Утвержден
ЛЯЮИ.00707-01 33 01-ЛУ

СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

«ИНТЕГРИРОВАННАЯ СРЕДА РАЗРАБОТКИ ПРОГРАММ НА ЯЗЫКАХ СТАНДАРТА МЭК 61131-3 ELPLC-LOGIC»

Руководство программиста

ЛЯЮИ.00707-01 33 01

Листов 119

Инв.№ подл.	Подпись и дата	Взам.инв.№	Инв.№ дубл.	Подпись и дата

Перв. примен. Литера

2023 г.

АННОТАЦИЯ

Данный документ содержит сведения о порядке работы с программным обеспечением «Интегрированная среда разработки программ на языках стандарта МЭК 61131-3 ELPLC-LOGIC» (далее ПО ELPLC-LOGIC). Руководство программиста содержит сведения о назначении, условиях выполнения, элементах пользовательского интерфейса, порядке разработки прикладных программ, работе с внешними модулями (далее плагинами).

Порядок установки, удаления и запуска ELPLC-LOGIC праведен в документе «Руководство системного программиста» (ЛЯЮИ.00707-01 32 01).

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ

ПО ELPLC-LOGIC предназначено для создания, отладки и передачи на целевое устройство (Программируемых логических контроллеров) программ стандарта МЭК 61131-3.

В качестве описания алгоритмов и логики работы прикладных программ могут выступать текстовые (Structured Text (далее ST) и Instruction List (далее IL)) и графические языки (Function Block Diagram (далее FBD), Ladder Diagram (далее LD), Sequential Function Chart (далее SFC)) программирования.

Основными функция ELPLC-LOGIC:

- разработка алгоритмов на языках ST, IL, FBD, LD, SFC;
- настройка сборки (компиляции) разработанных алгоритмов;
- сборка прикладных программ для передачи на целевые устройства (ПЛК) на архитектурах arm, e2k (Эльбрус) и amd64;
- настройка режимов работы ПЛК (настройка «retain» переменных и режимов резервирования);
- настройка внешних модулей (плагинов);
- отладка разработанных программ при помощи встроенных инструментов отладки.

ПО ELPLC-LOGIC может выполняться на операционных системах (далее ОС) семейства Windows (не ниже Windows 10) и Linux (с версией ядра не ниже 3.X).

Для выполнения ПО ELPLC-LOGIC требуется следующие программные продукты:

- ОС Windows или на основе ядра Linux (Astra Linux special edition или common edition, Базальт СПО или Эльбрус ОС);
- подсистема Windows для Linux (WSL) (только при выполнении на ОС Windows);
- компилятор на основе архитектуры E2K (Эльбрус) (только для сборки (компиляции) алгоритмов для ПЛК на основе ЦП Эльбрус), версии не ниже lcc:1.23.17;
- компилятор (GNU arm gcc version 6.3.0) на основе архитектуры ARM (только для сборки (компиляции) алгоритмов для ПЛК на основе ЦП на основе ядра ARM), версии не ниже gcc 6.3.0;

- компилятор (GNU gcc version 6.3.0) на основе архитектуры ADM64 (только для сборки (компиляции) алгоритмов для ПЛК на основе ЦП н на основе ядра ADM64), версии не ниже gcc 6.3.0;
- транслятор (ElplcCompiler) программного кода МЭК 61131-3 в язык Си;
- набор библиотек Qt5 из состава ОС, версии не ниже Qt5.9;
- программный продукт make из состава ОС, утилита, автоматизирующая процесс преобразования файлов из одной формы в другую. Чаще всего это компиляция исходного кода в объектные файлы и последующая компоновка в исполняемые файлы или библиотеки;
- ПО «Среда исполнения программ на языках стандарта МЭК 61131-3 ELPLC-RUNTIME» (ЛЯЮИ.00708-01), версии не ниже 3.0.0 (ELPLC-RUNTIME под необходимую архитектуру целевого устройства (ПЛК)).

1.1. Требование к процессорному модулю целевого устройства (ПЛК).

Процессорный модуль должен выполнять следующие требования:

- центральный процессор на архитектуре arm, e2k или amd64;
- частота центрального процессора не ниже 1000 МГц;
- количество ядер центрального процессора не менее 1;
- объем оперативной памяти не ниже 4 Гб;
- объем встроенного диска ssd и/или hdd не менее 32 Гб;
- количество Ethernet портов не менее 2 (без режима резервирования), не менее 3 (при использовании резервирования);
- количество USB портов версии не ниже 2.0 не менее 3;
- количество RS-485 портов не менее 2;
- количество RS-232 портов не менее 1;
- наличие индикации работоспособности процессорного модуля;
- аппаратный ключ для изменения режимов выполнения программы.

2. ХАРАКТЕРИСТИКИ ПРОГРАММЫ

ПО ELPLC-LOGIC позволяет работать в конфигурационном режиме и в режиме исполнения прикладной программы. В конфигурационном режиме происходит создание прикладной программы, написание алгоритмов и логики её основных программных модулей и их связывание с внешними модулями УСО (устройство связи с объектом). В режиме исполнения прикладная программа передаётся на целевое устройство и может быть запущена с режимом отладки и без отладки.

Контроль правильности выполнения осуществляется средствами среды:

- редакторами языков МЭК 61131-3;
- компиляторами языков МЭК 61131-3;
- компилятором для целевой платформы (ПЛК).

Соответствующие предупреждения и ошибки выводятся либо в виде сообщения в диалоге, либо в виде текстовой информации в отладочную консоль.

- временные характеристики;
- режим работы;
- средства контроля правильности выполнения;
- самовосстанавливаемость программы.

3. ОБРАЩЕНИЕ К ПРОГРАММЕ. ОПИСАНИЕ ПРОЦЕДУР ВЫЗОВА ПРОГРАММЫ. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Общая схема по созданию прикладной программы в ПО ELPLC-LOGIC представлена на рисунке (см. Рисунок 1). Входными данными являются программные модули, написанные пользователем (в большинстве случаев инженером по автоматизации) на текстовых (ST, IL) и/или графических (FBD, SFC, LD) языках в соответствии со стандартом МЭК 61131-3, объединённые в проект. Каждый такой проект хранится в базе данных (SQL подобная СУБД).

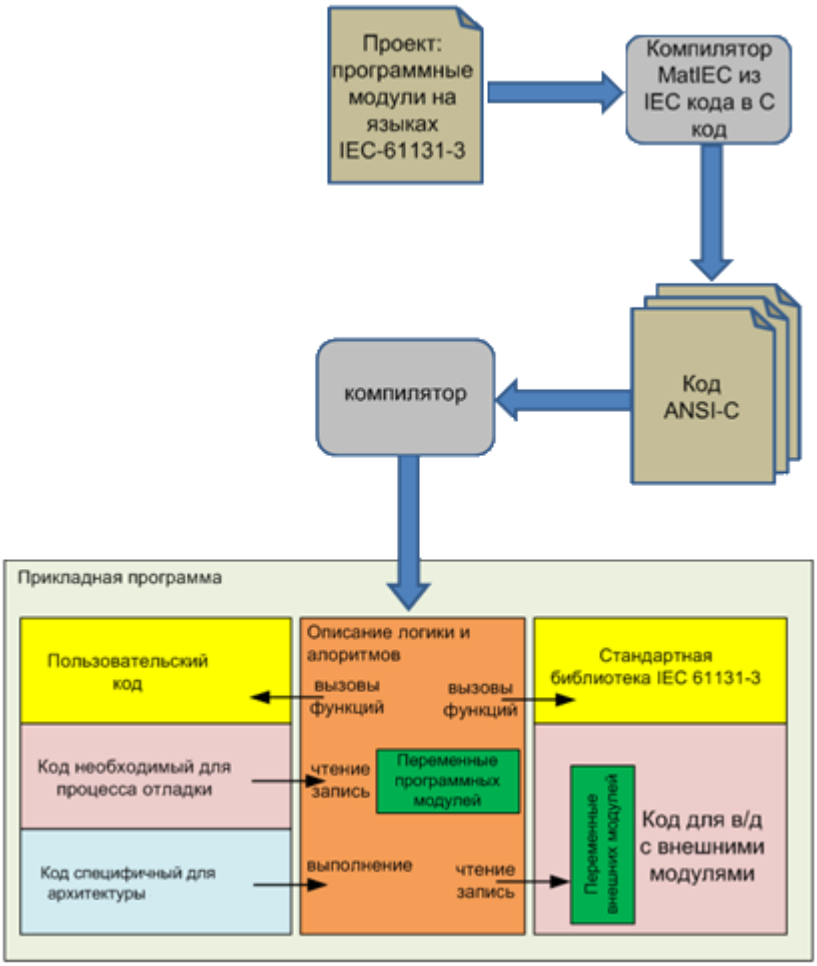


Рисунок 1- Общая схема по созданию прикладной программы

Входными данными является алгоритмы, разработанные в ПО ELPLC-LOGIC на языках МЭК 61131-3 с использованием внешних модулей. Разработанные алгоритмы сохраняются в базе данных.

Выходными данными является сгенерированный исходный код на языке Си и библиотека для передачи ее исполнения в целевое устройство (ПЛК).

Перечень входных файлов:

- файл <название проекта>.sln – база данных, сработанного в ПО ELPLC-LOGIC алгоритма;

Перечень выходных файлов:

- файл generated_plc.st, содержащий промежуточный код на языке ST, сгенерированный для всех программных модулей и ресурсов, транслируемый в язык Си;
- файлы: elplclogic.h, config.c, config.h, POUS.h, POUS.c, plc_debugger.c, plc_main.c, plc_retain.c, plc_variables.c, resource1.c и файлы, соответствующие программам – содержат код (на языке Си) реализации алгоритмов и логики работы программных модулей и ресурсов проекта;
- файлы, содержащие код на языке Си для взаимодействия с внешними модулями;
- исполняемый файл в виде динамической библиотеки (с расширением so), компилируемый из этих вышеперечисленных С файлов.

Сгенерированный код на языке Си, с помощью кросс-компилятора или компилятора, запущенного под UNIX-подобной оболочной или подсистемы Windows для Linux (WSL), компилируется в исполняемый бинарный файл, представленный в виде библиотеки.

Исполняемый файл, благодаря средствам ПО ELPLC-LOGIC, может быть передан на целевое устройство через локальную сеть.

На целевом устройстве исполняемый файл запускается и в процессе работы выполняет следующие действия (см. рис. 1):

- с помощью драйвера обменивается данными с внешними модулями;
- исполняет алгоритмы и логику, определенную пользователем в программных модулях проекта;
- предоставляет данные для трансляции в системы верхнего уровня;
- сохраняет и транслирует информацию для отладки прикладных программ.

4. СООБЩЕНИЯ

Тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действия, которые необходимо предпринять по этим сообщениям представлены в таблице (см. Таблица 1 – Сообщения выдаваемые ПО ELPLC-LOGIC).

Таблица 1 – Сообщения выдаваемые ПО ELPLC-LOGIC

№, строки	Текст сообщения на английском языке	Текст сообщения на русском языке	Описание сообщения
1	Variable 'avgload:%ID0.100' in 'plc0:program0' does not found in any plugin.	Переменная 'avgload:%ID0.100' в 'plc0:program0' не найдена ни в одном плагине.	В одном или нескольких плагинах были удалены переменные, добавленные ранее в POU
2	Build cancelled	Сборка отменена	По какой-то причине процесс сборки был завершен пользователем
3	Program analysis cancelled.	Статический анализ кода отменен.	По какой-то причине процесс статического анализа кода был завершен пользователем
4	Cleaning the build directory	Очистка директории сборки	От пользователя получена команда удаления собранных файлов проекта
5	You need to build the project first	Необходимо сначала собрать проект	Попытка передать проект без предварительной сборки
6	Can't open rdc.so!	Ошибка открытия rdc.so!	Файл проекта поврежден
7	Can't read rdc.so. Please build the PLC file again.	Ошибка чтения rdc.so. Пересоберите проект снова.	Файл проекта создан, но ошибка в чтении
8	PLC0 Saving project - in progress	PLC0 Сохранение проекта - выполняется	Запрошена процедура сохранения проекта
9	PLC0 Saving project completed	PLC0 Сохранение проекта завершено	Процедура выше выполнена успешно
10	Error: Maximum number of elements reached for components of type 'cpu_mp17'	Ошибка: Достигнуто максимальное количество данного типа 'cpu_mp17'!	Попытка добавить большее количество модулей данного типа, чем возможно в проекте
11	PLC0: Target already has this retain	PLC0: На ПЛК уже передан retain-	Попытка отправить файл еще раз

	file	файл	
12	PLC0: Connecting with PLC [192.168.30.108:3000]	PLC0: Подключение к ПЛК[192.168.30.108:3000]	Запрос подключения к ПЛК
13	PLC0: Error connecting to PLC: The PLC did not respond to the connection request	PLC0: Ошибка подключение к ПЛК: От ПЛК не получен ответ на запрос	При подключении к ПЛК возникла ошибка в транспортной или прикладной части
14	PLC0: Connected with PLC	PLC0: Подключение к ПЛК установлено	Подключение к ПЛК успешно установлено на запрос выше
15	PLC0: Disconnected with PLC	PLC0: Отключено от ПЛК	Запрос отключения от ПЛК
16	PLC0: Target already has this program	PLC0: На ПЛК уже передана данная программа	Попытка отправить файл еще раз
17	PLC0: Can't get target status	PLC0: Ошибка получения статуса ПЛК	При подключении статус ПЛК отличается от разрешенных
18	PLC0: Target is broken	PLC0: Ошибка в работе ПЛК.	ПЛК находится в состоянии BROKEN (ошибка сборки, ошибка инициализации, подключения итд)
19	PLC0: Target is empty	PLC0: Подключение к ПЛК[192.168.30.108:3000]	–
20	PLC0: Target started	PLC0: ПЛК запущен	–
21	PLC0: Target stoped	PLC0: ПЛК остановлен	–
22	PLC0: Error connecting to PLC: Error command: START_CONTROL	PLC0: Ошибка подключение к ПЛК: Ошибка выполения команды: START_CONTROL	При подключении к ПЛК возникла ошибка в отправке команды на подключении клиента
23	PLC0: Log enable error	PLC0: Ошибка включения системы логирования	При включении системы логирования произошла ошибка
24	PLC0: Library successfully transfered	PLC0: Библиотека успешно	Собранная программа успешна передана

	to PLC	передана на ПЛК	на контроллер
25	PLC0: Can't transfer the PLC: rdc.so	PLC0: Ошибка передачи на ПЛК: rdc.so	Ошибка передачи программы выше
26	PLC0: Can't start the PLC: Network error	PLC0: Ошибка запуска на ПЛК: Проблема с сетевым подключением.	Ошибка в запуске программе на контроллере
27	PLC0: Unable to stop the PLC: Network error.	PLC0: Ошибка остановки ПЛК: Проблема с сетевым подключением.	Ошибка в остановке программе на контроллере
28	PLC0: Can not get bin file size from plc	PLC0: Ошибка получения размера файла с ПЛК	При попытке получить размер запрашиваемого файла произошла ошибка
29	PLC0: Receive file from PLC, file name: retain.bin, file size: 136	PLC0: Получен файл с ПЛК, имя файла: retain.bin, размер 136	Успешно получен бинарный файл с контроллера
30	PLC0: Can not get bin file from plc	PLC0: Ошибка получения бинарного файла с ПЛК	При попытке получить бинарный файл произошла ошибка
31	PLC0: Send file to PLC, file name: retain.bin	PLC0: Отправка файла на ПЛК. Имя: retain.bin	Запрошена процедура отправки файла на контроллер
32	PLC0: Can not get trace variables, plc is not runing	PLC0: Ошибка получения отслеживаемых переменных, ПЛК не запущен	–
33	Archiver A1: Incorrect variable location: program0.Var2	–	–
34	Archiver A1: Incorrect trigger variable location	–	–
35	Analysis failed: Please build the project first	Ошибка анализа кода: Пожалуйста соберите сначала проект.	Попытка запуска статического анализа когда без сборки проекта
36	Generating SoftPLC IEC-61131 ST/IL/SFC code...	Генерация МЭК-61131 ST/IL/SFC кода ПЛК...	Генерация С-кода для ПЛК

37	Can't generate SoftPLC code:	Ошибка генерации SoftPLC кода:	Ошибка создания кода выше
38	Running static code analysis in /home/proj/ ...	Запуск статического анализатора в /home/proj/ ..	–
39	Analysis failed: Can't read /home/proj/plc.st. Please build the project first	Ошибка статического анализа: Чтение /home/proj/plc.st. Пожалуйста соберите сначала проект.	–
40	Static Analysis warnings:	Ошибки системы статического анализа:	–
41	Static Analysis: No errors found	Система статического анализа: Ошибок не найдено.	–
42	Build failed	Ошибка сборки	–
43	Solution file is corrupted!	Файл проекта поврежден!	
44	Starting build in /home/proj/..	Сборка запущена в /home/proj...	Локальная сборка проекта
45	Build failed: Internal error: Can't patch the header files	Ошибка сборки: Обновление заголовочных файлов	–
46	Build failed: C code was not generated properly	Ошибка сборки: С код не был сгенерирован.	–
47	C code generated successfully.	С-код успешно сгенерирован.	–
48	Running remote compilation on 192.168.30.108...	Запуск удаленной сборки на 192.168.30.108…	Попытка подключения и удаленной сборки сгенерированных файлов проекта
49	Compiling:	Компиляция:	
50	Build completed: /home/proj/rdc.so (MD5: 889d26e8a584e749ac73fc39c17078d3)	Сборка завершена: /home/proj/rdc.so (MD5: 889d26e8a584e749ac73fc39c17078d3)	Успешная компиляция и получения библиотеки на клиентское устройство
51	Successfully built.	Сборка прошла успешно.	–

52	Build failed: Can't create build directory: /home/proj/build_plc0. Please make sure that you have sufficient permissions	Ошибка сборки: Создание директории: /home/proj/build_plc0. Проверьте разрешения на запись в папку.	Ошибка с правами на доступ к целевой папке сборки. Возможно данная папка открыта в проводнике или в других инструментах
53	Build failed: ElplcCompiler path [] is not valid	Ошибка сборки: Некорректный путь к ElplcCompiler []	При попытке генерации С-кода средствами ElplcCompiler, не найден путь до исполняемого файла
54	cpu_mp17: incorrect module slot on ELPLC bus for ELPLC-bus [0_1:MDO.17_0] (same as [0_0:MDI32.17_0])	cpu_mp17: Некорректный номер слота модуля на шине ELPLC-bus [0_1:MDO.17_0] (совпадает с [0_0:MDI32.17_0])	Добавлены более одного модуля на один и тот же слот шины ELPLC-bus

5. ОСНОВНОЕ ОКНО ПРОГРАММЫ

Основное окно ПО представлено на рисунке (см. Рисунок 2).

