)

Отправляет необработанные данные PostScript на принтер.

Входные аргументы:

- имя принтера *printerName* типа STRING;
- указатель на строку *pPostscriptData* типа POINTER TO STRING.

Возвращаемое значение:

- "0", если ошибка отсутствует;
- "1", если наименование принтера некорректное (INVALID);
- "2", если некорректный (INVALID) файл;
- "3", если произошла ошибка службы печати;
- "4", если произошла ошибка файловой системы;
- "5", если произошла внутренняя ошибка.

Пример:

```
PsPrintPostscript(printerName, GetPostscript());
```

➢PsPrintPurgePrintQueue (PurgePrintingQueue до 1.7.2.0)

Очищает очередь печати.

Входной аргумент:

– имя принтера *printerName* типа STRING.

Возвращаемое значение:

– "0", если ошибка отсутствует;
– "1", если наименование принтера некорректное (INVALID);
– "2", если некорректный (INVALID) файл;
– "3", если произошла ошибка службы печати;
– "4", если произошла ошибка файловой системы;
– "5", если произошла внутренняя ошибка.

Пример:

```
PsPrintPurgePrintQueue('printerName');
```

▸ Функциональные блоки

➢AsciiPrinter

AsciiPrinter обеспечивает печать документов с использованием символов из набора ASCII моноширинным шрифтом (символы одинаковой ширины). Для печати документа его содержимое построчно добавляется методом AddTextLine. При добавлении текста, он автоматически конвертируется на язык PostScript и сохраняется в строковом буфере функционального блока. Для печати документа используется метод Print.

Пример:

```
FUNCTION Test_Print : BOOL
VAR_INPUT
Printer     : PsPrint.AsciiPrinter;
END_VAR
VAR
xInit : BOOL;
res         : INT;
Print : BOOL;
Clear   : BOOL;
END_VAR
...
IF NOT xInit THEN
Printer.AddTextLine('Hello world!');
res := Printer.Print('printerName');
Printer.Clear();
xInit := TRUE;
```

```
END_IF
```

Метод AddTextLine

Добавление текстовой строки.

Входной аргумент:

– добавляемая строка *line* типа STRING(255).

Пример:

```
Printer.AddTextLine('Hello world!');
```

Метод Clear

Очистка содержимого буфера.

Входной аргумент:

– отсутствует.

Пример:

```
Printer.Clear();
```

Метод GetPostscript

Получение документа, описанного на языке PostScript.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– указатель на строку типа POINTER TO STRING.

Пример:

```
PrintPostscript(printerName, GetPostscript());
```

Метод NewPage

Начать новую страницу.

Входной аргумент:

– отсутствует.

Пример:

```
NewPage();
```

Метод Print

Печать документа на принтере.

Входной аргумент:

– уникальный идентификатор принтера из файла plc.cfg *printerName* типа STRING.

Возвращаемое значение:

– "0", если ошибка отсутствует;

– "1", если наименование принтера некорректное (INVALID);

– "2", если некорректный (INVALID) файл;

– "3", если произошла ошибка службы печати;

– "4", если произошла ошибка файловой системы;

– "5", если произошла внутренняя ошибка.

Пример:

```
Printer.Print('printername');
```

➢ HTMLPrinter

Предоставляет возможность печати html-документов.

Пример:

```
FUNCTION Test_Main : BOOL
VAR_INPUT
Printer     : PsPrint.HtmlPrinter;
END_VAR
VAR
xInit : BOOL;
res         : INT;
Print : BOOL;
Clear   : BOOL;
END_VAR
...
IF NOT xInit THEN
Printer( PrinterName := 'printerName' );
Printer.Encoding := 'windows-1251';
Printer.AddText('<html>');
Printer.AddText('<head><meta charset="windows-1251"></head>');
Printer.AddText('<body>');
Printer.AddText('<h1>Заголовок1</h1>');
Printer.AddText('<h2>Заголовок2</h2>');
Printer.AddText('<h3>Заголовок3</h3>');
Printer.AddText('<h3>Русский алфавит</h3>');
Printer.AddText('<p>абвгдеёжзийклмнопрстуфхцчшщъыьэюя</p>');
Printer.AddText('<p>АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ</p>');
Printer.AddText('<h3>Греческий алфавит</h3>');
Printer.AddText('<p>&alpha;&beta;&gamma;&delta;&epsilon;&zeta;&eta;&theta;&iota;&k
appa;&lambda');
Printer.AddText(';&mu&nu;&xi;&omicron;&pi;&rho;&sigma;&sigmaf;&tau;&upsilon;&phi;&
chi;&psi;');
Printer.AddText('&omega;<sub>p</sub></p><p>&Alpha;&Beta;&Gamma;&Delta;&Epsilon;&Ze
ta;&Eta;');
Printer.AddText('&Theta;&Iota;&Kappa;&Lambda;&Mu;&Nu;&Xi;&Omicron;&Pi;&Rho;&Sigma;
&Tau;');
Printer.AddText('&Upsilon;&Phi;&Chi;&Psi;&Omega;</p>');
Printer.AddText('<p><sub>нижний индекс</sub><sup>верхний индекс</sup></p>');
```

```
Printer.AddText('</body>');
Printer.AddText('</html>');
res := Printer.Print();
Printer.Clear();
xInit := TRUE;
END_IF
```

Метод AddText

Добавление текста документа в буфер.

Входной аргумент:

– добавляемый текст *text* типа STRING.

Пример:

```
Printer.AddText('<h1>Заголовок1</h1>');
```

Метод Clear

Очистка содержимого буфера.

Входной аргумент:

– отсутствует.

Пример:

```
Printer.Clear();
```

Метод Print

Печать документа на принтере.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– "0", если ошибка отсутствует;
– "1", если наименование принтера некорректное (INVALID);
– "2", если некорректный (INVALID) файл;
– "3", если произошла ошибка службы печати;
– "4", если произошла ошибка файловой системы;
– "5", если произошла внутренняя ошибка.

Пример:

```
Printer.Print();
```

Свойство Encoding

Кодировка символов входной строки, преобразование текстовых данных из одного формата в другой.