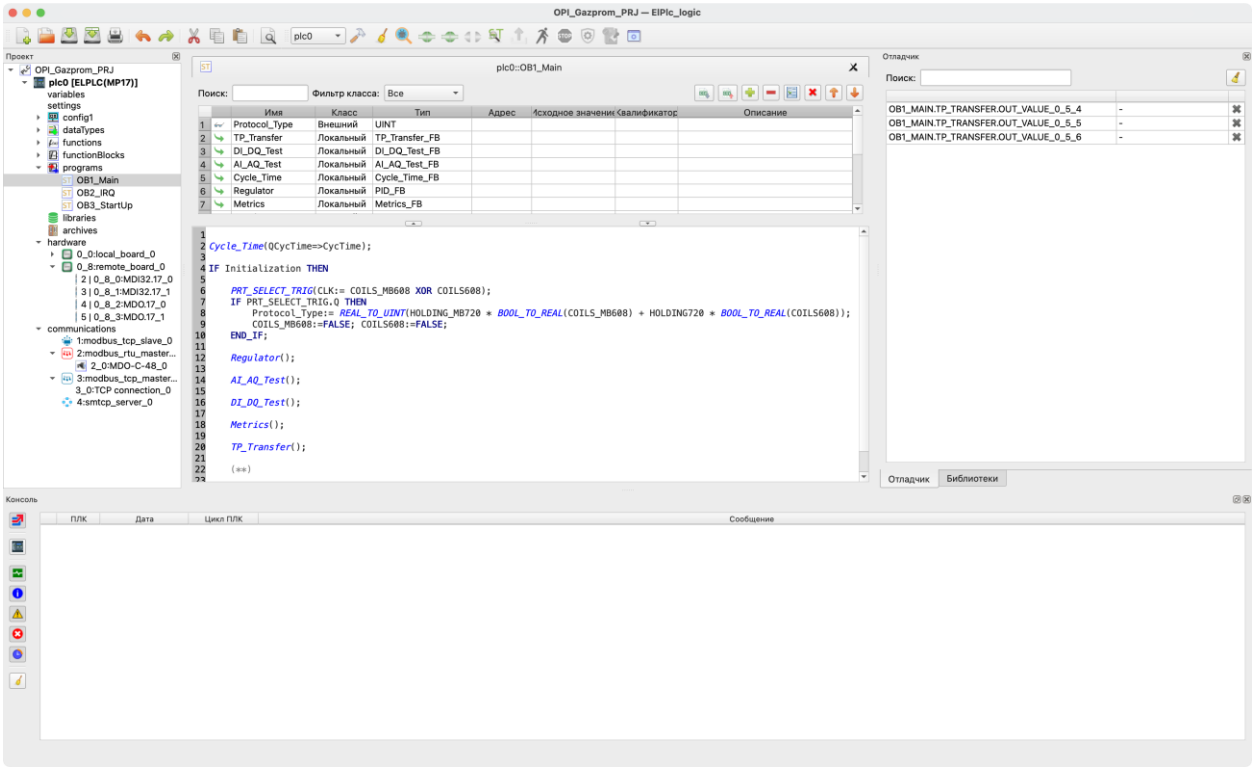


Рисунок 2 - Основное окно ПО ELPLC-LOGIC

Комбинация окон и вкладок (за исключением дерева проекта и редакторов) может быть изменена пользователем для удобства работы с ПО.

Представленная на рисунке 2 комбинация является стандартной (по умолчанию).

5.1. Меню программы

В верхней части основного окна (см. Рисунок 3) находится главное меню программы (см. рисунок 3).

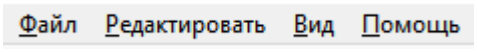


Рисунок 3 - Главное меню программы

Меню программы состоит из:

- меню «Файл»;
- меню «Редактировать»;
- меню «Вид»;
- меню «Помощь».

Пункты меню могут быть вызвана либо при помощи манипулятора типа «Мышь» (далее по тексту мышь) и/или комбинацией клавиш на клавиатуре – «горячая клавиша». Описание пунктов меню и комбинации клавиш представлено ниже.

5.1.1. Меню «Файл»

Меню «Файл» предназначено для работы с проектом.

На рисунке (см. Рисунок 4) представлен внешний вид меню «Файл»

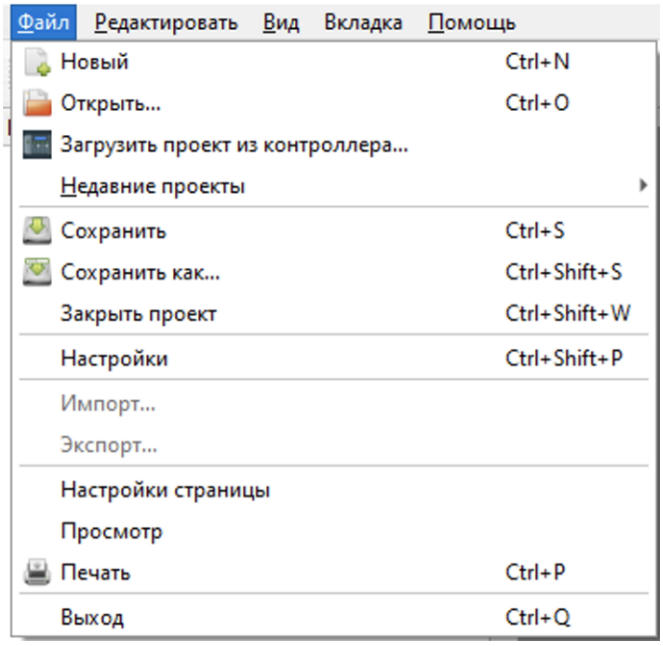


Рисунок 4 - Внешний вид меню «Файл»

Меню «Файл» содержит следующие пункты:

- «Новый» – создание нового проекта (CTRL + N);
- «Открыть» – открытие существующего проекта (CTRL + O);
- «Загрузить проект из контроллера» – скачать последний загруженный проект с ПЛК и сохранить его локально в IDE;
- «Недавние проекты» – быстрое открытие одного из десяти последних, недавно редактированных проектов;
- «Сохранить» – сохранение текущего проекта пункт (CTRL + S);
- «Сохранить как» – сохранение текущего проекта в папку отличную от той, в которой он сохранён на данный момент (CTRL + SHIFT + S);

- «Закрыть вкладку» – закрытие активной вкладки (например, вкладки переменных плагина, конфигурации и т.д.) для открытого проекта (CTRL + W);
- «Закрыть проект» – закрыть текущий, открытый проект (CTRL + SHIFT + W);
- «Параметры страницы» – настройка параметров страницы для печати на принтере активной программы, представленной в виде диаграммы (CTRL + ALT + P);
- «Предпросмотр» – предпросмотр перед печатью на принтере активной программы (CTRL + SHIFT + P);
- «Печать» – печать на принтере активной программы (CTRL + P);
- «Выход» – закрытие текущего проекта и выход из ПО ELPLC-LOGIC (CTRL+ Q).

5.1.2. Меню «Редактировать»

Меню «Редактировать» предназначено для работы с редакторами языков стандарта МЭК 61131-3 и предоставляет следующие возможности.

На рисунке (см. Рисунок 5) представлен внешний вид меню «Редактировать»

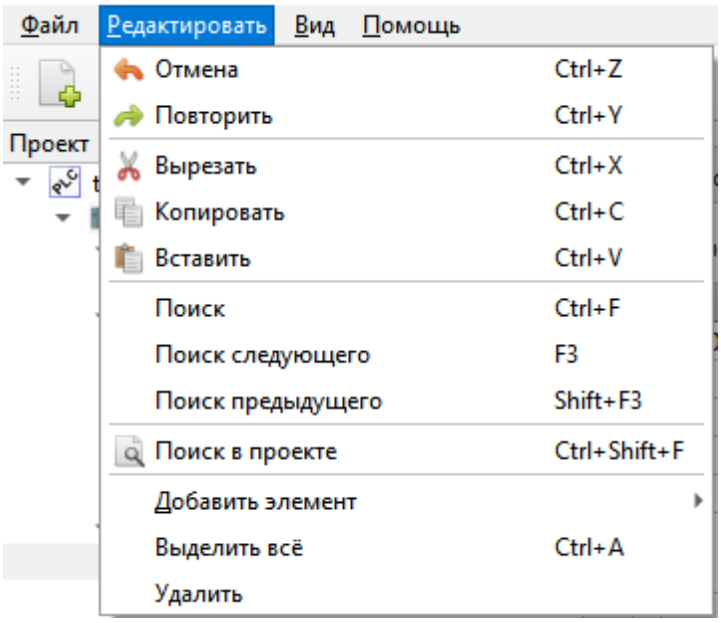


Рисунок 5 - Внешний вид меню «Редактировать»

Меню «Редактировать» содержит следующие пункты:

- «Отменить» – отмена последней манипуляции в редакторе (CTRL + Z);
- «Повторить» повтор отменённой манипуляции в редакторе (CTRL + Y);

- «Вырезать» – удалить в буфер обмена выделенный(е) элемент(ы) в редакторе (CTRL + X);
- «Копировать» – копировать в буфер обмена выделенный(е) элемент(ы) в редакторе (CTRL + C);
- «Вставить» – вставить из буфера обмена находящиеся там элемент(ы) в редактор (CTRL + V);
- «Поиск в проекте» – вызов диалога поиска данных в проекте (CTRL + SHIFT + F);
- «Добавить элемент» – добавление одного из следующих элемента в текущий проект:
- «Тип данных» – нового типа данных;
- «Функция» – новой функции;
- «Функциональный блок» – нового функционального блока;
- «Программа» – новую программу;
- «Ресурс» – новый ресурс;
- плагины для различных внешних модулей;
- «Выделить всё» – выделение всех элементов в активной вкладке редактора (CTRL + A);
- «Удалить» – удаление программного модуля, выделенного в дереве проекта.

5.1.3. Меню «Вид»

Меню «Вид» предназначено для работы с редакторами языков стандарта МЭК 61131-3:

На рисунке (см. Рисунок 6) представлен внешний вид меню «Вид»

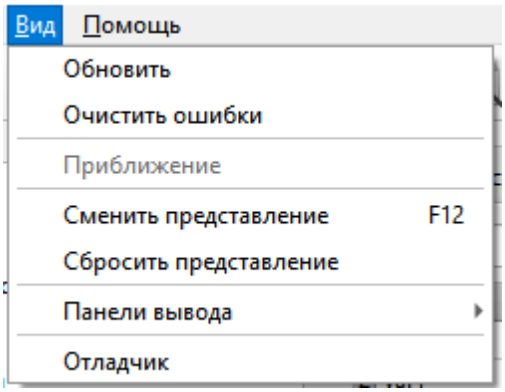


Рисунок 6 - Внешний вид меню «Вид»

Меню «Вид» содержит следующие пункты:

- «Обновить» – обновление данных и снятие выделения в редакторе (CTRL + R);
- «Очистить ошибки» – очистка указателей ошибок в редакторе (CTRL + K);
- «Масштаб» – пункт, в котором можно выбрать в процентах величину масштаба;
- «Сброс расположения панелей» – восстановление расположения вкладок ПО ELPLC-LOGIC в исходное состояние.

5.1.4. Меню «Вкладка»

Меню «Вкладка» предназначено для взаимодействия с открытыми страницами редактора. На рисунке (Рисунок 7) представлен внешний вид меню «Вкладка»

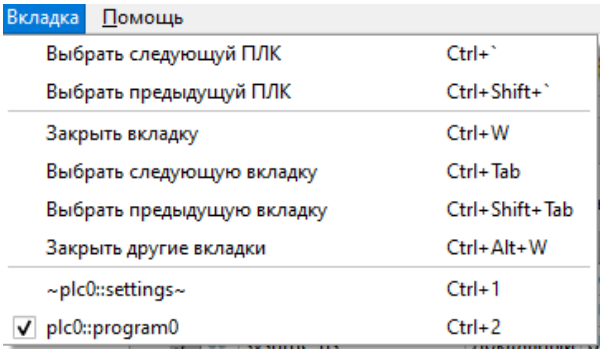


Рисунок 7- Внешний вид меню «Вкладка»

Меню «Вкладка» содержит следующие пункты:

- «Выбрать следующий ПЛК» – сделать следующий в дереве ПЛК активным (CTRL + `);
- «Выбрать предыдущий ПЛК» – сделать предыдущий в дереве ПЛК активным (CTRL + SHIFT + `);
- «Закрыть вкладку» – закрыть активное окно (CTRL + W);
- «Выбрать следующую вкладку» – циклическое переключение открытых окон вправо (CTRL + TAB);
- «Выбрать предыдущую вкладку» – циклическое переключение открытых окон влево (CTRL + SHIFT + TAB);
- «Закрыть другие вкладки» – закрыть все кладки редактора, кроме активной (CTRL + ALT + W);

5.1.5. Меню «Помощь»

Меню «Помощь» предназначено для обращения к выводу информации в виде диалога о создателях данной среды – пункт «О программе». Внешний вид меню представлен на рисунке (см. Рисунок 8).

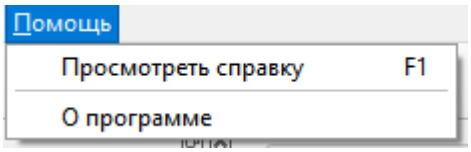


Рисунок 8 - Внешний вид меню «Помощь»

5.2. Панель инструментов

Панель инструментов представляет собой панель с кнопками для быстрого обращения к часто используемым функциям ПО ELPLC-LOGIC. Она состоит из нескольких панелей, содержащих кнопки: главного меню, сборки проекта и установки связи с целевым устройством. Подробнее об этих панелях рассказано ниже. При редактировании программных модулей, написанных на графических языках, появляются дополнительные панели с кнопками. Они рассмотрены при описании редакторов графических языков стандарта МЭК 61131-3.

Внешний вид панели инструментов представлен на рисунке (см. Рисунок 9).

Рисунок 9 – Внешний вид панели инструментов

Описание функций кнопок панели инструментов приведен в таблице (см. Таблица 2).

Таблица 2 – Кнопки панели инструментов

Внешний вид кнопки	Наименование кнопки	Функции кнопки
	Новый проект	Создать новый проект
	Открыть проект	Открыть существующий проект
	Сохранить проект	Сохранить текущий проект
	Сохранить проект как	Сохранить текущий проект в определённую папку
	Печать	Печать на принтере текущей программы
	Отменить	Отмена последнего действия в среде разработки
	Повторить	Повтор отменённого действия в среде разработки
	Вырезать	Удалить в буфер обмена выделенный(е) элемент(ы) в редакторе
	Копировать	Копировать в буфер обмена выделенный(е) элемент(ы) в редакторе
	Вставить	Вставить из буфера обмена находящиеся там элемент(ы) в редактор
	Поиск в проекте	Вызов диалога поиска данных в проекте
	Выделение объекта	Состояние, при котором возможно выделение объектов в редакторе с помощью мыши

6. ДЕРЕВО ПРОЕКТА

Слева основного окна ПО ELPLC-LOGIC статически расположено дерево проекта (см. Рисунок 10).

Дерево проекта отображает элементы, из которых состоит проект. Подробнее о элементах (функции, функциональные блоки, внешние модули и пр.) будет рассмотрено далее по данному руководству.

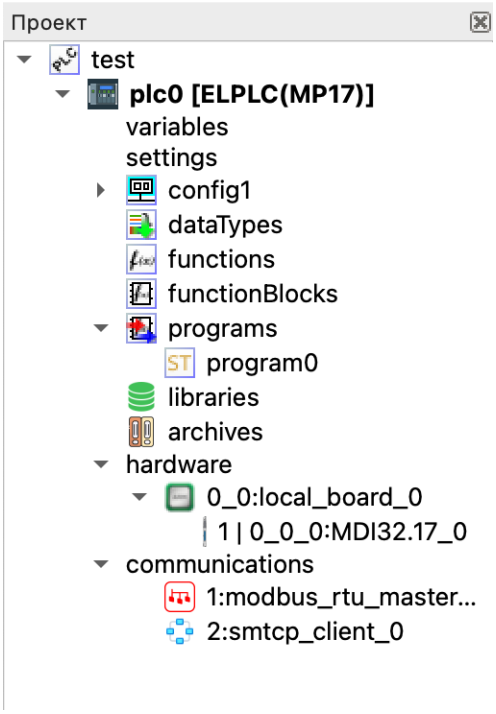


Рисунок 10 - Внешний вид дерева проекта

В роли элементов могут выступать:

- ресурсы;
- программные модули (функции, функциональные блоки и программ) и их составные части;
- типы данных;
- внешние модули (плагины).

Дерево проекта позволяет добавлять, удалять элементы. Операции копирования и вставки только доступны для программных модулей.

6.1. Добавление элементов в дерево проекта

Добавление элементов в дерево проекта происходит с помощью контекстного меню.

6.1.1. Контекстное меню добавления нескольких целевых устройств (ПЛК)

Контекстное меню добавления в проект одного или нескольких целевых устройств (ПЛК) представлено на рисунке (см. Рисунок 11).

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с именем проекта.

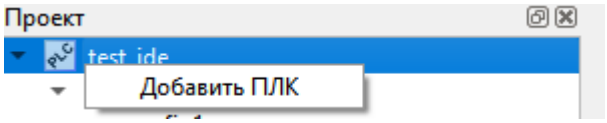


Рисунок 11 – Контекстное меню добавление целевых устройств (ПЛК)

6.1.2. Контекстное меню конфигурирования ПЛК

Добавление модулей, ресурсов и прочих элементов дерева к целевому устройству (ПЛК) происходит в контекстном меню, представленном на рисунке (см. Рисунок 12).

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с именем целевого устройства (ПЛК).

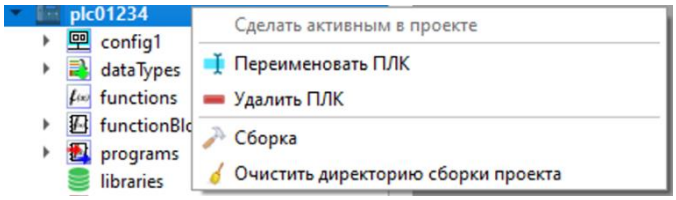


Рисунок 12 - Контекстное меню конфигурирования ПЛК

Контекстное меню состоит из следующих пунктов:

- «Сделать активным в проекте» - переключение коннектора, горячих клавиш и других элементов на выбранный ПЛК;
- «Переименовать ПЛК» - функции для изменения имени целевого устройства (ПЛК) в дереве проекта;
- «Удалить ПЛК» - удаление целевого устройства из дерева проекта;
- «Сборка» - собрать проект ПЛК в бинарный файл для системы исполнения;

- «Очистить директорию сборки проекта» - удалить все собранные ранее файлы;

6.1.3. Контекстное меню пользовательских типов данных

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева пользовательских типов данных (dataTypes) (Рисунок 13).

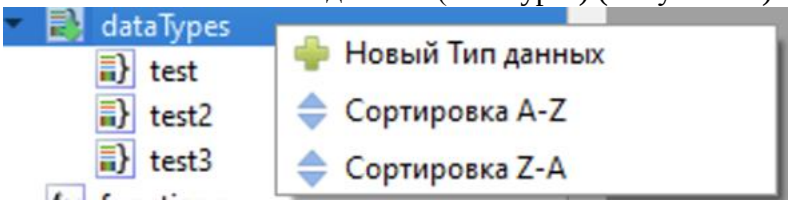


Рисунок 13 - Контекстное меню пользовательских типов данных

Контекстное меню состоит из следующих пунктов:

- «Тип данных» - объявление новых типов данных (согласно МЭК 61131-3) в проекте.
- «Сортировка A-Z» - сортировка списка элементов по имени по возрастанию.
- «Сортировка Z-A» - сортировка списка элементов по имени по убыванию.

6.1.4. Контекстное меню функций, функциональных блоков и программ

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с исполнительным элементом (functions, functionBlocks, programs) (Рисунок 14).

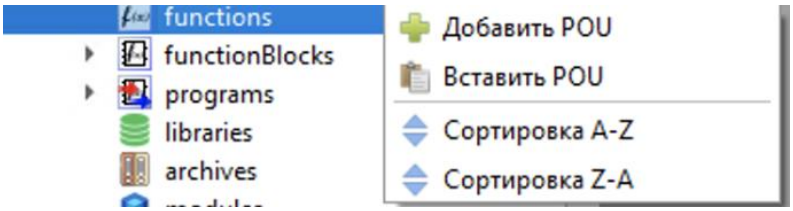


Рисунок 14 - Контекстное меню исполнительной программы

Контекстное меню состоит из следующих пунктов:

- «Добавить POU» - объявление нового исполнительного элемента (согласно МЭК 61131-3) в проект.
- «Вставить POU» -добавить ранее скопированный элемент.
- «Сортировка A-Z» - сортировка списка элементов по имени по возрастанию.
- «Сортировка Z-A» - сортировка списка элементов по имени по убыванию.

- «Функция» - пункт для объявления функций (согласно МЭК 61131-3) в проекте. При нажатии левой кнопкой мыши вызовется окно добавления функции с возможности выбора языка программирования ST, IL, FBD, SFC или LD;
- «Функциональный блок» - пункт для объявления функциональных блоков (согласно МЭК 61131-3) в проекте. При нажатии левой кнопкой мыши вызовется окно добавления функционального блока с возможности выбора языка программирования ST, IL, FBD, SFC или LD;
- «Программа» - пункт для объявления функций (согласно МЭК 61131-3) в проекте. При нажатии левой кнопкой мыши вызовется окно добавления программы с возможности выбора языка программирования ST, IL, FBD, SFC или LD;

ПРИМЕЧАНИЕ: При добавлении новой программы в дерево проекта, необходимо настроить ресурс для вызова ее в проекте см. пункт с настройками ресурса проекта.

6.1.5. Контекстное меню пользовательских библиотек

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с именем пользовательской библиотеки (libraries) (Рисунок 15).

Рисунок 15 - Контекстное меню пользовательской библиотеки

Контекстное меню состоит из следующих пунктов:

- «Новая пользовательская библиотека» - При нажатии левой кнопки мыши открывается окно, реализующие методы хранения функций, функциональных блоков, программ, типов данных, внешних модулей и прочих элементов дерева проектов во внешнем хранилище.

6.1.6. Контекстное меню системы архивирования

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с именем системы архивирования (archives) (Рисунок 16).

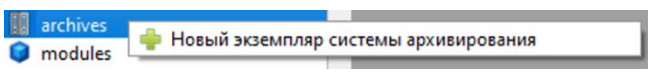


Рисунок 16 - Контекстное меню системы архивирования

Контекстное меню состоит из следующих пунктов:

- «Новая экземпляр системы архивирования» - при нажатии левой кнопки мыши открывается окно, реализующие функции архивирования переменных процесса на целевых устройствах ПЛК;

6.1.7. Контекстное меню модулей

Вызов контекстного меню происходит по нажатию правой клавиши мыши на элемент дерева с именем модули (communications) (Рисунок 17).

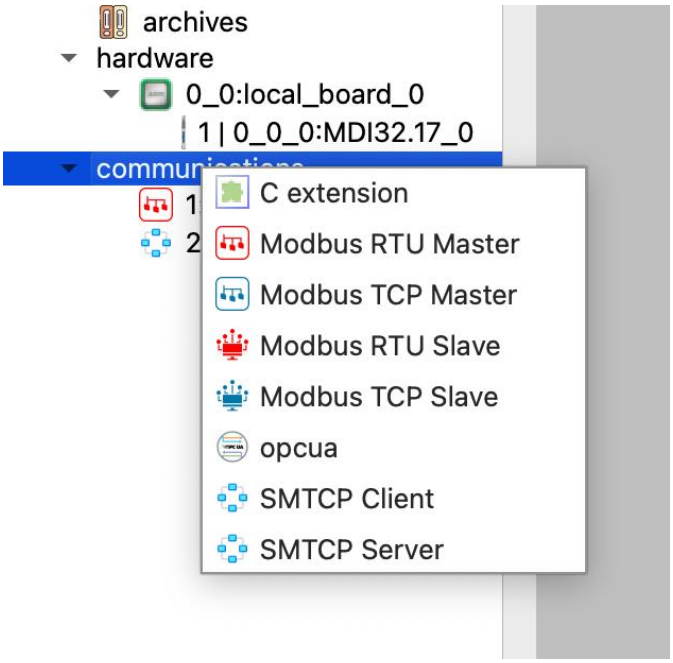


Рисунок 17- Контекстное меню модулей

- «C extension» - модуль расширения, позволяющий включить в проект микро программу на языке Си. При нажатии левой кнопки мыши в дерево проекта будет добавлен модуль «C extension» и будет доступен текстовый редактор для языка Си;

- «ModBus RTU Master», «ModBus TCP Master», «ModBus RTU Slave», «ModBus TCP Slave» - внешние модули (плагины) реализующие функции по настройке опроса/обработки запросов удалённых устройств по протоколу Modbus. При нажатии левой кнопки мыши в дерево проекта будет добавлен выбранный модуль «ModBus RTU

Master», «ModBus TCP Master», «ModBus RTU Slave» или «ModBus TCP Slave» и будет доступна вкладка для конфигурирования плагина;

- «opcua» - внешний модуль (плагин), реализующий функции OPC UA сервера. При нажатии левой кнопки мыши в дерево проекта будет добавлен выбранный модуль «opcua» и будет доступна вкладка для конфигурирования плагина;

- «SMTCP Client», «SMTCP Server» - внешние модули (плагины) реализующие функции TCP/IP клиентов или сервера с проприетарным протоколом обмена данными. Модули являются зарегистрированными РИД ПАО «ИНЭУМ им. И.С. Брука». При нажатии левой кнопки мыши в дерево проекта будет добавлен выбранный модуль SMTCP Client» или «SMTCP Server» и будет доступна вкладка для конфигурирования плагина.

-

6.2. Удаление элемента в дереве проекта

Удаление осуществляется наведением на определённый элемент в дереве проекта и нажатием на него правой кнопки мыши, а далее в появившемся меню выбирается пункт «Удалить» (см. Рисунок 18).

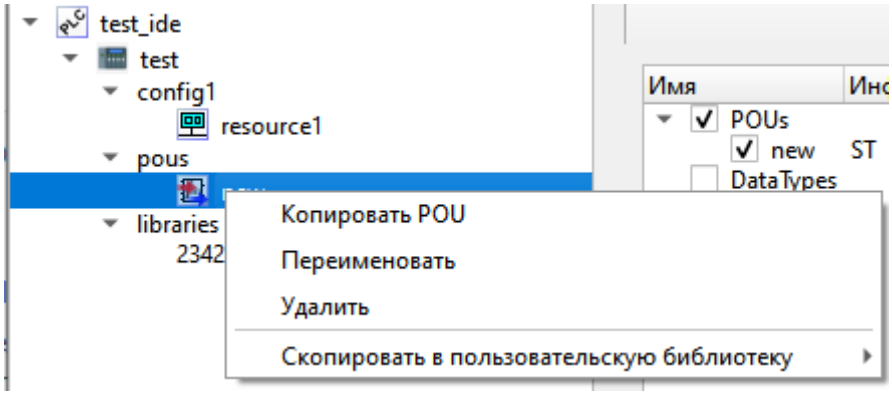


Рисунок 18 – Пример удаления элемента дерева проекта

6.3. Переименование, копирование и вставка программных модулей

Дерево проекта позволяет выполнять операции переименования, копирования и вставки для программных модулей. Копирование или переименование осуществляются с помощью нажатия правой клавиши мыши на элемент, соответствующий программному модулю в дереве проекта, и выбор соответствующего пункта появившегося меню.

Копирование, вставка и переименование внешних модулей (плагинов), модулей дерева проекта, которые не имеют в контекстном меню функции копирования, вставки или переименования, производится с помощью функций «Пользовательских библиотек».

7. РАБОТА С ПРОЕКТОМ

В данном разделе рассмотрены основные приёмы работы в ПО ELPLC-LOGIC, которые необходимы при создании прикладной программы.

7.1. Создать новый проект

Новый проект создаётся с помощью главного меню «Файл» – «Новый» (см. Рисунок 19), либо с помощью кнопки «Новый» ( ) на панели управления.

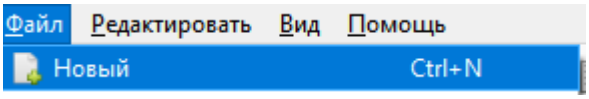


Рисунок 19 – Создание нового проекта. Вызов меню «Новый»

Далее появится диалог (см. Рисунок 20 и Рисунок 21), в котором необходимо выбрать папку, где будет храниться данный проект.

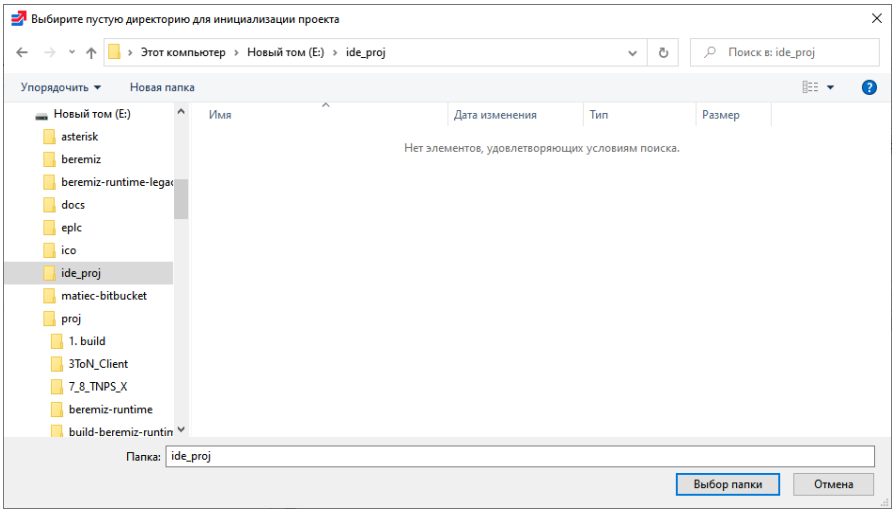


Рисунок 20 – Диалоговое окно выбора папки нового проекта (стандартное диалоговое окно Windows 10)

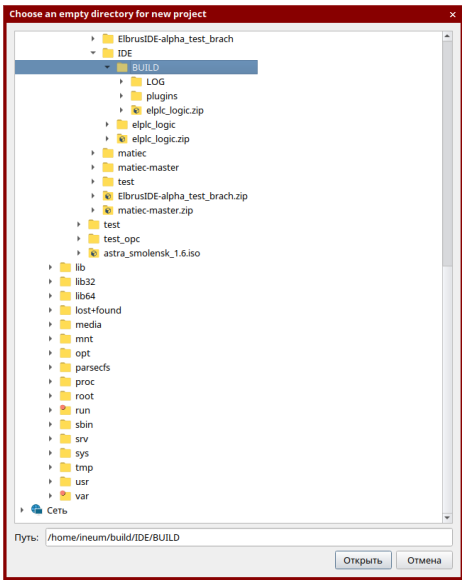


Рисунок 21 – Диалоговое окно выбора папки нового проекта (стандартное диалоговое окно Astra Linux SE)

Папка должна быть обязательно пустой и не защищена от записи. Если в папке уже есть файлы, будет выдана соответствующая ошибка.

После выбора папки удовлетворяющая условиям указанные выше, в ПО ELPLC-LODGIC выведется диалоговое окно (см. Рисунок 22) с возможностью создания проекта по умолчанию с функциями:

- задание имени целевого устройства (ПЛК);
- установка флага создания программы по умолчанию и выбора языка программирования;
- задание имени программы;
- выбор ПЛК из списка поддерживаемых контроллеров

Имя ПЛК является уникальным именем контроллера в проекте, верхний элемент дерева. Система логирования, вывод ошибок и другая информация пользователю будет содержать данное выбранное имя.

При задании флага «Создать программу по умолчанию» будет выбрано стандартное имя ПЛК (plc0) и стандартное имя программы (program0).

Для инициализации платформозависимых компонентов системы необходимо выбрать тип контроллера, на котором будет запускаться проект. В дереве узлами являются контроллеры, а элементами этих узлов – процессорный модуль или архитектура, на

котором реализован процессор. В правой части отображается схематическое изображение устройства, серийный номер, версии контроллера (необходимо для обратной совместимости при обновлении), аппаратные возможности (плагин, описывающий настройки, специфичные для конкретного вида ПЛК) и коммуникации (библиотеки и расширения, доступные в пользовательской программе). В данном примере выбран тип Linux X86. Данный тип используется при запуске на X86-совместимых компьютерах, виртуальных машинах и других системах с программным ПЛК.

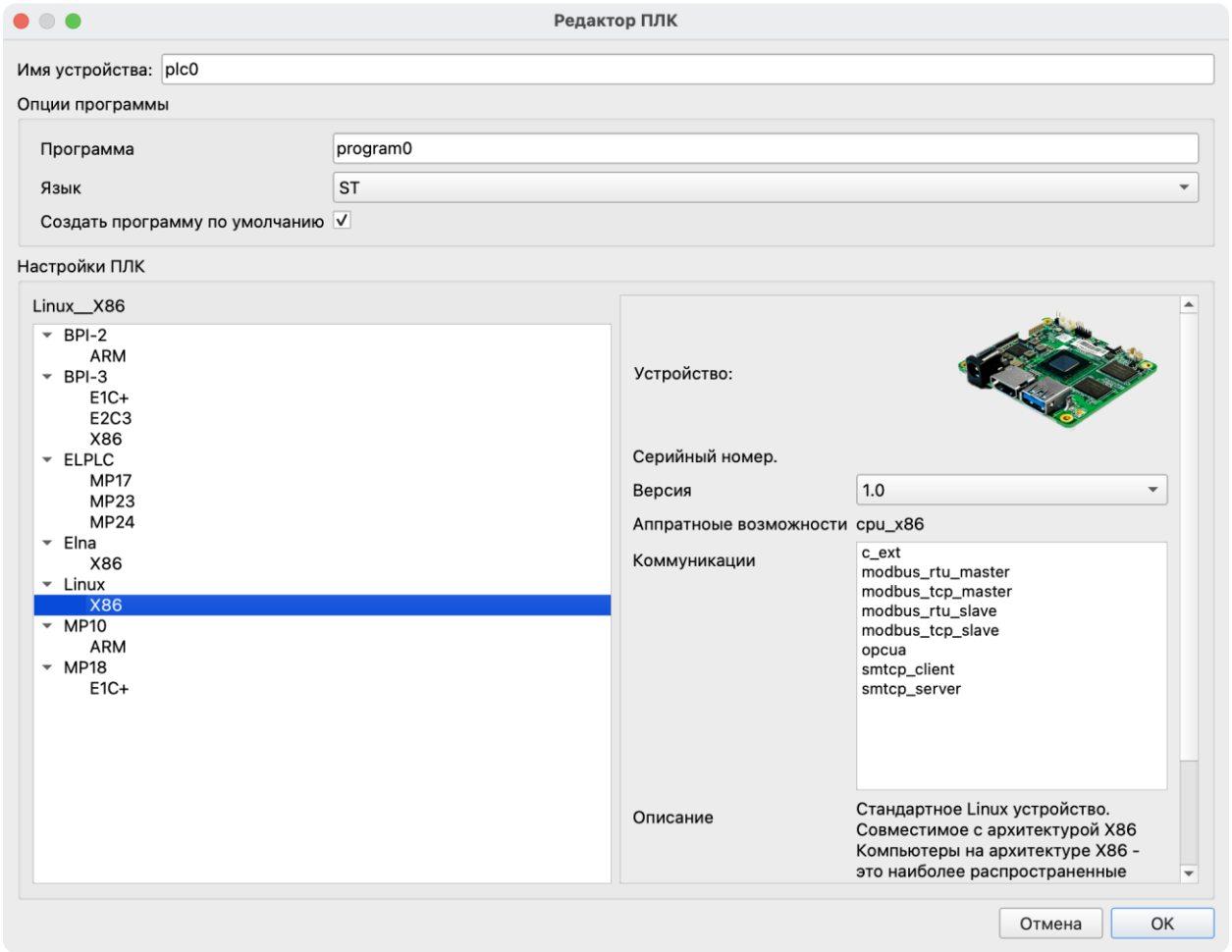


Рисунок 22 – создание проекта по умолчанию

После нажатие клавиши «ОК» в диалоговом окне создается проект в выбранной директории. В дереве проекта отображаются элементы, созданные по умолчанию (см. Рисунок 23).

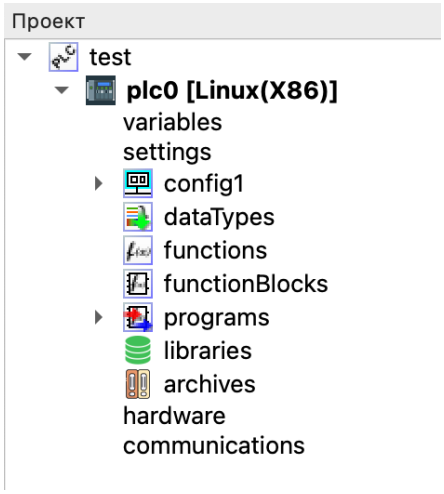


Рисунок 23 – Созданный новый проект программы (проект по умолчанию)

7.2. Открыть проект

Открыть ранее созданный проект можно с помощью главного меню «Файл» – «Открыть», либо с помощью кнопки «Открыть» () на панели управления

7.3. Программные модули

Добавление программных модулей (программ, функций, функциональных блоков) осуществляется с помощью всплывающего меню дерева проекта, в котором необходимо выбрать пункт «Функция», «Функциональный блок» или «Программа». Далее появится диалог создания нового программного модуля (см. Рисунок 24).