Поддерживаются следующие значения кодировки: IBM 850, IBM 866, IBM 874, ISO 8859-1 до 10, ISO 8859-13 до 16, KOI8-R, KOI8-U, Windows-1250 до 1258.

По умолчанию: Windows-1251.

Пример:

```
Printer.Encoding := 'windows-1251';
```

Свойство FontFamily

Семейство шрифтов для оформления текста.

Доступные шрифты: DejaVuSans-Bold, DejaVuSans-BoldOblique, DejaVuSans-Oblique, DejaVuSans, DejaVuSansMono-Bold, DejaVuSansMono-BoldOblique, DejaVuSansMono-Oblique, DejaVuSansMono, DejaVuSerif-Bold, DejaVuSerif-BoldOblique, DejaVuSerif-Oblique, DejaVuSerif, Vera, VeraBI, VeraBd, VeraIt, VeraMoBI, VeraMoBd, VeraMoIt, VeraMono, VeraSe, VeraSeBd.

По умолчанию: DejaVuSans.

Пример:

```
Printer.FontFamily := 'Vera';
```

Свойство FontSize

Размер шрифта. Задается абсолютный размер.

По умолчанию: 10.

Пример:

```
Printer.FontSize := 10;
```

Свойство Landscape

Ориентация страницы (альбомная или портретная/книжная).

Принимает одно из следующих значений:

– “FALSE” – книжная;

– “TRUE” – альбомная.

Пример:

```
Printer.Landscape := FALSE;
```

Свойство PageMargins

Размер полей в миллиметрах, в формате “l,t,r,b”:

– l – левое поле;
– t – верхнее поле;
– r – правое поле;
– b – нижнее поле.

Допускается указание не всех полей. Например - “25,15”:

– левое поле – 25 мм;
– верхнее поле – 15 мм.

По умолчанию: 20, 20, 20, 20.

Пример:

```
Printer.PageMargins := '20,20,20,20';
```

Свойство PageSize

Размер страницы (формат).

Принимает одно из следующих значений: A4, B5, Letter, Executive, A0, A1, A2, A3, A5, A6, A7, A8, A9, B0, B1, B2, B3, B4, B6, B7, B8, B9, B10, C5E, Comm10E, DLE, Folio, Ladger, Tabloid.

При указании неизвестного формата или пустой строки, используется значение по умолчанию (A4).

Пример:

```
Printer.PageSize := 'A4';
```

PsQueue

Компонент для работы с очередью.

Функциональные блоки

➢ArrayQueue

Функциональный блок для контейнера, реализующего функционал очереди.

Входные аргументы:

– указатель на массив для хранения pArrayData типа POINTER TO BYTE;
– размер массива udiArraySize типа UDINT.

Пример:

```
queue : PsQueue.ArrayQueue;
```

```
storage : ARRAY[0..100] OF BYTE;
queue.init(ADR(storage), 100);
```

Метод Clear

Удаление всех элементов из очереди.

Входной аргумент:

– отсутствует.

Пример:

```
queue : PsQueue.ArrayQueue;
queue.Clear();
```

Метод Dequeue

Удаляет объект из начала очереди и возвращает его.

Входные аргументы:

– указатель для размещения объекта pdata типа POINTER TO BYTE;

– размер объекта max_size_data типа UDINT.

Возвращаемое значение:

– количество записанных по указателю байт типа UDINT.

Пример:

```
storage : ARRAY[0..100] OF BYTE;
queue : PsQueue.ArrayQueue;
obj1 : BYTE := 1;
obj2 : BYTE := 2;
test_obj : BYTE;
////
queue.init(ADR(storage), 100);
arr.Enqueue(ADR(obj1), SIZEOF(BYTE));
arr.Enqueue(ADR(obj2), SIZEOF(BYTE));
arr.Dequeue(ADR(test_obj), SIZEOF(BYTE)); // test_obj = 1
arr.Dequeue(ADR(obj), SIZEOF(BYTE)); // test_obj = 2
```

Метод Enqueue

Добавление объекта в конец очереди.

Входные аргументы:

– указатель на объект pdata типа POINTER TO BYTE;

– размер объекта size типа UDINT.

Возвращаемое значение:

– результат добавления типа BOOL.

Пример:

```
storage : ARRAY[0..100] OF BYTE;
queue : PsQueue.ArrayQueue;
obj1 : BYTE := 1;
obj2 : BYTE := 2;
test_obj : BYTE;
////
queue.init(ADR(storage), 100);
arr.Enqueue(ADR(obj1), SIZEOF(BYTE));
arr.Enqueue(ADR(obj2), SIZEOF(BYTE));
arr.Dequeue(ADR(test_obj), SIZEOF(BYTE)); // test_obj = 1
arr.Dequeue(ADR(obj), SIZEOF(BYTE)); // test_obj = 2
```

Метод Init

Инициализация очереди.

Метод IsEmpty

Проверка очереди на отсутствие объектов.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– флаг отсутствия символов типа BOOL.

Пример:

```
queue : PsQueue.ArrayQueue;
empty_flag : BOOL := TRUE;
// ...
empty_flag := queue.isEmpty();
```

Свойство Count

Количество элементов очереди.

С помощью **Get** можно получить количество элементов очереди.

Пример:

```
queue : PsQueue.ArrayQueue;
count : UDINT;
// ...
count := queue.Count;
```

➢**ArrayQueueTs**

Функциональный блок для контейнера, реализующего функционал потокобезопасной очереди.

Входные аргументы:

– указатель на массив для хранения pArrayData типа POINTER TO BYTE;

– размер массива udiArraySize типа UDINT.

Пример:

```
queue : PsQueue.ArrayQueueTs;
storage : ARRAY[0..100] OF BYTE;
queue.init(ADR(storage), 100);
```

Метод Clear

Удаление всех элементов из очереди.

Входной аргумент:

– отсутствует.

Пример:

```
queue : PsQueue.ArrayQueueTs;
queue.Clear();
```

Метод Dequeue

Удаляет объект из начала очереди и возвращает его.