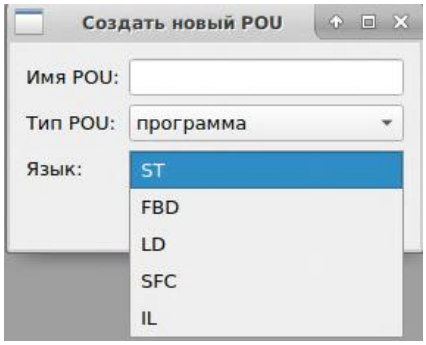


Рисунок 24 - Диалог создания программного модуля

В данном диалоге три поля:

- «Имя программного модуля»;
- «Тип программного модуля»;

- «Язык».

Имя, присвоенное по умолчанию, может быть заменено на имя, соответствующее назначению данного программного модуля. В зависимости от того, какой программный модуль был выбран во всплывающем меню, в поле «Тип программного модуля» будет подставлено именование данного программного модуля. В поле «Язык» необходимо выбрать из списка один из языков стандарта IEC 61131-3 (IL, ST, LD, FBD, SFC), на котором будет реализованы алгоритмы и логика работы данного добавляемого программного модуля.

Внимание! Проект должен иметь по крайней мере одну программу.

Далее рассмотрено добавление каждого программного модуля в отдельности.

7.3.1. Программы, функции, функциональные блоки

Ниже будет приведён пример добавления в проект программы, написанной на языке FBD. Логика и алгоритм работы данного программного модуля, следующие: определены две глобальные переменные «globalValue» и «globalLevel», если значение «globalValue» больше 10.0, то присвоить переменной «globalLevel» значение 100, в противном случае присвоить «globalLevel» значение 50.

Сначала следует добавление программы в проект, осуществляемое с помощью меню дерева проекта, выбором пункта «Программа». В появившемся диалоге выбирается язык FBD и нажимается кнопка «OK».

Далее в открывшейся вкладке с панелью редактирования данного программного модуля в панели переменных и констант (см. Рисунок 25) добавляются переменные: «globalValue» типа REAL, класса «Внешняя» и «globalLevel» типа INT, класса «Внешняя».

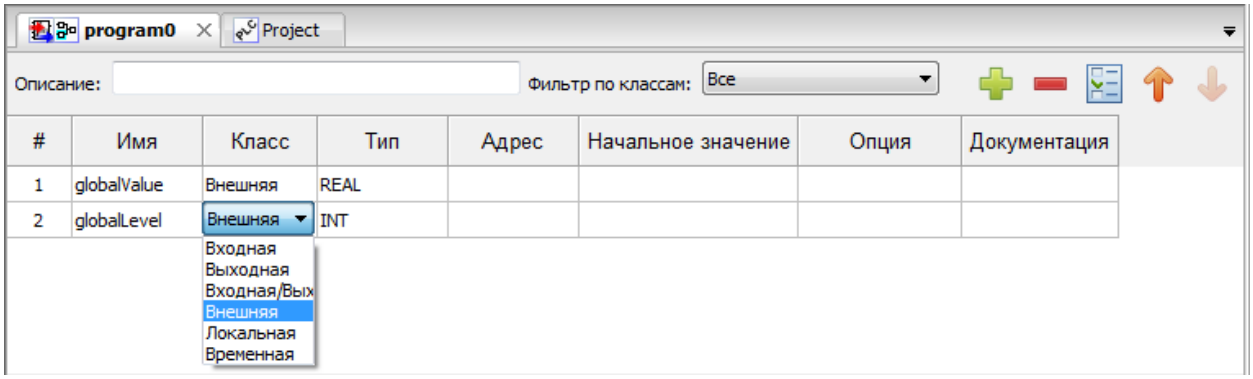


Рисунок 25 - Объявление в программе внешних переменных

Предполагается, что эти переменные уже определены глобальными переменными проекта в окне описания ресурсов.

Далее необходимо обратиться к редактору языка FBD. Для написания алгоритма и логики выполнения данной программы будут добавлены две функции: «GT» и «SEL». Функция «GT» обозначает сравнение «Больше чем» и находится во вкладке «Операции сравнения». Она может содержать от 2 до 20 входных значений (в данном примере их будет 2) и одно выходное значение «OUT». Если значение «IN1» больше значения «IN2», то на выходе «OUT» будет TRUE, в противном случае FALSE.

Функция «SEL» обозначает «Выбор одного из двух значений» и находится во вкладке «Операции выбора». Она содержит три входных переменных «G», «IN0», «IN1» и одну выходную «OUT». Если «G» равно 0 (или FALSE), то выходной переменной «OUT» присваивается значение «IN0». Если «G» равно 1 (или TRUE), то выходной переменной «OUT» присваивается значение «IN1».

Добавление данных функций удобнее осуществить переносом соответствующей функции с помощью мыши (Drag&Drop) из панели библиотеки функций и функциональных блоков в область редактирования FBD диаграммы данного программного модуля (см. Рисунок 26).

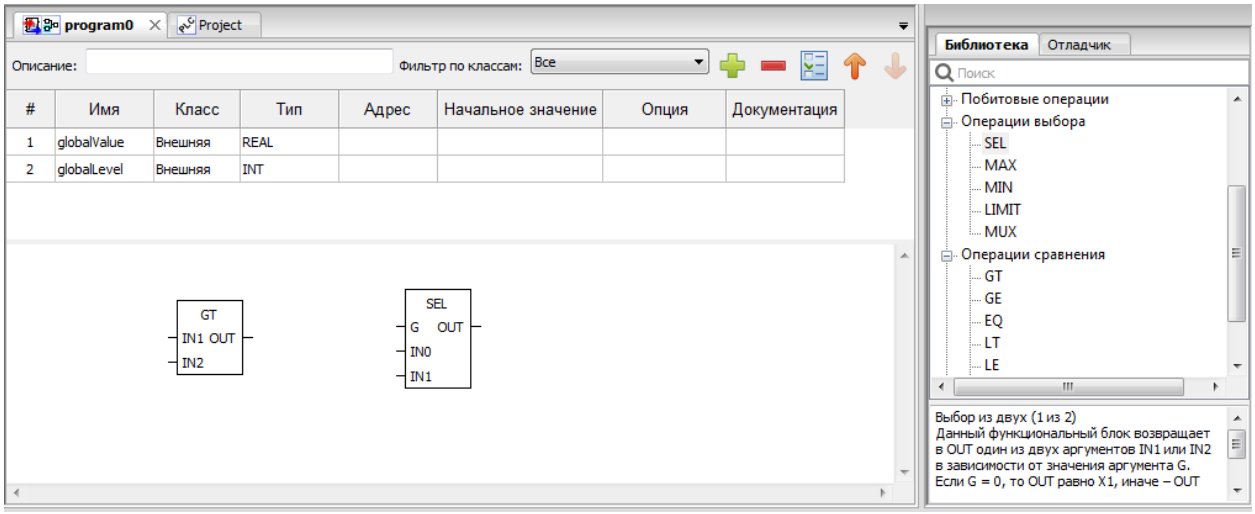


Рисунок 26 - Добавление двух функций на FBD диаграмму

Далее, так же используя мышь, переносятся переменные «globalValue» и «globalLevel» на FBD диаграмму. Как уже упоминалось ранее, необходимо левой клавишей мыши зажать столбец «#» для переменной в панели переменных и констант,

далее перенести указатель на область редактирования FBD диаграммы и отпустить кнопку мыши (Drag&Drop). Такую манипуляцию нужно произвести для обеих переменных.

Переменную «globalValue» можно сразу соединить с входом «IN1» функции «GT». Чтобы переменную «globalValue» можно было соединить с выходом «OUT» функции «SEL», необходимо сделать двойной щелчок левой кнопкой мыши по «globalValue» на FBD диаграмме и в появившемся диалоге «Свойства переменной» указать класс «Выходная» (см. Рисунок 27).

После этого переменную «globalValue» можно соединить с выходом «OUT» функции «SEL» и выход «OUT» функции «GT» с входом «G» функции «SEL».

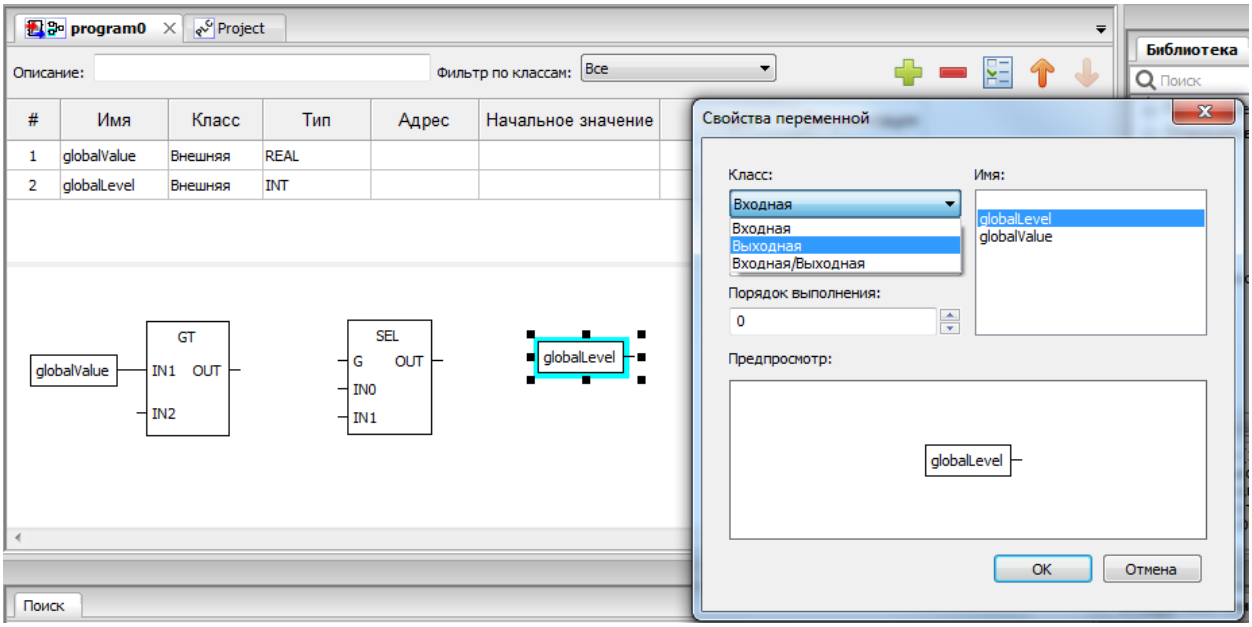


Рисунок 27 - Указание класса «Выходная» для переменной

Далее необходимо добавить 3 числовых литерала, которые будут напрямую соединены с входом «IN2» функции «GT» и входами «IN0» и «IN1» функции «SEL». В панели редактирования FBD диаграммы выбирается кнопка «Добавить переменную» и в появившемся диалоге «Свойства переменной» в поле выражения пишется «10.0». Нажимается кнопка «OK».

Добавленный литерал соединяется с входом «IN2» функции «GT». Аналогичным образом создаются литералы со значениями 50 и 100 для входов «IN0» и «IN1» функции «SEL» и соединяются с ними.

Соответственно, в случае если значение переменной «globalValue» больше 10.0, то на выходе «OUT» функции «GT» будет значение TRUE, тем самым на вход функции «SEL» поступит тоже True и выходное значение «OUT» будет равно «IN1», т.е. 100.

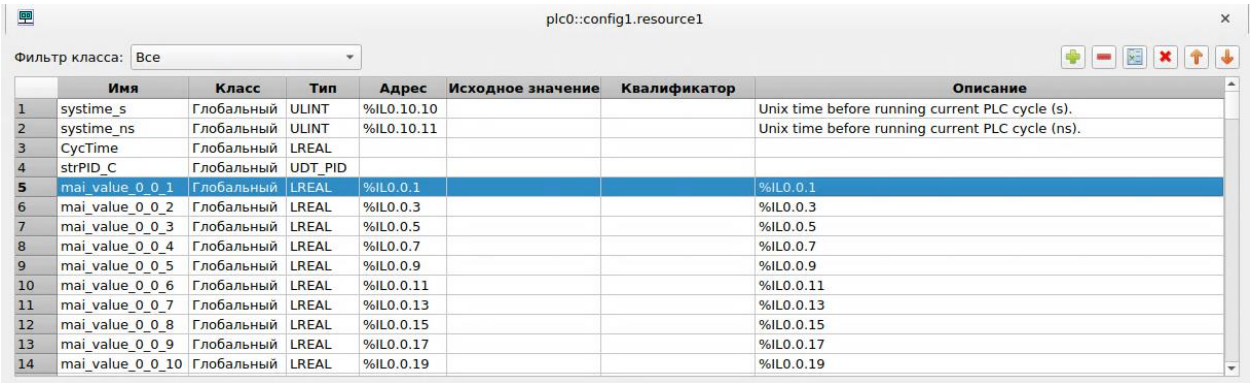
Функции и функциональные блоки добавляются аналогичным образом.

7.4. Ресурсы

Согласно стандарту IEC 61131-3, каждый проект должен иметь как минимум один ресурс, с определённым в нём как минимум одним экземпляром. Экземпляр представляет собой элемент, связанный с программным модулем типа «Программа» и одной определённой задачей. По умолчанию, среда разработки ELPLC-LOGIC создаёт для нового проекта один ресурс.

7.4.1. Глобальные переменные ресурса

Глобальные переменные ресурса объявляются аналогично глобальным переменным проекта на панели переменных и констант выбранного ресурса с использованием кнопки «Добавить переменную», либо «Добавить переменные».



	Имя	Класс	Тип	Адрес	Исходное значение	Квалификатор	Описание
1	systime_s	Глобальный	ULINT	%IL0.10.10			Unix time before running current PLC cycle (s).
2	systime_ns	Глобальный	ULINT	%IL0.10.11			Unix time before running current PLC cycle (ns).
3	CycTime	Глобальный	LREAL				
4	strPID_C	Глобальный	UDT_PID				
5	mai_value_0_0_1	Глобальный	LREAL	%IL0.0.1			%IL0.0.1
6	mai_value_0_0_2	Глобальный	LREAL	%IL0.0.3			%IL0.0.3
7	mai_value_0_0_3	Глобальный	LREAL	%IL0.0.5			%IL0.0.5
8	mai_value_0_0_4	Глобальный	LREAL	%IL0.0.7			%IL0.0.7
9	mai_value_0_0_5	Глобальный	LREAL	%IL0.0.9			%IL0.0.9
10	mai_value_0_0_6	Глобальный	LREAL	%IL0.0.11			%IL0.0.11
11	mai_value_0_0_7	Глобальный	LREAL	%IL0.0.13			%IL0.0.13
12	mai_value_0_0_8	Глобальный	LREAL	%IL0.0.15			%IL0.0.15
13	mai_value_0_0_9	Глобальный	LREAL	%IL0.0.17			%IL0.0.17
14	mai_value_0_0_10	Глобальный	LREAL	%IL0.0.19			%IL0.0.19

Рисунок 28 - Глобальные переменные ресурса

Использование данных глобальных переменных на уровне ресурса также аналогично использованию глобальных переменных проекта в программных модулях.

В программном модуле эти переменные можно добавить с классом «Внешняя», с такими же именами. Это позволит получить доступ к этим переменным из нескольких программ.

7.4.2. Задачи и экземпляры ресурса

Для создания экземпляра необходимо наличие как минимум одного программного модуля типа «Программа» в проекте и как минимум одной задачи, определённой в панели редактирования ресурса.

После добавления задачи с помощью кнопки «Добавить» (данная кнопка аналогична кнопке «Добавить» на панели переменных и констант), необходимо задать её уникальное имя (поле «Имя») и выбрать тип выполнения задачи (поле «Запуск», см. Рисунок 29):

- «Цикличное» – выполнение программного модуля типа «Программа» через заданный интервал времени, указанный в поле «Интервал»;
- «По условию» – выполнение программного модуля типа «Программа» один раз при наступлении значения TRUE глобальной переменной типа BOOL, определённой на уровне проекта, либо на уровне ресурса, указанной в поле «Флаг условия».

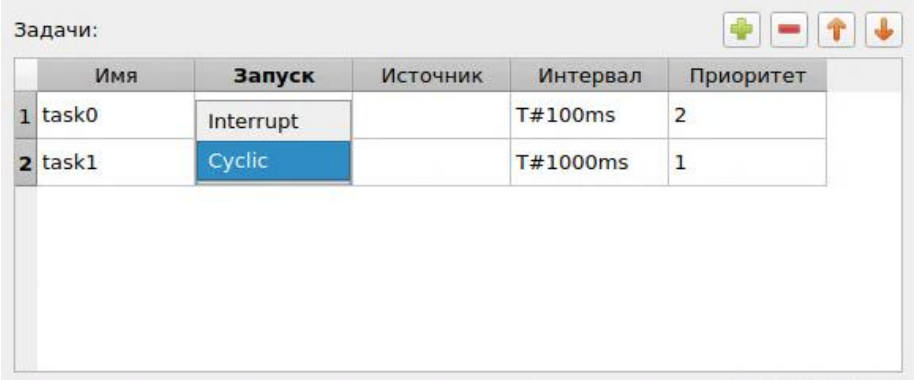


Рисунок 29 - Задачи ресурса

В случае выбора типа выполнения «Цикличное», в поле «Интервал» необходимо указать интервал, с которым будет выполняться данная задача. Двойной щелчок левой кнопкой мыши по полю «Интервал» приводит к появлению диалога задания временного интервала.

В случае выбора типа выполнения «По условию» в поле «Источник» необходимо указать переменную типа BOOL, определённую глобально либо на уровне проекта, либо на уровне ресурса.

Задача будет выполнена один раз, как только значение переменной, определённой в этом поле, будет TRUE.

Поле «Приоритет» позволяет указать приоритет выполнения задачи, по умолчанию все задачи имеют приоритет 0. Следует отметить, что в ресурсе должна быть определена как минимум одна задача с типом выполнения «Цикличное», в противном случае будет ошибка в компиляции в отладочной консоли.

После того как задачи определены, их можно использовать в экземплярах. Создание экземпляра происходит аналогичным образом с помощью кнопки «Добавить». Необходимо выбрать уникальное имя экземпляра и далее указать программный модуль типа «Программа» в поле «Тип» и одну из задач в поле «Задача» (см. Рисунок 24).

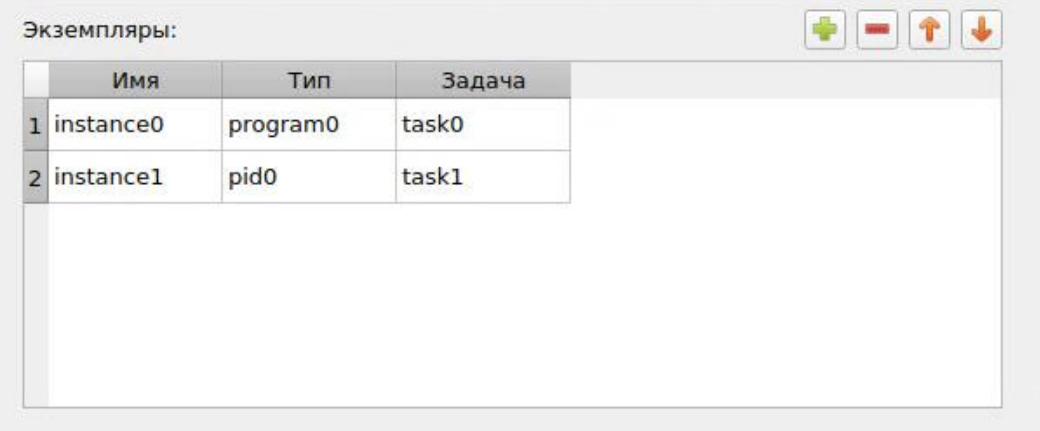


Рисунок 30 - Создание экземпляра

7.5. Сохраняемые переменные (Retain)

Система retain переменных предназначена для сохранения текущих значений переменных в программе, исполняемой на контроллере в энергонезависимую память. При старте программы эти переменные считываются с носителя, а при каждом изменении сохраняются на диск.

Создать retain переменную можно, указав соответствующий квалификатор, при добавлении переменной в программу.

Внимание! Переменная может быть отмечена как retain только если создана в программе. Глобальная переменная не может быть объявлена как retain. В случае необходимости использования значения retain переменной в нескольких программах используйте глобальные переменные, присвоив им значения, считанные из retain переменной в программе.

Внимание! Не рекомендуется отмечать переменные как «сохраняемые» в случае их частого изменения по ходу выполнения программы. Это может замедлить цикл исполнения, а также будет вырабатывать ресурс твердотельных носителей ПЛК.

При обновлении проекта на ПЛК запускается система разрешения конфликтов хранимых переменных. Переменные проекта и целевого устройства сравниваются и пользователю предлагается диалог для решения конфликтов, если они есть. Для каждой переменной можно выбрать один из следующих сценариев:

- обновить значение в программе из ПЛК (при этом будет выполнена повторная сборка проекта и отправка бинарного файла на контроллер);
- инициализация переменной значением из программы (текущее значение, полученное на последнем цикле ПЛК будет утрачено и перезаписано значением из POU);
- продолжить выполнение (значение переменной будет продолжено с момента остановки контроллера).

Диалоговое окно разрешения конфликтов представлено на рисунке (см. Рисунок 31).

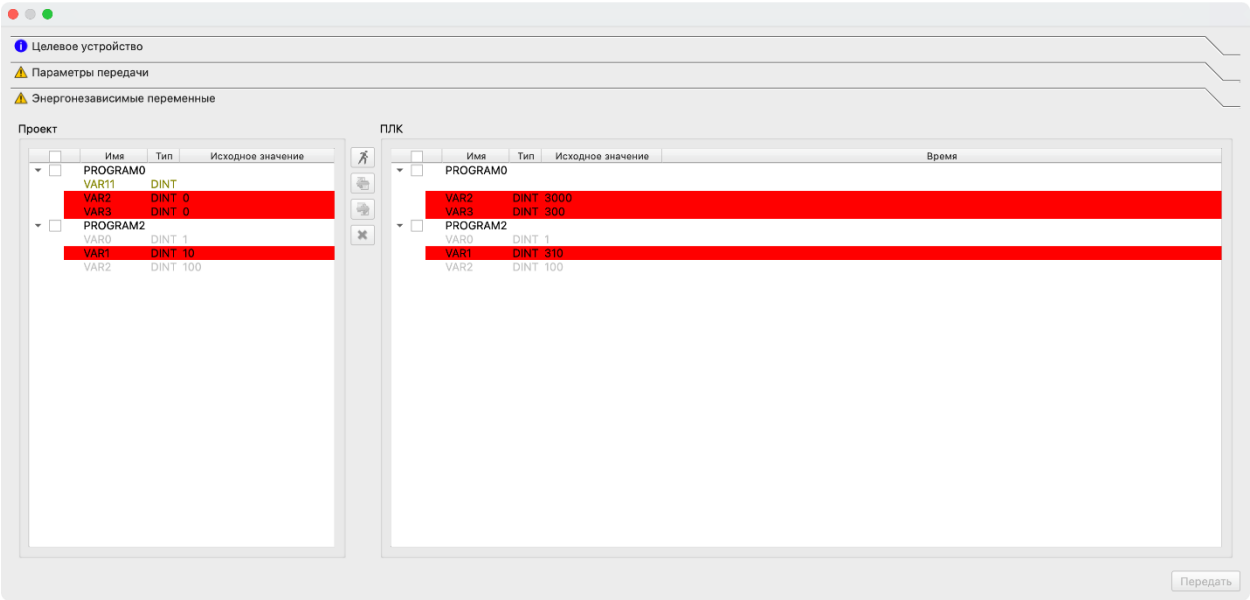


Рисунок 31 - Окно разрешения конфликтов хранимых переменных

Переменные помечаются следующими цветами:

- красный – переменная удалена или изменена на контроллере;
- желтый – переменная добавлена в проект, но на контроллере ее не было;
- серый – значение переменной в проекте совпадает с контроллером.

Интерфейс корректировки значений выполнен с использованием следующих элементов:

-  – продолжить выполнение;
-  – обновить значение переменной в проекте;
-  – использовать значение переменной из программы;
-  – отменить изменения.

Необходимо выбрать активные элементы и выбрать действие выше.

В случае удаления переменных из проекта, дополнительно нужно будет подтвердить данное действие.

7.6. Настройка проекта

Для того чтобы приступить к работе с проектом необходимо ввести данные о проекте, авторе и др. индивидуальные параметры проекта. Для открытия вкладки со свойствами, необходимо нажать двойным кликом левой клавиши мыши на имя проекта в «дереве проекта».

Вкладка «Свойства проекта» (см. Рисунок 32) позволяет задать: имя проекта, версию проекта, имя продукта, версию продукта и релиз продукта

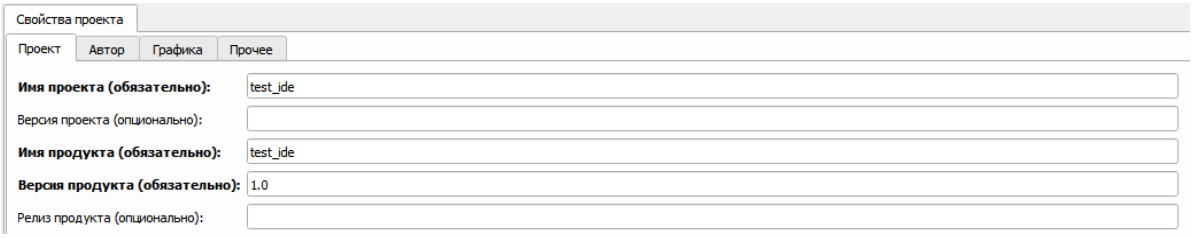


Рисунок 32 - Вкладка «Свойства проекта»

Вкладка «Проект» (см. Рисунок 32) позволяет задать: имя проекта, версию проекта, имя продукта, версию продукта и релиз продукта.

Вкладка «Автор» (см. Рисунок 33) позволяет задать: Имя компании, URL-адрес компании, Имя автора, Название организации.

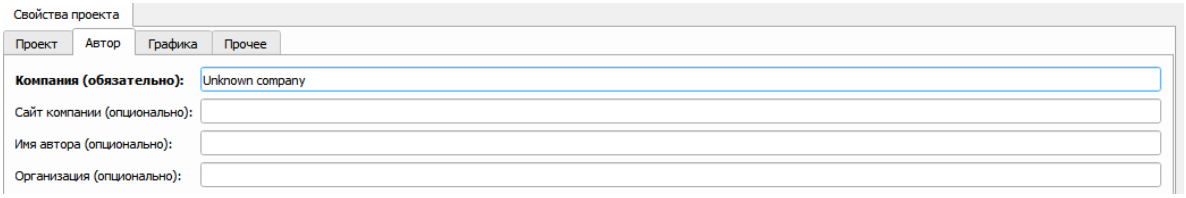


Рисунок 33 - Вкладка «Автор»

Вкладка «Графика» (см. Рисунок 34) позволяет задать размеры страницы и разрешение сетки для редакторов диаграмм графических языков FBD, LD и SFC.

Рисунок 34 - Вкладка «Графика»

Вкладка «Прочее», изображенная на рисунке (см. Рисунок 35), позволяет выбрать язык и указать дополнительное текстовое описание для проекта.

Рисунок 35 - Вкладка «Прочее»

Примечание. Заполнение всех полей вкладки «Свойства проекта» не обязательны, значения, выставленные по умолчанию, будут сохранены в СУБД файле проекта.

7.7. Настройки сборки проекта и соединения с целевым устройством (ПЛК)

Для того что бы настроить сборку и передачу программы на целевое устройство (ПЛК), необходимо произвести специальные настройки.

Что бы открыть вкладку с настойками сборки проекта, необходимо развернуть необходимый плк (целевое устройство) в «дереве проекта» и нажать двойным кликом левой клавиши мыши на элемент settings.

Внешний вид вкладки настройки сборки проекта и соединения с целевым устройством (ПЛК) представлен на рисунке (см. Рисунок 36).

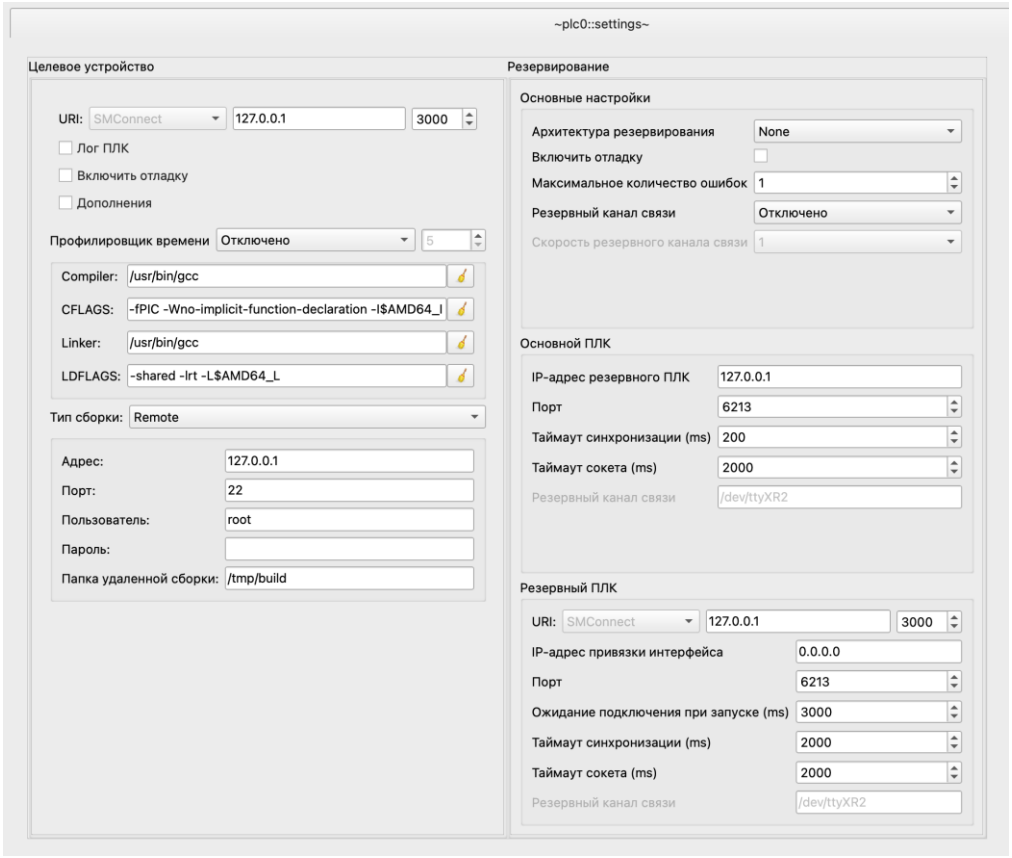


Рисунок 36 - Настройки сборки проекта и соединения с целевым устройством (ПЛК)

На вкладке слева представлены настройки компиляции и подключения к целевому устройству. Справа окно настройки системы резервирования.

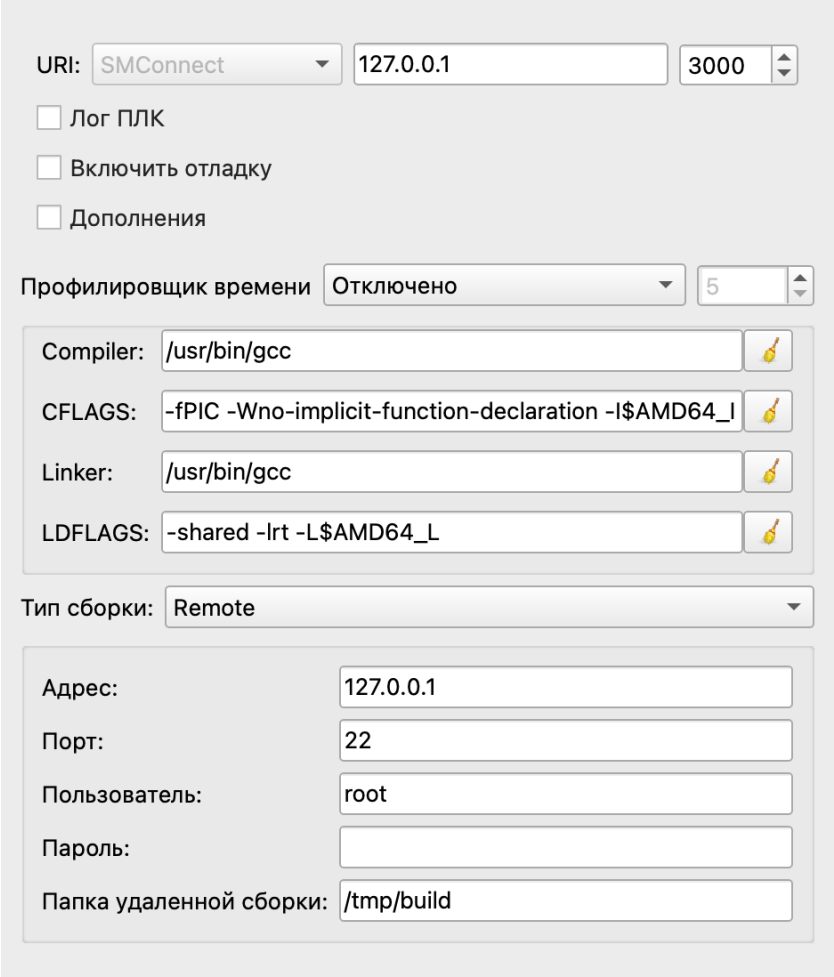


Рисунок 37 - Выбор целевой платформы для сборки прикладной программы

- URI – IP адрес целевого устройства должен быть известен и необходима возможность подключения к нему по локальной сети (требуется обратить внимание на сетевые настройки операционной системы). По умолчанию порт имеет номер 3000.
- «Компилятор» – имя исполняемого файла компилятора (если он определён в глобальных переменных среды), либо полный путь к нему;
- «CFLAGS» – указание флагов Си компилятора;
- «Компоновщик» – имя исполняемого файла компоновщика (если он определён в глобальных переменных среды), либо полный путь к нему;
- «LDFLAGS» – указание флагов компоновщика;
- «Окружение» – выбор из списка окружения, используемого для сборки проекта под конкретный ПЛК.

Далее стоит обратить внимание на «Тип сборки» (см. Рисунок 38)

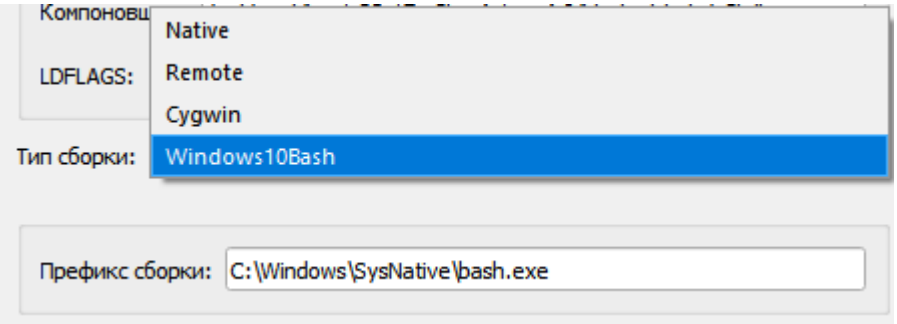


Рисунок 38 – Выбор типа сборки проекта

«Native» - означает что сборка проекта будет проходить с помощью вызова стантартных средств компиляции на устройстве где установлено ПО ELPLC-LOGIC.

«Remote» - настройка собрки проекта на удаленном устройстве через сеть TCP/IP (например для сборки на целевом устройстве (ПЛК). На рисунке (см. Рисунок 39) показаны настройки по умолчанию для удаленной компиляции проекта. Такой режим сборки поддерживается не всеми целевыми устройствами. Для работы этого режима необходимо, чтобы целевая платфома имела установленные в операционной системе средства сборки (компилятор).

«Cygwin» - UNIX-подобная среда и интерфейс командной строки для Microsoft Windows. Cygwin обеспечивает тесную интеграцию приложений, данных и ресурсов Windows с приложениями, данными и ресурсами UNIX-подобной среды. Данный тип сборки с помощью консоли Cygwin для операционных систем Windows.

«Windows10Bash» - слой совместимости для запуска Linux-приложений (двоичных исполняемых файлов в формате ELF) в ОС Windows 10 (WSL). Данный тип сборки с помощью консоли WSL для операционных систем Windows.

«Префикс сборки» - путь до терминала окружения (данные настройки могут быть по умолчанию для выбранной целевой платформы и типа сборки, так и настроены пользователем).

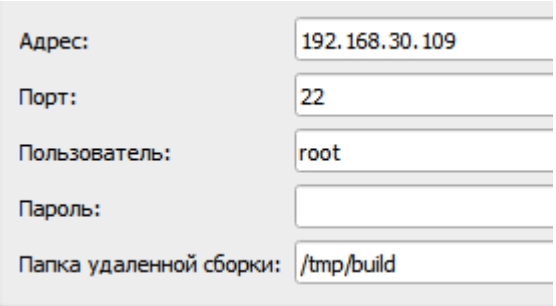


Рисунок 39 – Настройки удаленной компиляции проекта

Для всех типов локальной сборки необходимо, чтобы в системе присутствовали соответствующие компиляторы и набор заголовочных файлов и библиотек (toolchain). Этот набор должен иметь актуальные версии, полностью соответствующие версиям на применяемом целевом устройстве. Структура каталога должна иметь вид, соответствующий рисунку (см. Рисунок 40).