Входные аргументы:

– указатель для размещения объекта pdata типа POINTER TO BYTE;

– размер объекта max_size_data типа UDINT.

Возвращаемое значение:

– количество записанных по указателю байт типа UDINT.

Пример:

```
storage : ARRAY[0..100] OF BYTE;
queue : PsQueue.ArrayQueueTs;
obj1 : BYTE := 1;
obj2 : BYTE := 2;
test_obj : BYTE;
////
queue.init(ADR(storage), 100);
arr.Enqueue(ADR(obj1), SIZEOF(BYTE));
arr.Enqueue(ADR(obj2), SIZEOF(BYTE));
arr.Dequeue(ADR(test_obj), SIZEOF(BYTE)); // test_obj = 1
arr.Dequeue(ADR(obj), SIZEOF(BYTE)); // test_obj = 2
```

Метод Enqueue

Добавление объекта в конец очереди.

Входные аргументы:

– указатель на объект pdata типа POINTER TO BYTE;

– размер объекта size типа UDINT.

Возвращаемое значение:

– результат добавления типа BOOL.

Пример:

```
storage : ARRAY[0..100] OF BYTE;
queue : PsQueue.ArrayQueueTs;
obj1 : BYTE := 1;
obj2 : BYTE := 2;
test_obj : BYTE;
////
queue.init(ADR(storage), 100);
arr.Enqueue(ADR(obj1), SIZEOF(BYTE));
arr.Enqueue(ADR(obj2), SIZEOF(BYTE));
arr.Dequeue(ADR(test_obj), SIZEOF(BYTE)); // test_obj = 1
arr.Dequeue(ADR(obj), SIZEOF(BYTE)); // test_obj = 2
```

Метод Init

Инициализация очереди.

PsRedundancy (PS_Redundancy)

Компонент резервирования ЦП.

▸ Перечисления

➢RedMode

Перечисление RedMode типа SINT содержит режимы работы резервирования, указанные в таблице Л.18.

Таблица Л.18 – Режимы работы резервирования

Режим	Значение	Описание режима
CONN_ERROR	-3	Ошибка соединения
CR_ERROR	-2	Критическая ошибка
ERROR	-1	Ошибка
DISABLED	0	Выключено
BOOTUP	1	Инициализация
SYNC	2	Синхронизация
STANDBY	3	Ведомый
ACTIVE_STANDALONE	4	Автономный
ACTIVE	5	Активный

▸ Структуры

➢TAppInfo

Структура с информацией о приложении.

Содержит переменные, указанные в таблице Л.19.

Таблица Л.19 – Переменные TAppInfo

Переменная	Тип	Описание
ProjInfo	PROJECT_INFO	Информация о проекте
sApplicationName	STRING	Имя приложения
sProfile	STRING	Профиль
dtLastChanges	DWORD	Последнее изменение
sCompilerVersion	STRING	Версия компилятора
dataGuid	TGuid	Уникальный идентификатор IEC данных
codeGuid	TGuid	Уникальный идентификатор IEC кода

➢TGuid

Структура для уникального идентификатора.

Содержит переменные, указанные в таблице Л.20

Таблица Л.20 – Переменные TGuid

Переменная	Тип
data1	DWORD
data2	WORD
data3	WORD
data4	ARRAY [0..7] OF BYTE

➢TStat2

Структура с диагностической информацией.

Содержит переменные, указанные в таблице Л.21.

Таблица Л.21 – Переменные TStat2

Переменная	Тип	Описание
TotalDataSize	UDINT	Общий объем резервируемых данных
xPassiveCpu	BOOL	Роль ведомого ПЛК
xCpuA	BOOL	Признак идентификации ЦП как CPU_A
SyncOkCount	ULINT	Счетчик успешной синхронизации
SyncErrCount	ULINT	Счетчик ошибок синхронизации

Переменная	Тип	Описание
SyncOk	BOOL	Задача синхронизирована с ведущим контроллером
TaskStat	TTaskStat	Статистика по задаче
DoSyncTimeUs	UDINT	Время выполнения синхронизации
SendVarTimeUs	UDINT	Время отправки данных
xConnection1	BOOL	Наличие связи по каналу 1
xConnection2	BOOL	Наличие связи по каналу 2

➢**TTaskStat**

Структура со статистикой по задаче резервирования.

Содержит переменные, указанные в таблице Л.22.

Таблица Л.22 – Переменные TTaskStat

Переменная	Тип	Описание
CallCount	ULINT	Счетчик вызовов задачи
CallIntervalUs	UDINT	Время с момента последнего вызова (мкс)
ConfigIntervalDefaultUs	UDINT	Интервал задачи по умолчанию (мкс)
ConfigIntervalCurrentUs	UDINT	Текущий интервал выполнения задачи (мкс)
OffsetUs	DINT	Текущее смещение задачи (мкс)
JitterUs	DINT	Джиттер задачи (мкс)
StepCorrectionCnt	ULINT	Счетчик жестких коррекций
SoftCorrectionCnt	ULINT	Счетчик мягких коррекций

Функции

➢ **PsRddRtGetAppInfo (GetAppInfo до 1.7.2.0)**

Запрос информации о приложении.

Входной аргумент:

– информация о приложении *appInfo* типа REFERENCE TO TAppInfo.

Возвращаемое значение:

– результат запроса *PsRddRtGetAppInfo* типа RTS_IEC_RESULT.

Пример:

```
info : PsRedundancy.TAppInfo;
```

```
res : RTS_IEC_RESULT;
// ...
res := PsRddRtGetAppInfo(info);
```

➢ **PsRddRtGetConnection (GetConnection до 1.7.2.0)**

Запрос состояния каналов связи.

Входные аргументы:

– состояние первого канала *xConnection1* типа REFERENCE TO BOOL;

– состояние второго канала *xConnection2* типа REFERENCE TO BOOL.

Возвращаемое значение:

– результат запроса *PsRddRtGetConnection* типа RTS_IEC_RESULT.