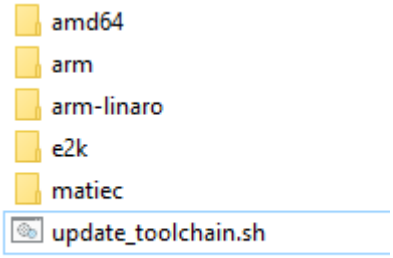


Рисунок 40 - Структура каталога toolchains

Существует несколько сценариев применения среды разработки для сборки проектов под различные платформы.

1) Среда разработки запускается на ОС Windows 10, сборка проектов возможна под платформы Эльбрус, ARM, x86. В этом случае необходимо, чтобы в ОС была активирована подсистема WSL. Также требуется размещение кросс-компиляторов для упомянутых платформ. Кросс-компиляторы размещаются, как правило, в каталоге /opt подсистемы WSL. Набор библиотек (toolchain) должен быть размещен IDE/BUILD/toolchains.

2) Среда разработки запускается на ОС Linux (или на виртуальной машине с ОС Linux). Сборка проектов возможна под платформы Эльбрус, ARM, x86. В случае установленной системы исполнения возможен также запуск проектов. Необходимо, чтобы в ОС присутствовали кросс-компиляторы, а такжеь нативный компилятор,

соответствующей платформе ОС. Кросс-компиляторы размещаются, как правило, в каталоге /opt. Набор библиотек (toolchain) должен быть размещен IDE/BUILD/toolchains.

Внимание! При обновлении библиотек toolchain необходимо их также обновлять и на целевом устройстве. Несоответствие библиотек гарантировано приведет к неработоспособному состоянию целевого устройства.

7.8. Управление целевым устройством

Средства взаимодействия ПЛК вынесены на панель «панель инструментов ПЛК» (см. Рисунок 41) в верхнем меню программы.

Рисунок 41 - Панель инструментов

Кнопки панели сборки позволяют управлять процессом сборки, а также передачей проекта на целевое устройство и его исполнение в таблице (см. Таблица 3).

Таблица 3 - Кнопки панели управления ПЛК

Внешний вид кнопки	**Наименование**	**Функция**
	Сборка проекта	Выполняется сборка исполняемого файла проекта для выбранной целевой платформы.
	Очистка проекта	Выполняется очистка объектных файлов предыдущих сборок.
	Просмотр ST кода	Открывает окно для просмотра сгенерированного полного ST-кода программы.
	Запуск статического анализатора	Запускает встроенный статический анализатор кода на языке ST.
	Подключение	Выполняет подключение к основному или резервному процессорному модулю ПЛК.
	Отключение	Отключается от основного или резервного процессорного модуля ПЛК.
	Передача программы	Выполняется передача новой программы на ПЛК.
	Запуск	Запуск на исполнения программы.
	Стоп	Остановка исполнения программы.

	Генерация .ccb	Генерация системой исполнения файла оборудования для последующего создания лицензии
	Передача .lic	Загрузка файла лицензии на контроллер
	Сервис управления Runtime Manager	Сервис, осуществляющий мониторинг, обновление и управление исполнительным процессом.

7.8.1. Сборка проекта

Перед исполнением проекта на целевом устройстве необходимо выполнить его сборку (компиляцию).

Сборка проекта запускается кликом по элементу (Пункт *Сборка проекта* в панели инструментов) в панели инструментов

При этом выполняется несколько этапов сборки:

- генерация общего ST кода всей программы;
- интерпретация ST кода и генерация набора файлов на языке C;
- компиляция программы на языке C в исполняемый файл – динамическую библиотеку (shared library);
- динамическая библиотека должна быть собрана кросс-компилятором, предназначенным для выбранной платформы ПЛК.

Полученный промежуточный код на языке ST полностью доступен для просмотра пользователем (Пункт *Просмотр ST кода* в панели инструментов) и может быть отображен с помощью соответствующей кнопки на панели инструментов. **Внимание!** Этот код автоматически сгенерирован. Изменения в коде будут перезаписаны при следующем запуске компилятора.

Результат компиляции и сообщения об ошибках выводятся в консоль (см. Рисунок 42).

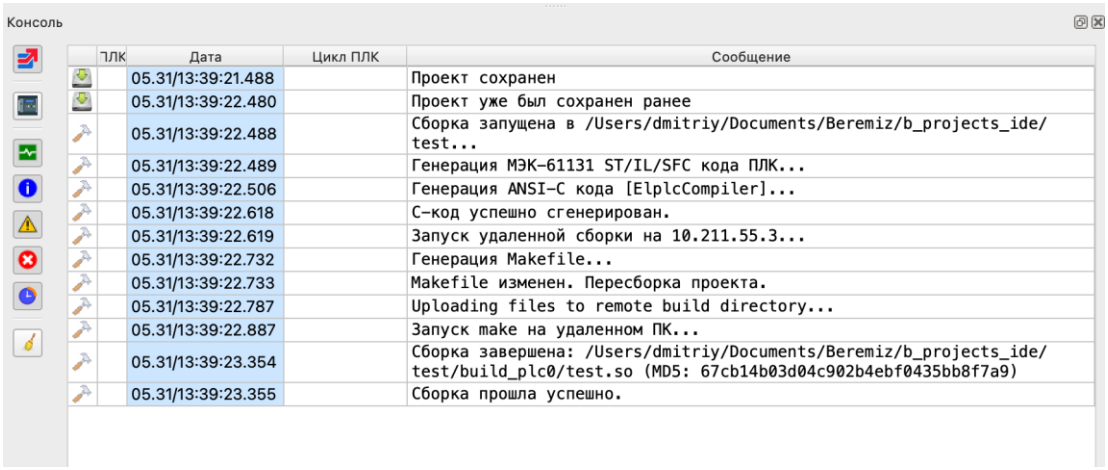


Рисунок 42 - Консоль вывода процесса компиляции

7.8.1. Очистка проекта

Очистку проекта рекомендуется проводить в случае сборки проекта после обновления IDE или фалов тулчейна. После очистки, следующая сборка может занять длительное время.

Во всех остальных случаях, данное действие не требуется. Система сборки автоматически определит измененные файлы.

7.8.2. Запуск проекта на ПЛК

После успешной сборки проекта исполняемый файл необходимо передать на целевое устройство. Для этого необходимо воспользоваться панелью управления ПЛК и кнопками, представленными в (Таблица 3).

Выполняется подключение к ПЛК кнопкой «Подключение». После установки соединения система определит состояние программы ПЛК. Если она находится в режиме исполнения, то будет активна кнопка «Стоп». В противном случае будет активна кнопка «Запуск».

Во время операции передачи новой программы на целевое устройство используется следующее окно (см. Рисунок 43). Окно разделено на три раздела, отображающие следующие данные:

- общую информацию об устройстве и проекте;
- изменения программы, относительно исполняющейся на контроллере;
- энергонезависимые переменные (retain).

Рисунок 43 - Окно подключения к целевому устройству

В данном окне отображается диагностическая информация о текущем проекте в системе разработки и проекте, загруженном на контроллер. Доступны метаданные проекта в системе разработки и на целевом устройстве. Также отображается информация о версии исполнительной программы.

Второй раздел (см. Рисунок 44) содержит информацию об изменениях относительно проекта на контроллере. При сохранении и сборке проекта рассчитываются контрольные суммы всех элементов программы (программы, переменные программ, пользовательские типы данных, модули и расширения), далее эта информация сохраняется в исполнительный файл, загружаемый на целевое устройство. Согласно данной таблице, осуществляется принятие решение о возможности и типе обновления программы. В графе «md5 модулей» перед именем модуля/ системы расширения отображается признак возможности выполнения «быстрой» онлайн замены.

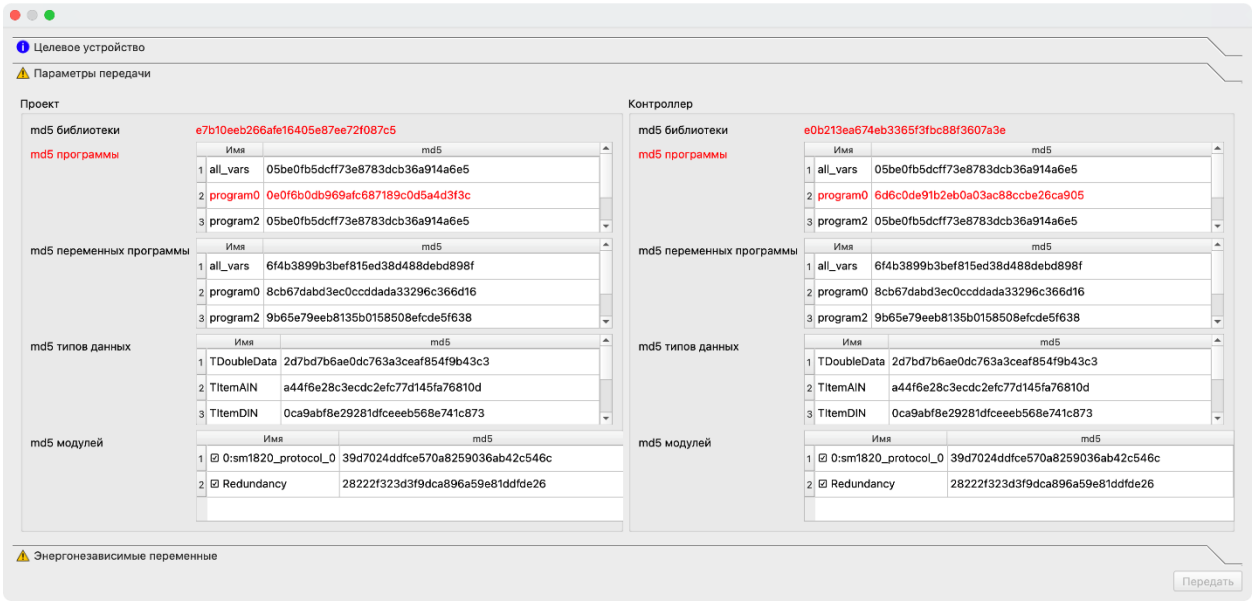


Рисунок 44- Окно отображения локальных изменений в проекте

Для примера был изменен программный код программы program0. Это отображается в строке md5 данной программы и в общем статусе (“md5 программы” выделено красным), также изменилась контрольная сумма программы.

Последние версии IDE поддерживают несколько режимов обновления программы на целевом устройстве, оффлайн и онлайн замены программы. При оффлайн замене выполняется полная остановка исполнительного алгоритма, загрузка новой программы и запуск. При онлайн замене остановка программы производиться не будет, новый проект будет запущен на следующем цикле контроллера с сохранением данных.

Онлайн замена невозможна при активной системе резервирования. Запрещено безударное обновление программы на резервном ПЛК с запущенной программой, а также на основном, если система резервирования находится в активном состоянии. Для выполнения онлайн замены на резервированном ПЛК необходимо выполнить следующие действия, идентичные оффлайн замене:

- остановить standby - модуль;
- обновить основной модуль системы одним из режимов;
- выполнить оффлайн обновление и запуск резервного модуля.

7.8.2.1. Offline обновление программы

После остановки исполнения программы необходимо передать новый исполняемый файл на целевое устройство (кнопка «Передача программы»), а после этого запустить обновленную программу на исполнение кнопкой «Запуск».

После перезапуска программа будет исполняться исходя из значений переменных по умолчанию, за исключением тех переменных, которые определены как retain.

7.8.2.2. Online обновление программы

Необходимость замены управляющей программы в безударном режиме реализовано с использованием механизма “online” замены. В данном типе передачи программы на целевое устройство алгоритм не останавливается, а на следующем цикле контроллера продолжает выполнение с сохранением значений всех переменных, подключений и состояний устройств.

Реализованы два сценария выполнения онлайн замены программы на целевом устройстве:

- быстрая замена;
- замена с переинициализацией плагинов, систем расширения и переменных.

«Быстрая» онлайн замены выполняется без переинициализации ресурсов, поэтому при следующих изменениях в проекте не может быть использована:

- изменен один из плагинов (например modbus_tcp_master);
- изменена одна из систем расширения (например система архивирования);
- добавлены, удалены или изменены переменные в программах, функциях или функциональных блоках;
- добавлены, удалены или изменены пользовательские типы данных;
- изменены retain-переменные (удален модификатор для существующих переменных или новая переменная помечена таким модификатором);
- изменены настройки системы резервирования.

Алгоритм выполнения безударной замены представлен на (см. Рисунок 45).

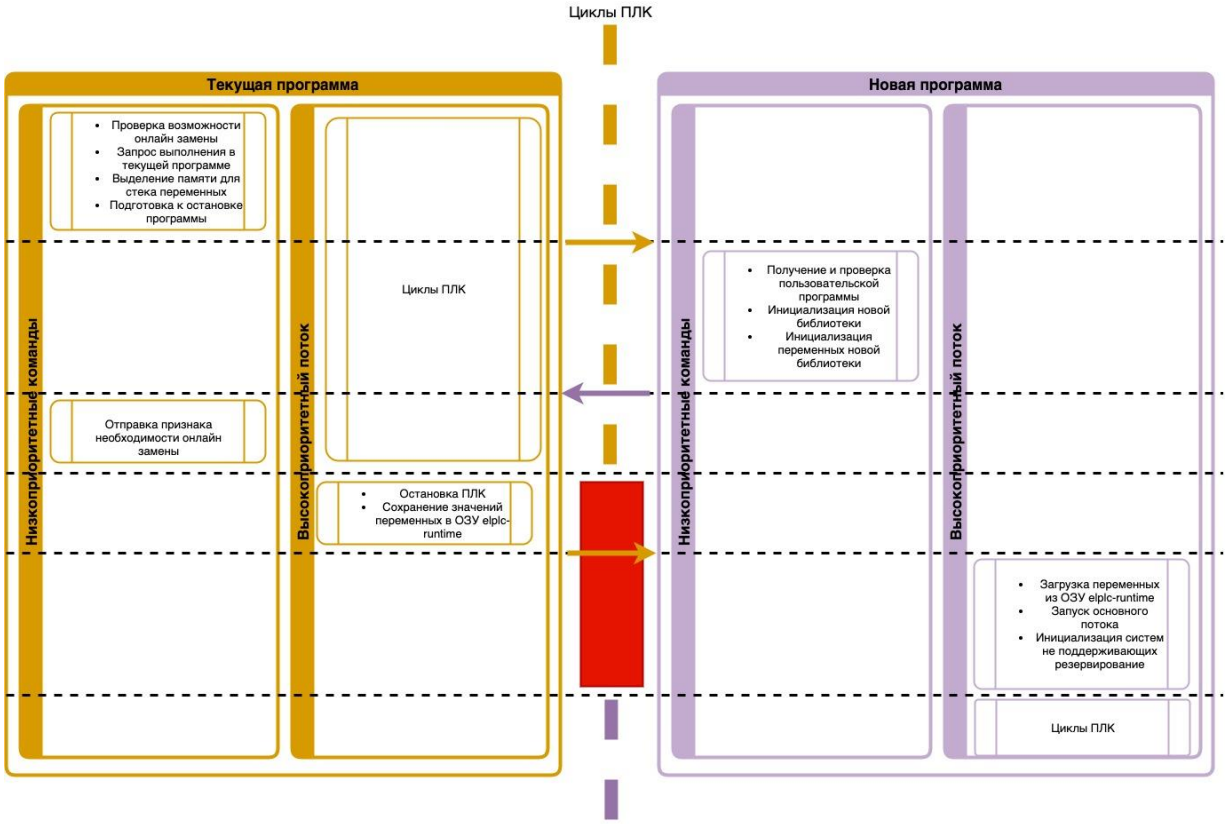


Рисунок 45 - Алгоритм безударной замены программы

Выполнение операции замены программы в онлайн режиме преимущественно выполняется в низкоприоритетных потоках текущего и нового процесса.

При получении команды на выполнение безударной замены в запущенную программу отправляется запрос на подтверждение возможности выполнения, запускается подготовка к выполнению. Механизм низкоприоритетных потоков программ предназначен для снижения влияния процедуры на работающий процесс и уменьшения времени «простоя» между циклами прошлой и новой программ.

В адресном пространстве службы elplc-runtime размечается область памяти для переменных программ и ФБ, записывается служебная информация. Инициализируется процесс онлайн замены в плагинах и системах расширения. Ставится статус службы «ПЛК выполняет операцию онлайн замены». Данный статус будет изменен позже либо при удачном выполнении операции на «ПЛК запущен» либо придет сообщение «Ошибка загрузки программы на ПЛК».

Создается новая программа и открывается обработчик разделяемой библиотеки системой исполнения. В отдельном потоке новой программы создается контекст

программы (ElplcCompiler), инициализируются переменные в структуре, используемые системой архивирования, резервирования, отладчика и т.д. В случае выполнения онлайн замены с переинициализацией происходит поиск переменных новой программы в сохраненном массиве значений переменных прошлой программы.

Операция подготовки выполняется в фоне и может занимать некоторое количество циклов ПЛК. Во время выполнения операции отключается возможность просматривать переменные в отладчике, устанавливать новые значения, управлять состоянием программы из IDE.

После завершения работы низкоприоритетных потоков текущей и новой программ запускается непосредственно подмена программ в исполнительной программе. При этом в текущем процессе останавливается основной поток программы, завершается работа плагинов и сетевых подключений, не поддерживающие систему быстрой онлайн замены, копируются значения переменных и ФБ в ранее размеченную область памяти elplc-runtime. Завершается работа обработчика текущей исполнительной программы (.so).

Запускается высокоприоритетный поток новой программы, создается таймер на основной цикл программы. Программа продолжает свою работы с данными, загруженными в действиях выше.

7.8.3. Установка лицензии целевого устройства

В процессе запуска на исполнение программы может быть выдано сообщение об отсутствии или несоответствии лицензии целевого устройства. Система исполнения ELPLC-RUNTIME лицензируется на исполнительное устройство. Проверка лицензии осуществляется в процессе запуска проекта на исполнение. В случае истечения срока действия лицензии (если она срочная) в процессе работы системы исполнения остановки исполнения программы не произойдет, однако после остановки повторный запуск будет невозможен до обновления лицензии.

Генерация и установка лицензии возможна через среду разработки. Для этого запустите среду разработки, откройте любой проект и подключитесь к системе исполнения, установленной на виртуальной машине, и нажмите кнопку сбора информации об устройстве (см. Рисунок 46).

Рисунок 46 - Кнопка чтения информации об устройстве

Перешлите полученный файл поставщику программного обеспечения для генерации персональной лицензии. В ответ будет получен файл лицензии. Положите его на хост-машине в папку обмена. Далее воспользуйтесь функцией загрузки лицензии на устройство (см. Рисунок 47).

Рисунок 47 - Кнопка загрузки лицензии

После установки новой лицензии система готова к работе.

7.9. Запуск проекта на резервированном ПЛК

В случае работы с резервированным ПЛК важно соблюдать идентичность проектов на процессорных модулях. Система резервирования может корректно функционировать только в случае полного совпадения исполняемых проектов на двух процессорных устройствах. Таким образом, в случае обновления программы на резервированном ПЛК необходимо выполнить следующие действия:

- остановить исполнение проекта на обоих процессорных модулях;
- обновить программу на основном процессорном модуле и запустить ее на исполнение;
- обновить программу на резервном процессорном модуле и запустить ее на исполнение.

7.10. Управление программой аппаратным ключом.

Некоторые исполнительные процессорные устройства, выполняющие программы ELPLC-LOGIC, могут быть оснащены аппаратным ключом управления. Ключ применяется для блокировки возможности замены программы пользователем, а также для принудительной отмены автоматического запуска программы в случае, если разработанная программа приводит устройство в неработоспособное состояние.

Аппаратные ключи имеют, как минимум, два положения: RUN и CONF. В некоторых вариантах три положения: RUN, CONF, RUN-CONF.

В положении RUN будет производиться автоматический запуск программы ПЛК после включения питания. При этом пользователь не сможет воздействовать на программу из среды разработки. Будет заблокирована функция остановки программы и замены исполняемого файла. Не будет работать режим отладки и установки значений переменных (force).

В положении CONF подключение среды разработки будет разрешено. При этом, при перезагрузке или после подачи питания на процессорный модуль программа не будет переведена в режим запуска. Система исполнения будет находиться в режиме ожидания команд среды разработки. При этом, если перевести ключ из положения RUN в положение CONF на работающем ПЛК, то исполнение программы не будет остановлено.

В положении RUN-CONF (поддерживается некоторыми процессорными модулями) программа после запуска ПЛК будет автоматически стартовать, но будет также возможность воздействия на нее из среды разработки.

Внимание! В целях повышения уровня информационной безопасности рекомендуется рабочие ПЛК оставлять с ключами в положении RUN.

7.11. Сервис управления Runtime Manager.

Сервис представляет собой службу, работающую в фоновом режиме на целевом устройстве, осуществляющий мониторинг и конфигурацию исполнительной программы.

Окно сервиса разделено на 3 группы:

- Информация о сервисе (Рисунок 48)

  В левой части окна отображается статус подключения к системному сервису (Подключено) и статус работы исполнительной программы runtime (Запущен)

  Сверху находятся две кнопки, отвечающие за остановку и запуск исполнительной программы. При нажатии на (STOP) программа, работающая на контроллере может быть аварийно завершена.

  В левой части располагается UDP-терминал, отображающий сообщения, выдаваемые системой исполнения. В режиме отладки также будут показываться сообщения в данном окне.

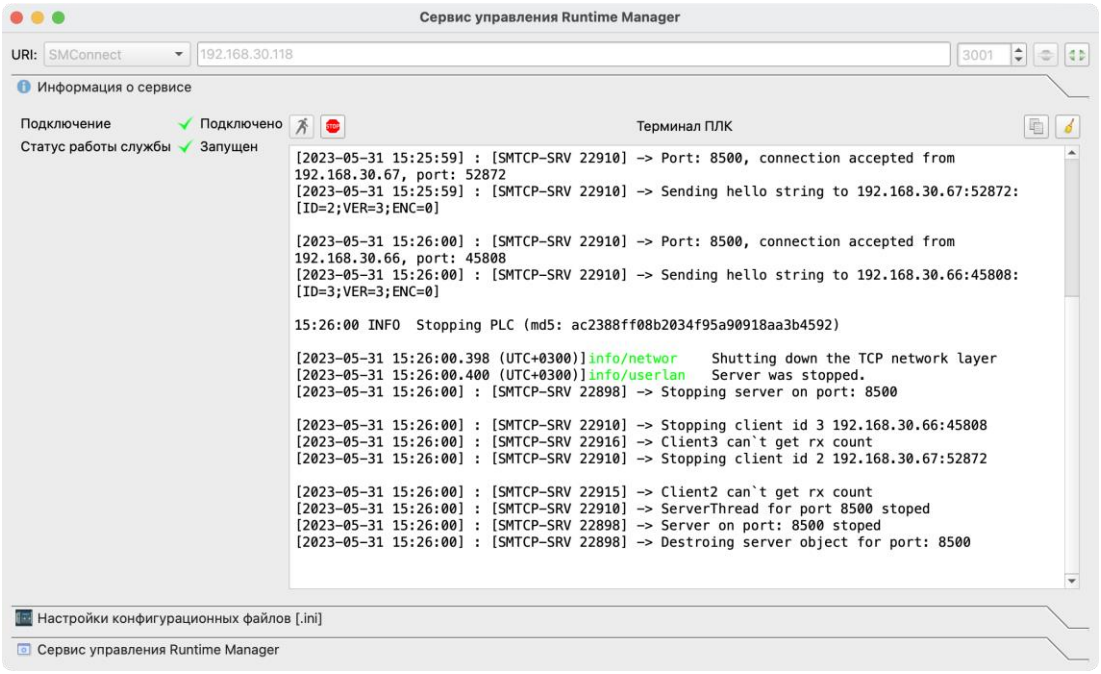


Рисунок 48 - Информация о сервисе

- Настройки конфигурационных файлов

Конфигурационные файлы служб. Описание в документации на elplc-runtime и elplc-runtime-service

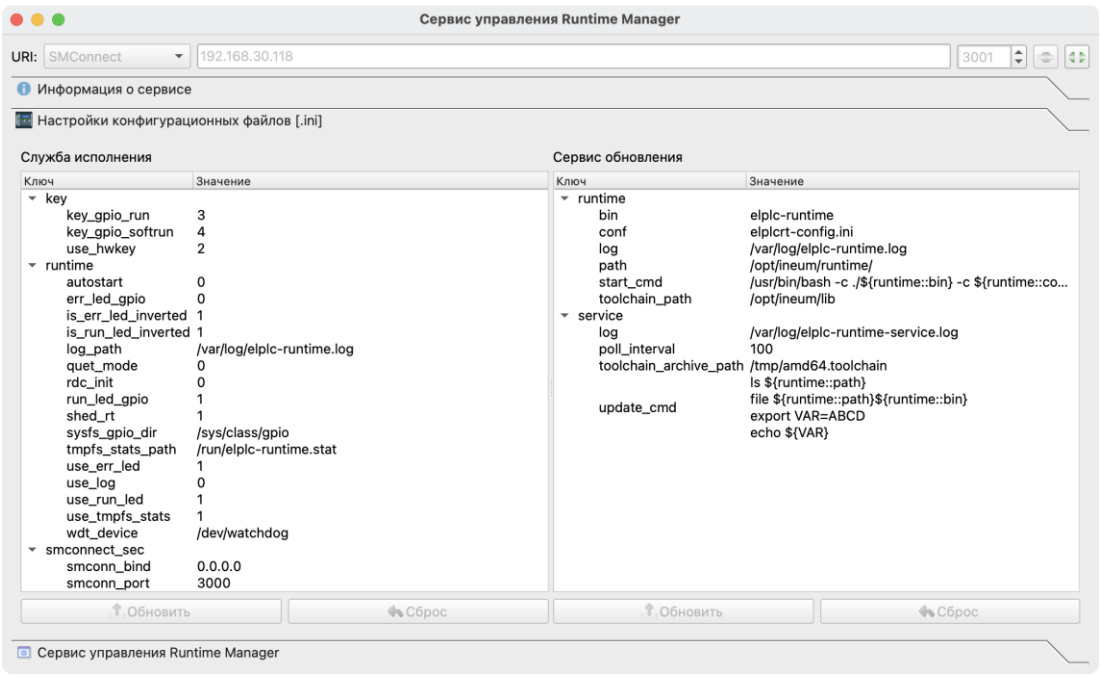


Рисунок 49 - Настройки конфигурационных файлов

- Сервис управления

  Окно обновления исполнительной программы и библиотек на целевом устройстве.

  При нажатии на кнопку «Выберите файл» в стандартном диалоговом окне операционной системы необходимо указать файл к запакованному файлу обновления системы. Например amd64.toolchain для устройства на архитектуре X86. После этого необходимо выбрать кнопку правее «Передать файл и обновить тулчейн». При этом файл будет сохранен на контроллере и выполнен код из конфигурации сервиса.

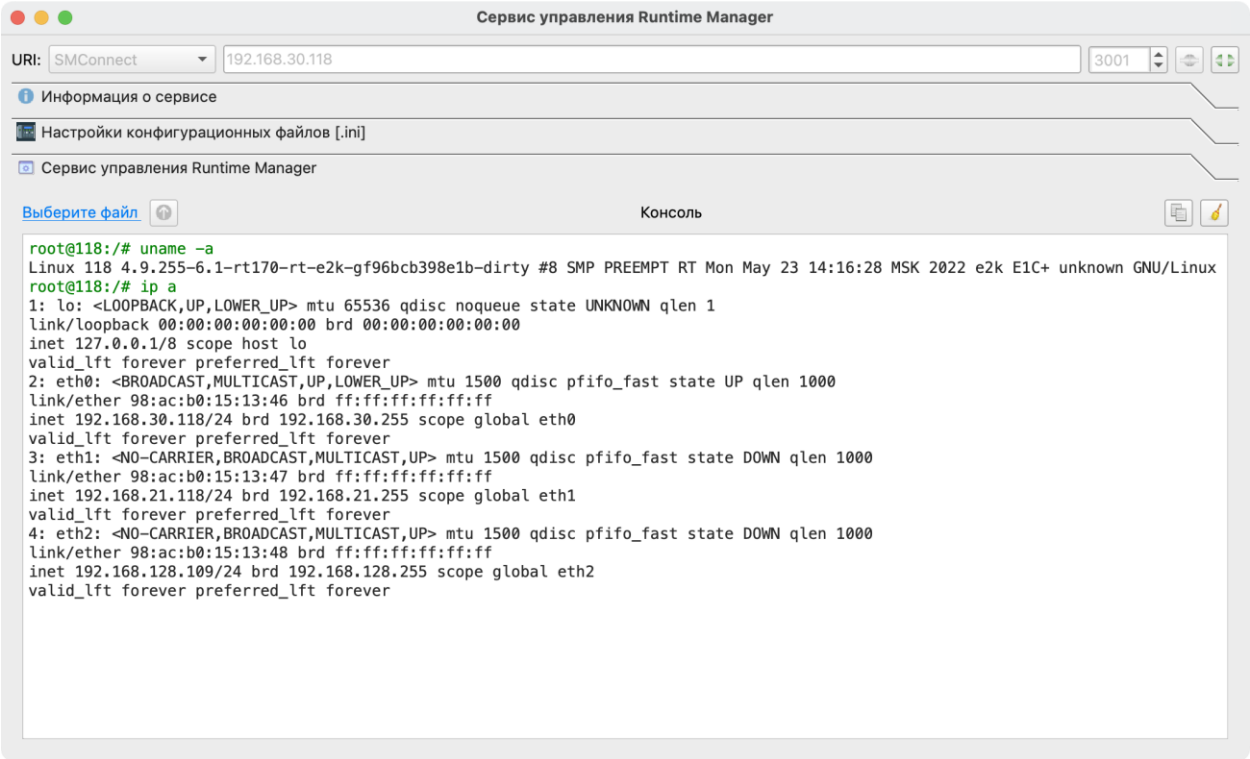


Рисунок 50 - Сервис управления

8. СИСТЕМЫ, ДРАЙВЕРЫ, ПРОТОКОЛЫ

САПР ELPLC-LOGIC имеет в своем составе набор драйверов устройств, поддержку ряда протоколов передачи данных и дополнительные системы, обеспечивающие полноценное функционирование ПЛК для решения задач контроля и управления.

8.1. Система резервирования

Исполнительная система ELPLC-LOGIC имеет функцию резервирования в конфигурациях целевых устройств (ПЛК) с двумя процессорными модулями или при дублировании ПЛК. Система резервирования обеспечивает синхронизацию блоков памяти, содержащих значения переменных систем исполнения на каждом такте ПЛК.

8.1.1. Аппаратное обеспечение

Система резервирования может работать на контроллерах с различными процессорными модулями целевых устройств (ПЛК) (например MP17), под управлением операционной системы на базе ядра Linux.

Для работы необходимо не менее двух процессорных модулей, удовлетворяющих требованием приведенных в разделе 1 данного руководства.

Процессорные модули целевых устройств (ПЛК) для реализации функций резервирования используют информационный канал через сеть Ethernet.

Примечание. Для обеспечения достаточной пропускной способности порты ethernet процессорных модулей должны иметь скорость не ниже 100Мбит/с. Рекомендуется выделять отдельный Ethernet интерфейс для прямого соединения резервированных процессорных модулей. Не рекомендуется использовать в канале резервирования сетевые коммутаторы. Не рекомендуется организовывать канал резервирования через сети, где идет прочий обмен данными.

С целью повышения достоверности диагностики отказа процессорного модуля в системе резервирования предусматривается дополнительный последовательный канал обмена данными (SafetyLink). В ПЛК на базе системного интерфейса ELPLC-BUS такой канал предусмотрен через объединительную панель. SafetyLink позволяет, в случае получения ошибки связи на основном канале резервирования, получить дополнительную

информацию о состоянии процессорного модуля-партнера, с целью недопущения возникновения в системе двух управляющих модулей.

8.1.2. Программное обеспечение

Система резервирования интегрирована в ПО ELPLC-LOGIC и позволяет конфигурировать и отслеживать текущее состояние системы.

Доступны диалоговые окна для настройки запускаемого процесса на контроллере и МЭК-переменные, добавляемые в программах проекта.

8.1.3. Архитектура системы резервирования

8.1.3.1. Режим 1oo2

Система резервирования предусматривает работу в режиме «горячего резерва» по схеме 1oo2 («1 out of 2», «один из двух»). Этот режим подразумевает наличие в составе целевого устройства (ПЛК) двух процессорных модулей (удовлетворяющие требования приведенные в разделе 1 данного руководства). При этом одно устройство имеет роль «основного», а второе «резервного». Оба процессорных модуля, помимо интеграции в ЛВС технологического процесса, должны иметь непосредственный выделенный канал связи между собой. Процессорные модули целевых устройств должны иметь в составе три канала Ethernet 10/100/1000 Мбит/с. Наличие трех каналов позволяет:

- от каждого процессорного модуля использовать два канала для резервированного (в «горячем» резерве) информационного обмена с устройствами или системами «верхнего уровня», такими как серверы SCADA-систем, рабочие места оператора и т.д;
- третий канал должен быть использован для информационного обмена между процессорными модулями для организации системы резервирования.

Описание выполнения системы резервирования

Каждый процессорный модуль получает в момент старта выполнения прикладной программы одну из двух ролей: PRIMARY («основного») и STANDBY («резервного»). Роль может быть задана соответствующим параметрам конфигурационного файла системы исполнения или определяется аппаратно. Шина ELPLC-BUS имеет соответствующий контакт на объединительной панели, позволяющий программному обеспечению определить место установки процессорного модуля. Так, процессорный слот

«слева» предназначен для работы «основного», а «справа» для «резервного». Таким образом, указание роли для ПЛК на базе шины ELPLC-BUS не требуется, определение роли будет осуществлено автоматически.

Процессорный модуль целевого устройства с ролью PRIMARY выполняет всю работу, связанную с обменом данными с модулями УСО и обслуживанием коммуникационных протоколов. Процессорный модуль целевого устройства с ролью STANDBY лишь получает на каждом цикле выполнения прикладной программы данные с текущими значениями переменных для синхронизации. В случае потери связи между двумя процессорными устройствами по какой-либо причине, устройство с ролью STANDBY начинает выполнять роль PRIMARY, проверяя на каждом цикле подключение основного устройства.

Примечания:

1. При восстановлении связи между процессорными модулями целевого устройства (ПЛК) роли в системе резервирования возвращаются в первоначальное состояние.
2. Процессорный модуль с ролью STANDBY не сможет перейти в роль PRIMARY, если не была выполнена успешная синхронизация по крайней мере один раз. Таким образом, запуск ПЛК без процессорного модуля в слоте PRIMARY невозможен.

8.1.4. Настройка системы резервирования

Если целевое устройство удовлетворяет требованиям к системе резервирования (см пункт Аппаратное обеспечение), то в настройках проекта необходимо настроить дополнительные параметры на вкладке «Настройки сборки проекта и соединения с целевым устройством (ПЛК)».

Справа на вкладке располагаются настройки системы резервирования (см. Рисунок 51).

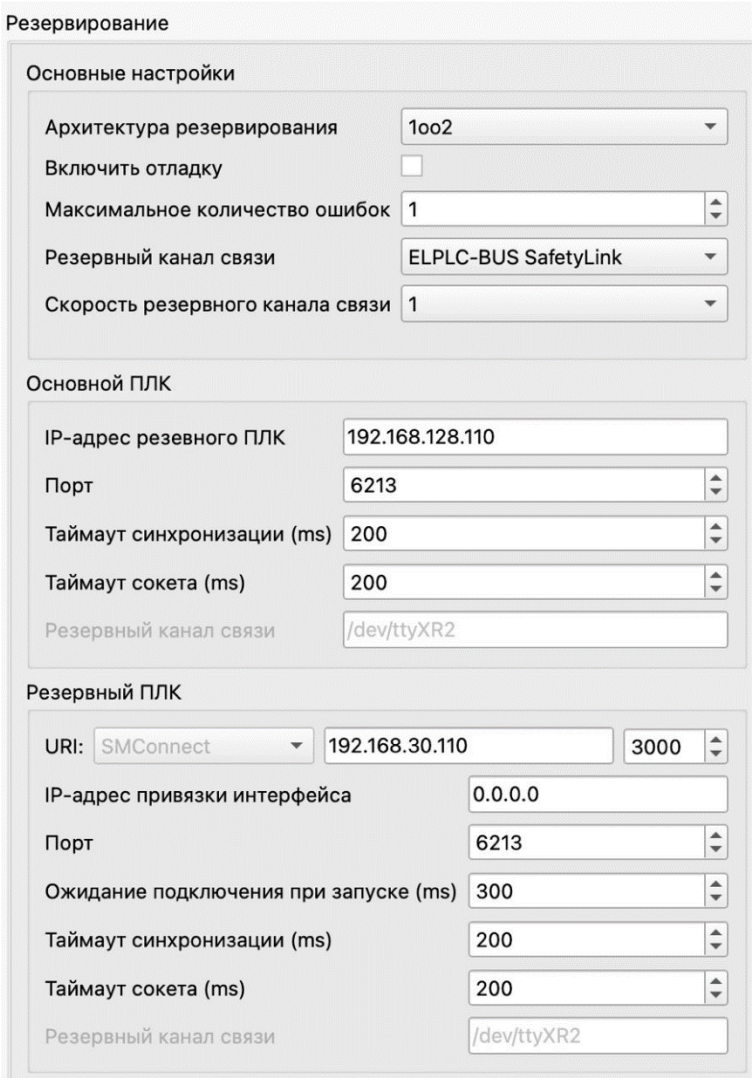


Рисунок 51 - Настройки системы резервирования

Далее рассмотрим подробно все параметры настройки системы резервирования.