Пример:

```
ch1 : BOOL;
ch2 : BOOL;
res : RTS_IEC_RESULT;
// ...
res := PsRddRtGetConnection(ch1, ch2);
```

➢ **PsRddRtGetMode (GetMode до 1.7.2.0)**

Запрос режима работы резервирования.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– режим работы резервирования *PsRddRtGetMode* типа RedMode.

Пример:

```
mode : PsRedundancy.RedMode;
// ...
mode := PsRddRtGetMode();
```

➢ **PsRddRtGetStats2 (GetStats2 до 1.7.2.0)**

Запрос на получение диагностической информации.

Входной аргумент:

– диагностическая информация *stats* типа REFERENCE TO TStat2.

Возвращаемое значение:

– результат запроса *PsRddRtGetStats2* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy.TStat2;
res : RTS_IEC_RESULT;
// ...
```

```
res := PsRddRtGetStats2(stats);
```

➢ **PsRddRtIsCpuA (IsCpuA до 1.7.2.0)**

Запрос признака CPU_A.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– TRUE, если данный CPU является CPU_A;

– FALSE, если данный CPU не является CPU_A.

Пример:

```
IF PsRddRtIsCpuA() THEN
// Данный ПЛК является CPU A.
END_IF
```

➢ **PsRddRtSwitchToStandby (SwitchToStandby до 1.7.2.0)**

Запрос на передачу управления ПЛК-партнеру.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат запроса *PsRddRtSwitchToStandby* типа RTS_IEC_RESULT.

Пример:

```
IF PsRddRtSwitchToStandby() <> CmpErrors.Errors.ERR_OK THEN
// Запрос отклонен
END_IF
```

➢ **PsRddRtSynchronize (Synchronize до 1.7.2.0)**

Функция для организации синхронизации задач и данных на двух ПЛК.

Функция должна вызываться перед алгоритмической частью задачи.

Возвращаемое значение:

– TRUE, если синхронизация выполнена в данном цикле;

– FALSE, если синхронизация не выполнена в данном цикле.

Пример:

```
IF PsRddRtSynchronize(FALSE) THEN
// Синхронизация выполнена в данном цикле
...
ELSE
// Синхронизация не выполнена в данном цикле
...
```

```
END_IF
```

➢Synchronize4

Функция для организации синхронизации задач и данных на двух ПЛК с получением диагностической информации.

Функция должна вызываться перед алгоритмической частью задачи.

Входные аргументы:

– диагностическая информация stats типа REFERENCE TO TStat2.

Возвращаемое значение:

– TRUE, если синхронизация выполнена в данном цикле;

– FALSE, если синхронизация не выполнена в данном цикле.

Пример:

```
stats : PsRedundancy.TStat2;
// ...
IF PsRedundancy.Synchronize4(FALSE, stats) THEN
// Синхронизация выполнена в данном цикле
...
ELSE
// Синхронизация не выполнена в данном цикле
...
END_IF
```

▸ Функциональные блоки

➢CrossMemory

Функциональный блок для обмена данными между модулями ЦП.

Пример:

```
LocalVariable : BOOL;
RemoteVariable : BOOL;
CrossData : PsRedundancy.CrossMemory(SIZEOF(LocalVariable), ADR(LocalVariable),
ADR(RemoteVariable));
...
PsRedundancy.Synchronize(FALSE);
// После вызова Synchronize данные в RemoteVariable синхронизованы
IF LocalVariable <> RemoteVariable
THEN
      ...
END_IF
```

Метод fb_init

Инициализация функционального блока CrossMemory.

Входной аргумент:

– размер переменной *udiSize* типа UDINT;

– указатель на переменную *pVariable* типа POINTER TO BYTE;

– указатель на переменную - обратная связь *pFeedbackVariable* типа POINTER TO BYTE.

Пример:

```
LocalVariable : DINT;
RemoteVariable : DINT;
CrossData : PsRedundancy.CrossMemory(SIZEOF(LocalVariable), ADR(LocalVariable),
ADR(RemoteVariable));
```

➢**SharedMemory**

Функциональный блок разделяемой памяти.

При синхронизации данные копируются от ведущего модуля ЦП к ведомому, заменяя предыдущие данные в ведомом ЦП.

Пример:

```
haredVariable : BOOL;
SharedData : PsRedundancy.SharedMemory(SIZEOF(SharedVariable),
ADR(SharedVariable));
...
PsRedundancy.Synchronize(FALSE);
// После вызова Synchronize данные в SharedVariable синхронизованы
IF SharedVariable
THEN
      ...
END_IF
```

Метод fb_init

Инициализация функционального блока SharedMemory.

Входной аргумент:

– размер переменной *udiSize* типа UDINT;

– указатель на переменную *pVariable* типа POINTER TO BYTE.

Пример:

```
SharedVariable : BOOL;
SharedData : PsRedundancy.SharedMemory(SIZEOF(SharedVariable),
ADR(SharedVariable));
```

PsRedundancy_OS (PS_Redundancy_OS)

Компонент резервирования ЦП. Описание **RedMode, TAppInfo, TGuid** смотри выше.

Структуры

➢**TTaskIds**

Структура с описанием идентификаторов задач, задействованных в резервировании.

Содержит переменные, указанные в таблице Л.23.

Таблица Л.23 – Переменные TTaskIds

Переменная	Тип	Описание
ids	ARRAY [0..(255 - 1)] OF INT	Массив идентификаторов задач (-1 - invalid id)

➢TRedundancyStat

Структура с диагностической информацией.

Содержит переменные, указанные в таблице Л.24.

Таблица Л.24 – Переменные TRedundancyStat

Переменная	Тип	Описание
TotalDataSize	UDINT	Общий объем резервируемых данных
AsyncTasksCount	UDINT	Количество асинхронных задач
xPassiveCpu	BOOL	Роль ведомого ПЛК
xCpuA	BOOL	Признак CPU_A. Для CPU_A всегда TRUE
xConnection1	BOOL	Наличие связи канал 1
xConnection2	BOOL	Наличие связи канал 2

➢TSyncTaskStat

Структура со статистикой по синхронной задаче.

Содержит переменные, указанные в таблице Л.25.

Таблица Л.25 – Переменные TSyncTaskStat

Переменная	Тип	Описание
TaskStat	TTaskStat	Статистика по задаче
SyncStat	TSyncStat	Дополнительная статистика по синхронной задаче

➢TTaskStat

Структура со статистикой по задаче.

Содержит переменные, указанные в таблице Л.26.

Таблица Л.26 – Переменные TTaskStat

Переменная	Тип	Описание
TaskName	STRING	Имя задачи
TaskDataSize	UDINT	Объем резервируемых данных в задаче

Переменная	Тип	Описание
CallCount	ULINT	Счетчик вызовов задачи
CallIntervalUs	UDINT	Время с момента последнего вызова (мкс)
MaxCallIntervalUs	UDINT	Максимальное значение CallIntervalUs
ConfigIntervalDefaultUs	UDINT	Интервал задачи по умолчанию (мкс)
JitterUs	DINT	Джиттер задачи (мкс)
MaxJitterUs	DINT	Максимальное значение JitterUs
DoSyncTimeUs	UDINT	Время выполнения синхронизации данных
MaxDoSyncTimeUs	UDINT	Максимальное значение DoSyncTimeUs
SendVarTimeUs	UDINT	Время отправки данных
MaxSendVarTimeUs	UDINT	Максимальное значение SendVarTimeUs
IsSyncTask	BOOL	true - синхронная задача, false - асинхронная

➢**TSyncStat**

Структура с дополнительной статистикой по синхронной задаче.

Содержит переменные, указанные в таблице Л.27.

Таблица Л.27 – Переменные TSyncStat

Переменная	Тип	Описание
ConfigIntervalCurrentUs	UDINT	Текущий интервал выполнения задачи (мкс)
OffsetUs	DINT	Текущий смещение задачи (мкс)
StepCorrectionCnt	ULINT	Счетчик жестких коррекций
SoftCorrectionCnt	ULINT	Счетчик мягких коррекций
SyncOkCount	ULINT	Счетчик успешной синхронизации
SyncErrCount	ULINT	Счетчик ошибок синхронизации
SyncOk	BOOL	Задача синхронизирована с ведущим контроллером

Функции

➢ **PsRddGetRddStat (GetRedundancyStat до 1.7.2.0)**

Запрос на получение общей диагностической информации. Перед вызовом обновите статистику, выполнив UpdateStats().

Входной аргумент:

– диагностическая информация по синхронной задаче *stats* типа REFERENCE TO TRedundancyStat.

Возвращаемое значение:

– результат запроса *PsRddGetRddStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TRedundancyStat;
res : RTS_IEC_RESULT;
// ...
res := PsRddGetRddStat(stats);
```

➢ **PsRddGetSyncTaskStat (GetSyncTaskStat до 1.7.2.0)**

Запрос на получение диагностической информации по синхронной задаче. Перед вызовом обновите статистику, выполнив UpdateStats().

Входной аргумент:

– диагностическая информация по синхронной задаче *stats* типа REFERENCE TO TSyncTaskStat.

Возвращаемое значение:

– результат запроса *PsRddGetSyncTaskStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TSyncTaskStat;
res : RTS_IEC_RESULT;
// ...
res := PsRddGetSyncTaskStat(stats);
```

➢**PsRddGetTaskStat (GetTaskStat до 1.7.2.0)**

Запрос на получение диагностической информации по задаче. Перед вызовом обновите статистику, выполнив UpdateStats().

Входные аргументы:

– идентификатор задачи *task_id* типа BYTE;

– диагностическая информация по задаче *stats* типа REFERENCE TO TTaskStat.

Возвращаемое значение:

– результат запроса *PsRddGetTaskStat* типа RTS_IEC_RESULT.

Пример:

```
stats : PsRedundancy_OS.TTaskStat;
task_id: BYTE := 0;
res : RTS_IEC_RESULT;
// ...
```

```
res := PsRddGetTaskStat(task_id, stats);
```

➢ **PsRddUpdateStats (UpdateStats до 1.7.2.0)**

Запрос на обновление статистики по всем задачам.

Для получения актуальной информации, при использовании функций GetRedundancyStat(), GetSyncTaskStat(), GetTaskStat(), GetConnection() необходимо, перед их вызовом, обновить статистику, выполнив UpdateStats().

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат запроса *PsRddUpdateStats* типа RTS_IEC_RESULT.

Пример:

```
res : RTS_IEC_RESULT;
// ...
res := PsRddUpdateStats();
```

➢ **GetAppInfo**

Запрос информации о приложении.

Входной аргумент:

– информация о приложении *appInfo* типа REFERENCE TO TAppInfo.

Возвращаемое значение:

– результат запроса *GetAppInfo* типа RTS_IEC_RESULT.

Пример:

```
info : PsRedundancy_OS.TAppInfo;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetAppInfo(info);
```

➢ **GetConnection**

Запрос состояния каналов связи. Перед вызовом, обновите статистику, выполнив UpdateStats().

Входные аргументы:

– состояние первого канала *xConnection1* типа REFERENCE TO BOOL;

– состояние второго канала *xConnection2* типа REFERENCE TO BOOL.

Возвращаемое значение:

– результат запроса *GetConnection* типа RTS_IEC_RESULT.

Пример:

```
ch1 : BOOL;
ch2 : BOOL;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.GetConnection(ch1, ch2);
```

➢ **GetMode**

Запрос режима работы резервирования.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– режим работы резервирования *GetMode* типа RedMode.

Пример:

```
mode : PsRedundancy_OS.RedMode;
// ...
mode := PsRedundancy_OS.GetMode();
```

➢**PsRddGetTaskList (GetTaskList до 1.7.2.0)**

Запрос идентификаторов задач, задействованных в резервировании.

Входной аргумент:

– идентификаторы задач *task_ids* типа REFERENCE TO TTaskIds.

Возвращаемое значение:

– результат запроса *PsRddGetTaskList* типа RTS_IEC_RESULT.

Пример:

```
task_ids : PsRedundancy_OS.TTaskIds;
res : RTS_IEC_RESULT;
// ...
res := PsRddGetTaskList(task_ids);
```

➢ **IsCpuA**

Запрос признака CPU_A.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– TRUE, если данный CPU является CPU_A;

– FALSE, если данный CPU не является CPU_A.

Пример:

```
IF PsRedundancy_OS.IsCpuA() THEN
    // Данный ПЛК является CPU A.
END_IF
```

➢ **SwitchToStandby**

Запрос на передачу управления ПЛК-партнеру.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат запроса *SwitchToStandby* типа RTS_IEC_RESULT.

Пример:

```
IF PsRedundancy_OS.SwitchToStandby() <> CmpErrors.Errors.ERR_OK THEN
    // Запрос отклонен
END_IF
```

➢ **IsApplicationEqual**

Запрос признака одинаковых приложений на CPU_A и CPU_B.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– результат запроса *PsRddIsApplicationEqual* типа BOOL:
  - TRUE, если приложения на CPU_A и CPU_B одинаковые;
  - FALSE, если приложения на CPU_A и CPU_B разные.

Пример:

```
IF PsRddIsApplicationEqual() <> CmpErrors.Errors.ERR_OK THEN
    // Приложения на CPU_A и CPU_B одинаковы
END_IF
```

➢ **setDenyBeActive (устаревшая)**

Запрос на установку/снятия флага на запрет быть ведущим.

Входной аргумент:

– флаг запрета быть ведущим *DenyBeActive* типа BOOL.

Возвращаемое значение:

– результат запроса *setDenyBeActive* типа RTS_IEC_RESULT.

Пример:

```
xDenyBeActive : BOOL := TRUE;
res : RTS_IEC_RESULT;
// ...
res := PsRedundancy_OS.setDenyBeActive(xDenyBeActive);
```

PsSysFile

Компонент для работы с файлами.

Функции

➢PsSysFileStartCopyScript

Запуск сценария копирования данных.

Функция работает синхронно. Копирование большого файла может привести к длительной блокировке исполнения задачи. Следует использовать только в задачах с низким приоритетом.

Входной аргумент:

– имя сценария из конфигурационного файла *copyjobs.cfg* *script* типа STRING(255).

Пример:

```
res : RTS_IEC_RESULT;
// ...
res := PsSysFileStartCopyScript('Script');
```

➢PsSysFileTouch

Запрос на обновление даты модификации файла или директории.

Входной аргумент:

– путь к файлу или директории *filePath* типа STRING(255).

Пример:

```
res : RTS_IEC_RESULT;
// ...
res := PsSysFile.PsSysFileTouch('file');
res := PsSysFile.PsSysFileTouch('directory');
res := PsSysFile.PsSysFileTouch('directory/file');
```

Функциональные блоки

➢PsSysFileAsyncStartCopyScript

Функциональный блок для асинхронного копирования данных.

Входной аргумент:

– флаг управления работой *xExecute* типа BOOL.

Выходные аргументы:

– статус успеха *xDone* типа BOOL;

– статус неудачи *xError* типа BOOL;

– код возврата *returnCode* типа DINT.

Пример:

```
test(xExecute := TRUE); // Запустить сценарий
// ...
IF test.xError or test.xDone THEN
      // Сценарий завершился
      returnCode := test.returnCode;
ELSE
      // Сценарий еще не завершился
      test(xExecute := FALSE); // Принудительно остановить сценарий
END_IF
```

Метод FB_init

Инициализация функционального блока PsSysFileAsyncStartCopyScript.

Входной аргумент:

– имя сценария из конфигурационного файла *copyjobs.cfg* *script* типа STRING.

Пример:

```
test : PsSysFileAsyncStartCopyScript('Script');
```

PsTime (PS_Time)

Компонент для работы со временем и NTP.

▸ Перечисления

➢connection_t

Перечисление connection_t типа INT содержит статусы соединения, указанные в таблице Л.28.

Таблица Л.28 – Статусы соединения

Статус	Значение	Описание статуса
connected_invalid	-1	Ошибка получения статуса
connected_false	0	Соединение не активно
connected_true	1	Соединение активно

➢TPeerStatus

Перечисление TPeerStatus типа UINT содержит статусы источника времени, указанные в таблице Л.29.

Таблица Л.29 – Статусы соединения

Статус	Значение	Символ	Описание статуса
sel_reject	0	' '	Исключен как неработающий сервер
sel_falsetick	1	'x'	Исключен как ненадежный источник по алгоритму пересечения
sel_excess	2	'.'	Исключен из-за переизбытка кандидатов (не используется)
sel_outlyer	3	'-'	Исключен из списка кандидатов блоком алгоритмов
sel_candidate	4	'+'	Включен алгоритмом в список кандидатов
sel_backup	5	'#'	Резервный (Число кандидатов больше, чем необходимо для организации отбора)
sel_sys_peer	6	'*'	Сервер выбран для синхронизации
sel_pps_peer	7	'o'	Выбран для синхронизации, используется PPS

Структуры

➢peer_description_t

Структура с описанием параметров источника времени.

Содержит переменные, указанные в таблице Л.30.

Таблица Л.30 – Переменные peer_description_t

Переменная	Тип	Описание
assid	UINT	Идентификатор соединения
status	TPeerStatus	Статус источника времени
remote	STRING[31]	Адрес источника времени
refid	STRING[31]	Идентификатор источника
stratum	UDINT	Номер часового слоя
when	UDINT	Время с последнего принятого пакета
poll	UDINT	Интервал опроса
reach	UDINT	Значение reach регистра (в восьмеричной системе счисления)
delay	REAL	Задержка
offset	REAL	Смещение

Переменная	Тип	Описание
jitter	REAL	Джиттер
connection	connection_t	Статус соединения

➢**time_channels_data_t**

Структура с описанием статистики NTP сервера.

Содержит переменные, указанные в таблице Л.31

Таблица Л.31 – Переменные time_channels_data_t

Переменная	Тип	Описание
is_ntpd_reachable	UDINT	Доступность NTPD
sync	UDINT	Признак наличия сигнала точного времени
tm	timespec_t	Время обновления данных
peer_count	UDINT	Количество источников синхронизации
peers	ARRAY [0..7] OF peers_description_t	Описание источников времени
offset	REAL	Отклонение системы от источника
frequency	REAL	Ошибка частоты
rootdisp	REAL	Разброс показаний сервера
stratum	UDINT	Часовой слой
leap	UDINT	Корректировка секунд
satellites	UDINT	Число видимых спутников
has_sys_peer	UDINT	Наличие источника синхронизации

➢**timespec_t**

Структура с данными для обозначения времени.

Содержит переменные, указанные в таблице Л.32.

Таблица Л.32 – Переменные timespec_t

Переменная	Тип	Описание
tv_sec	DINT	Количество секунд
tv_nsec	DINT	Количество наносекунд

Функции

➢ **PsTimeGetUtcTimeNs (GetUtcTimeNs до 1.7.2.0)**

Получение текущего времени.

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– метку времени высокого разрешения типа ULINT.

Пример:

```
time_ns : ULINT := PsTimeGetUtcTimeNs();
```

➢ **PsTimeNtpGetStats (ntp_get_stats до 1.7.2.0)**

Получение статистики NTP сервера.

Входной аргумент:

– структура *stats* типа time_channels_data_t с описанием статистики NTP сервера.

Пример:

```
stats : PsTime.time_channels_data_t;
PsTimeNtpGetStats(stats);
```

➢**UTCTIMENS_TO_STRING**

Преобразование метки времени из ULINT в STRING.

Входные аргументы:

– метка времени высокого разрешения *ts* типа ULINT;

– флаг добавления временной зоны *xAddTimeZone* типа BOOL;

– флаг отображения микросекунд *xUsec* типа BOOL.

Возвращаемое значение:

– метку времени типа STRING.

Пример:

В примере представлен результат выполнения для часового пояса UTC+05:00.

```
ts : ULINT := PsTimeGetUtcTimeNs();
ts_str_tz_us : STRING := PsTime.UTCTIMENS_TO_STRING(ts, TRUE, TRUE);
// '24-06-2021 15:02:07.883430'
ts_str_tz : STRING := PsTime.UTCTIMENS_TO_STRING(ts, TRUE, FALSE);
// '24-06-2021 15:02:07.883'
ts_str_us : STRING := PsTime.UTCTIMENS_TO_STRING(ts, FALSE, TRUE);
// '24-06-2021 10:02:07.883430'
```

➢ **PsTimeWriteCurTimeToRtc (WriteCurrentTimeToRtc до 1.7.2.0)**

Запись текущего времени в аппаратные часы.

Входной аргумент:

– отсутствует.

Пример:

```
PsTimeWriteCurTimeToRtc()
```

PsUnittest (PS_Unittest)

Компонент для тестирования проектов.

Функции

➢SendTestResult

Добавить в лог результат теста.

Добавляет запись о результате теста в журнал "TestLogger". При положительном прохождении теста тип записи – *сообщение*, при отрицательном – *ошибка*.

Входные аргументы:

– имя теста *testName* типа STRING(70);

– результат теста *res* типа BOOL. Если результат теста "TRUE", то тест прошел успешно.

Возвращаемое значение:

– наличие ошибки выставления результата типа BOOL. Если функция возвращает значение "FALSE", то ошибка отсутствует.

Пример:

```
SendTestResult('Test1', TRUE); // Запись "Test1: [passed]"
SendTestResult('Test2', FALSE); // Запись "Test2: [FAILED]"
```

➢UnittestEnd

Добавить в лог метку конца юнит-тестов.

Добавляет запись "APP ENDED" в журнал "TestLogger".

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– наличие ошибки выставления результата типа BOOL. Если функция возвращает значение "FALSE", то ошибка отсутствует.

Пример:

```
UnittestStart();
// Запуск различных юнит-тестов...
```

```
UnittestEnd();
```

➢**UnittestStart**

Добавить в лог метку начала юнит-тестов.

Добавляет запись "APP STARTED" в журнал "TestLogger".

Входной аргумент:

– отсутствует.

Возвращаемое значение:

– наличие ошибки выставления результата типа BOOL. Если функция возвращает значение "FALSE", то ошибка отсутствует.

Пример:

```
UnittestStart();
// Запуск различных юнит-тестов...
UnittestEnd();
```

PsUtils (PS_Utils)

Компонент для работы со строками.

▸ Функции

➢ **PsUtilsUtf8ToWString (UTF8_TO_WSTRING_1 до 1.7.2.0)**

Конвертация STRING (UTF-8) к WSTRING (UTF-16) с ограничением размера буфера.

При указании размера буфера следует добавлять NULL-символ (2 байта).

Входные аргументы:

– указатель на STRING *srcStr* типа POINTER TO STRING;

– указатель на WSTRING *destWStr* типа POINTER TO WSTRING;

– размер выходного буфера в байтах *destSize* типа UDINT.

Возвращаемое значение:

– количество записанных байт типа DINT.

Пример:

```
STR : STRING := 'TEST_TEST';
W : WSTRING(9);
PsUtilsUtf8ToWString(ADR(STR), ADR(W), 10); // WSTR будет содержать "TEST"
PsUtilsUtf8ToWString(ADR(STR), ADR(W), 20); // WSTR будет содержать "TEST_TEST"
```

➢ **PsUtilsWStringToUtf8 (WSTRING_TO_UTF8_1 до 1.7.2.0)**

Конвертация WSTRING (UTF-16) к STRING (UTF-8) с ограничением размера буфера.

При указании размера буфера следует добавлять NULL-символ (1 байт).

Входные аргументы:

– указатель на WSTRING *srcWStr* типа POINTER TO WSTRING;

– указатель на STRING *destStr* типа POINTER TO STRING;

– размер буфера в байтах *destSize* типа UDINT.

Возвращаемое значение:

– количество записанных байт типа DINT.

Пример:

```
WSTR : WSTRING := "TEST_TEST";
STR : STRING(9);
PsUtilsWStringToUtf8(ADR(WSTR), ADR(STR), 5); // STR будет содержать 'TEST'
PsUtilsWStringToUtf8(WSTR), ADR(STR), 10); // STR будет содержать 'TEST_TEST'
```

▸ Функциональные блоки

➢ PsUtilsTStringBuffer (TStringBuffer до 1.7.2.0)

Функциональный блок для работы с буфером строк.

Блок использует динамическую память.

Пример:

```
string_buffer : PsUtilsTStringBuffer();
```

Метод Append

Добавление строки в конец буфера.

Входной аргумент:

– строка *str*, которая будет добавляться в конец буфера, типа STRING(255).

Пример:

```
buf : PsUtils.TStringBuffer;
buf.Append('TEST'); // Буфер теперь содержит 'TEST'
```

Метод AppendPtr

Добавление строки в конец буфера.

Входной аргумент:

– указатель на строку *ptrStr* типа POINTER TO STRING.

Пример:

```
buf : PsUtils.TStringBuffer;
str : STRING := '_123';
buf.Append('TEST'); // Содержит 'TEST'
```

```
buf.AppendPtr(ADR(str)); // Содержит 'TEST_123'
```

Метод AppendW

Преобразование WSTRING(UTF-16) к STRING(UTF-8) и добавление строки в буфер.

Входной аргумент:

– указатель на строку в UTF-16 *wstrPtr* типа POINTER TO WSTRING.

Пример:

```
buf : PsUtils.TStringBuffer;
str : WSTRING := "_123";
buf.Append('TEST'); // Содержит 'TEST'
buf.AppendW(ADR(str)); // Содержит 'TEST_123'
```

Метод Clear

Очищение содержимого буфера.

Пример:

```
buf : PsUtils.TStringBuffer;
//...
buf.Append('TEST'); // Буфер теперь содержит 'TEST'
buf.Clear(); // Буфер пустой
```

Метод Destroy

Освобождение внутренних ресурсов буфера, выделенные динамически.

После вызова метода функциональный блок будет невалиден.

Пример:

```
string_buffer : PsUtils.TStringBuffer();
// ...
string_buffer.Destroy();
```

Метод GetStringPtr

Получение указателя на содержимое буфера.

После вызова других методов функционального блока указатель становится невалидным.

Возвращаемое значение:

– указатель на содержимое буфера типа POINTER TO STRING.

Пример:

```
string_buffer : PsUtils.TStringBuffer();
string_ptr : POINTER TO STRING;
// ...
string_ptr := string_buffer.GetStringPtr();
```

Метод InsertPtr

Вставка заданного количества символов в определенную позицию в буфере.

Если *insPos* > **Size(),** тогда *insPos* := **Size(),**

если *insPos* < 0, тогда *insPos* := **Size()** + *insPos* + 1,

где *insPos* – позиция внутри буфера,

Size() – результат **TStringBuffer.Size(),** размер буфера.

Входные аргументы:

– позиция внутри буфера *insPos* типа DINT;

– указатель на вставляемую строку *str* типа POINTER TO STRING;

– количество символов для вставки *strLen* типа UDINT.

Пример:

```
buf : PsUtils.TStringBuffer;
str : STRING := 'TEST';
buf.Append('>>><<<'); // Содержит '>>><<<'
buf.InsertPtr(3, ADR(str), 4); // Содержит '>>>TEST<<<'
```

Метод InsertW

Преобразование WSTRING(UTF-16) к STRING(UTF-8) и вставка заданного количества символов в определенную позицию в буфере.

Если *insPos* > **Size(),** тогда *insPos* := **Size(),**

если *insPos* < 0, тогда *insPos* := **Size()** + *insPos* + 1,

где i*nsPos* – позиция внутри буфера,

Size() – результат **TStringBuffer.Size(),** размер буфера.

Входные аргументы:

– позиция внутри буфера *insPos* типа DINT;

– указатель на строку для преобразования и вставки *str* типа POINTER TO WSTRING;

– количество символов для вставки *strLen* типа UDINT.

Пример:

```
buf : PsUtils.TStringBuffer;
str : WSTRING := "TEST";
buf.Append('>>><<<'); // Содержит '>>><<<'
buf.InsertW(3, ADR(str), 4); // Содержит '>>>TEST<<<'
```

Метод Resize

Изменение количества сохраненных символов.

Входной аргумент:

– количество символов, которые нужно сохранить, *newSize* типа UDINT.

Пример:

```
buf : PsUtils.TStringBuffer;
buf.Append('TEST_TEST');
buf.Resize(4); // Буфер теперь содержит 'TEST'
```

Метод Size

Получение размера буфера.

Возвращаемое значение:

– размер буфера в байтах типа UDINT.

Пример:

```
buf : PsUtils.TStringBuffer;
size : UDINT := 0;
// ...
buf.Append('TEST_TEST');
size := buf.Size(); // size := 9
```

Свойство Capacity

Размер, выделенный для хранения.

Используется для резервирования места, для хранения в целях оптимизации и получения текущего размера выделенной области.

С помощью **Get** можно получить текущий размер, выделенный под хранение.

С помощью **Set** можно установить размер, выделенный под хранение.

Пример:

```
buf : PsUtils.TStringBuffer;
capacity : UDINT := 0;
// ...
IF buf.Capacity < 200 THEN
buf.Capacity := 200;
END_IF
capacity := buf.Capacity; // capacity будет больше или равен 200 байт
```

PsIoDrvRegulBus_OS

Компонент для работы с шиной RegulBus_OS

Свойство RingIsBroken

Позволяет получить статус наличия обрыва кольца

Пример:

```
VAR
temp1 : BOOL;
END_VAR

...

temp1 := Regul_Bus_OS. RingIsBroken;
```

Свойство RingBrokenPlace

Указывает на место обрыва кольца в случае, если обрыв был обнаружен

Пример:

```
VAR
temp1 : STRING;
END_VAR

...

temp1 := Regul_Bus_OS.RingBrokenPlace;
```