8.1.4.1. Группа параметров «Основные настройки»:

- «Архитектура резервирования» – имеет параметры None, 1oo2 и 2oo2:

а) None – система резервирования отключена;

б) 1oo2 – система резервирования целевого устройства (ПЛК) с двумя процессорными модулями в режиме «горячего резерва» по схеме 1oo2 («1 out of 2», «один из двух»).

в) 2oo2 – система резервирования целевого устройства (ПЛК) с двумя процессорными модулями в режиме «горячего резерва» по схеме 2oo2 («2 out of 2», «два из двух»).

Примечания:

1. 2oo2 архитектура в планах реализации, в текущей версии доступна только 1oo2.

- «Включить отладку» – вывод в консоль диагностической информации о работе системы резервирования (просмотр вывода возможен в случае ручного запуска системы исполнения ПЛК на целевом устройстве).

2. данный параметр должен быть снят после окончания пусконаладочных работ целевого устройства (ПЛК).

- «Максимальное количество ошибок» - количество ошибок связи на канале резервирования, после которого будет принято решение об отказе «партнера».

- «Резервный канал связи» – параметр включения/выключает работы системы резервирования по дополнительному каналу (SafetyLink) (возможны значения: нет, ELPLC-BUS SafetyLink, Serial safety link);

- «Скорость резервного канала» – параметр, отвечающий за настройку скорости обмена данными по каналу SafetyLink, в Мбит/с;

8.1.4.2. Группа параметров «Основной ПЛК»:

Группа параметров «Основной ПЛК» отвечает за настройку процессорного модуля целевого устройства (ПЛК) имеющего роль PRIMARY («основной»):

- «IP-адрес резервного ПЛК» – адрес процессорного модуля с ролью STANDBY («резервный»);

- порт – tcp/ip порт для связи с резервным ПЛК;

- таймаут синхронизации – время ожидания партнера в регулярном цикле. В течении этого времени должна выполниться сессия синхронизации. Если синхронизация за указанное время не выполняется, то связь считается утерянной, что означает отказ партнера;

- таймаут сокета – таймаут выполнения команд чтения/ записи;

- резервный канал связи – идентификатор устройства, в случае выбора резервного канала связи «Serial safety link».

8.1.4.3. Резервный ПЛК:

- URI: IP – адрес для подключения к процессу elplc-logic;

- URI: Порт – порт для подключения к процессу elplc-logic;
- IP-адрес привязки интерфейса – локальный IP адрес интерфейса ПЛК, к которому будет привязан сервер системы резервирования. Для привязки ко всем интерфейсам необходимо указать адрес 0.0.0.0 (по умолчанию);
- порт – tcp/ip порт для связи с основным ПЛК;
- ожидание подключения при запуске – время в миллисекундах, в течении которого ожидается подключение основного ПЛК при запуске контроллера. В процессе этого ожидания ПЛК находится на этапе init и программа не выполняется. Функция блокируется на время, заданное в этом параметре;
- таймаут синхронизации – время ожидания партнера в регулярном цикле. В течении этого времени проверяется факт подключения партнера, если его нет или ожидается обновление данных. Если синхронизация за указанное время не выполняется, то STANDBY устройство проверяем отклик основного через Safety link и, в случае отсутствия ответа, переходит в режим PRIMARY;
- таймаут сокета – таймаут выполнения команд чтения/ записи;
- резервный канал связи – идентификатор устройства, в случае выбора резервного канала связи «Serial safety link».

8.1.5. Взаимодействие с системой резервирования из проекта

После добавления системы резервирования в проект доступны переменные для диагностики ее работы, которые необходимо добавить в окне внешних переменных проекта.

 -- добавление переменных.

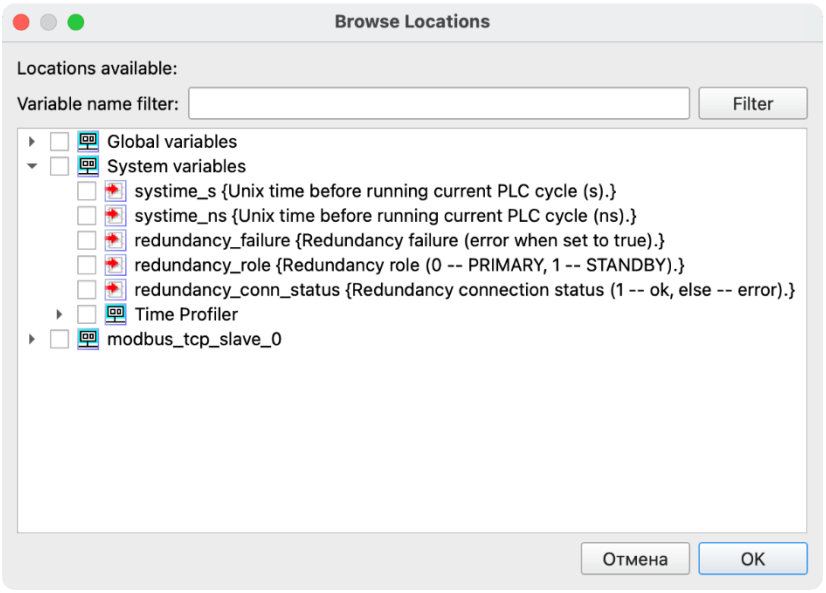


Рисунок 52 - Добавление переменных

На рисунке (см. Рисунок 43) представлено окно добавления переменных системы резервирования, доступны следующие поля:

- redundancy_failure – статус работы службы резервирования, BOOL, false - нет ошибок, true – резервирование нарушено;
- redundancy_role – текущая роль процессорного модуля, 0 – основной, 1 – резервный;
- redundancy_conn_status – статус подключения к партнеру резервирования (1 – ошибок связи не обнаружено, другое – код ошибки).
-

8.2. Система архивирования

Система исполнения ELPLC-LOGIC имеет в своем составе модуль архивирования, предназначенный для сохранения истории изменения переменных процесса на целевых устройствах (ПЛК). САПР позволяет определить набор архивов с определением правил архивирования для каждого архива индивидуально:

Поддерживаются следующие стратегии:

- непрерывное архивирование на жесткий диск с ограничением объема;
- аварийное архивирование с накоплением данных в ОЗУ и сохранением на диск по триггеру.

Формат файла сохранения: csv или bin.

Тактирование работы системы архивирования выполняется общим циклом ПЛК на стадии publish.

Система архивирования реализует следующие функции:

- накопление данных в кольцевом архиве по заданному периоду по времени или по объему;
- сохранение данных аварийного архива в ОЗУ на жесткий диск по триггеру с выбором фронта;
- настройка параметров архива через IEC переменные;
- присвоение астрономических меток времени, а также порядковый номер записи;
- доступ к списку архивов.

8.2.1. Работа системы архивирования

Подсистема архивирования выполняет архивирование переменных исполнительной системы ELPLC-LOGIC с присвоением меток времени и кодов качества.

Переменные, подлежащие архивированию, группируются в архивы. Настройка стратегии архивирования выполняется для архива. Все переменные, входящие в архив, имеют единые настройки архивирования. Также для архива выбирается размер поля данных тега архивирования.

Определяются следующие стратегии архивирования:

- периодическое непрерывное циклическое архивирование на диск;
- архивирование по триггеру.

Непрерывное архивирование выполняется на жесткий диск, в один из выбранных пользователем формат файла: бинарный файл определенной структуры, csv-файл, SQLite БД. В случае с БД - на каждую переменную создается отдельная таблица. Все данные при архивировании в БД приводятся к размеру, указанному в настройках архива. Архив также содержит информацию о типе данных (целочисленном или вещественном) каждой переменной.

Таблица архивирования содержит следующие поля:

- метка времени;
- значение.

Архивирование по триггеру подразумевает хранение определенного пользователем объема данных (объем задается в количестве срезов, соответствующих количеству циклов исполнительной системы) в ОЗУ. Также осуществляется отслеживание значения триггерной переменной (IEC переменная типа byte). В случае, если триггерная переменная принимает значение, отличное от 0, накопленный объем данных в ОЗУ записывается в файл на диск, в соответствии с выбранным для архива типом данных. Триггерная переменная сбрасывается системой архивирования. Записанный файл, в случае bin или csv файла, помещается в определенный пользователем каталог на диске и имеет имя, содержащее дату и время момента записи (срабатывания триггера).

Архивы, при записи в файл любого типа, записываются построчно на каждый такт архивирования. Сохраняется информация об абсолютной дате и времени, номере цикла ПЛК, а также перечне значений параметров на указанную метку времени. При этом все данные приводятся к типу данных FLOAT или DOUBLE, в зависимости от выбранного параметра в настройке архива.

Диаграмма работы системы архивирования по триггеру приведена на рисунке (см. Рисунок 53).

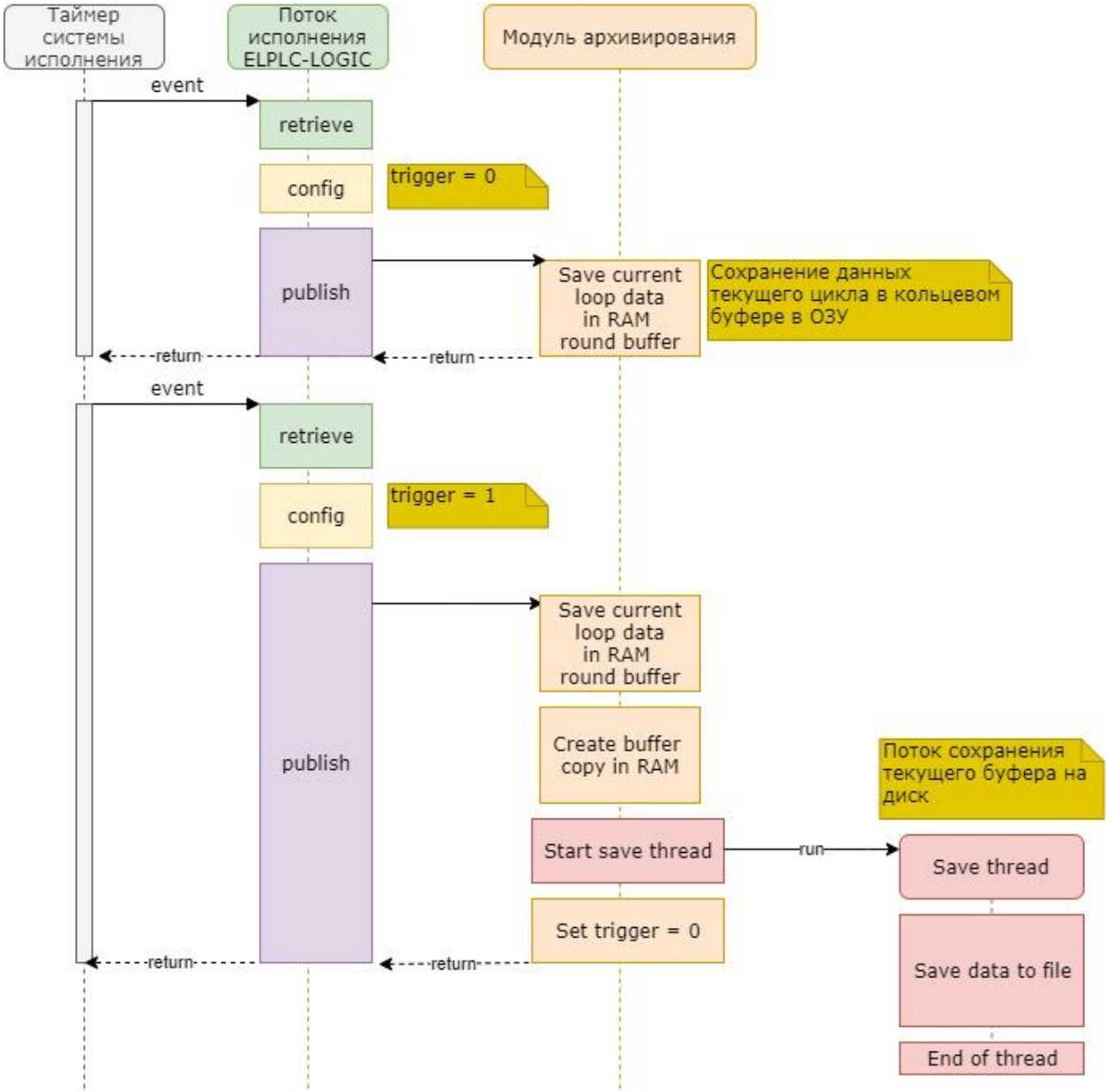


Рисунок 53 - Архивирование по триггеру

На представленной диаграмме видно, что сохранение переменных, определенных для архивирования, осуществляется на каждом цикле ПЛК в кольцевой буфер системы архивирования. В случае, если логическая часть пользовательской программы сформировала активное состояние триггерной переменной (trigger = 1), то на этом цикле, после сохранения данных текущего цикла в архиве, будет выполнено копирование кольцевого архива и запущен отдельный, низкоприоритетный поток, который выполнит отложенную запись скопированного архива на диск. При этом система архивирования сама сбросит триггерную переменную в 0 и вернет управление основной программе.

Таким образом, основной высокоприоритетный цикл ПЛК не будет задержан на время записи данных на диск.

8.2.2. Структура бинарного файла данных архива

Система архивирования формирует бинарные файлы архивов на диске. Эти файлы имеют формат, представленный на рисунке (см. Рисунок 54).

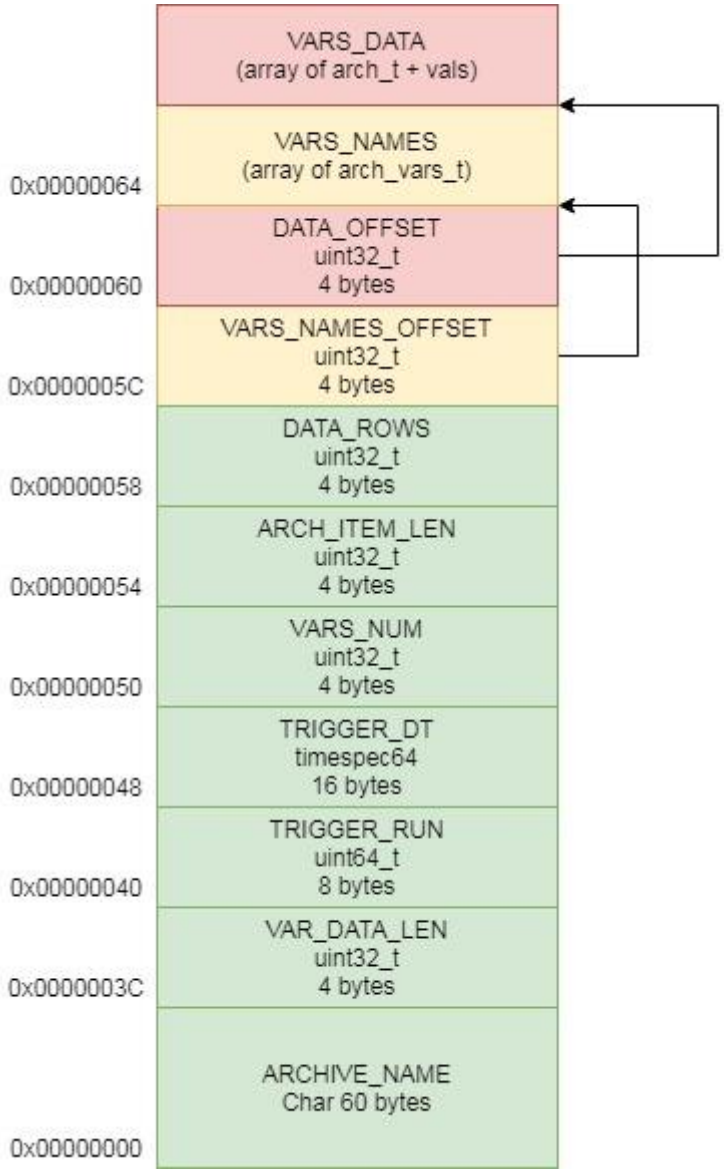


Рисунок 54 - Структура двоичного файла архива

Файлы сохраняются на жесткий диск ПЛК и доступны для последующего анализа в среде разработки ELPLC-LOGIC или в стороннем программном обеспечении.

Заголовок архива при этом имеет следующие поля:

```
typedef struct {
char name[VAR_NAME_MAX_LENGTH];
uint32_t variable_data_len;
uint64_t trigger_run_number;
uint8_t trigger_dt[16];
uint32_t variables_number;
uint32_t item_len;
uint32_t data_rows;
uint32_t variables_names_offset;
uint32_t data_offset;
} archive_trigger_header_t;
```

Поля структуры:

- **name** - текстовое имя архива (60 символов);
- **variable_data_len** - длина данных каждой переменной (может быть 4 или 8 байт);
- **trigger_run_number** - номер цикла ПЛК, на котором произошло срабатывание триггерной переменной (8 байт);
- **trigger_dt** - отметка времени, на которой произошло срабатывание триггерной переменной (время struct timespec64, 16 байт);
- **variables_number** - количество переменных в архиве (4 байта);
- **item_len** - длина каждой строки данных (4 байта);
- **data_rows** - количество строк данных (4 байта);
- **variables_names_offset** - смещение (адрес) начала блока описания имен переменных (4 байта);
- **data_offset** - смещение (адрес) начала блока данных (4 байта).

Далее, после заголовка, следует блок VARS_NAMES, содержащий массив из variables_number структур типа archiver_var_t, описывающих имена и типы данных архивных переменных.

```
typedef struct {
  //имя архивной переменной
  char name[60];
  uint8_t type;
} archiver_var_t;
```

Где type может принимать значения из ARCHIVE_TAG_DATA_LEN:

```
enum ARCHIVE_TAG_DATA_LEN {
          ARCHIVE_TAG_DATA_LEN_4,
          ARCHIVE_TAG_DATA_LEN_8,
};
```

Далее следует блок VARS_DATA, содержащий данные архива. Данные представлены в виде массива data_rows элементов (строк). Каждый элемент (строка)

состоит из отметки времени, описываемой структурой archiver_t, содержащей номер цикла ПЛК RUN_NUMBER и отметку времени date_time типа timespec64 и далее последовательности значений. Количество значений - variables_number. Длина данных каждого значения - variable_data_len. Структура данных строки приведена на рисунке (см. Рисунок 55).

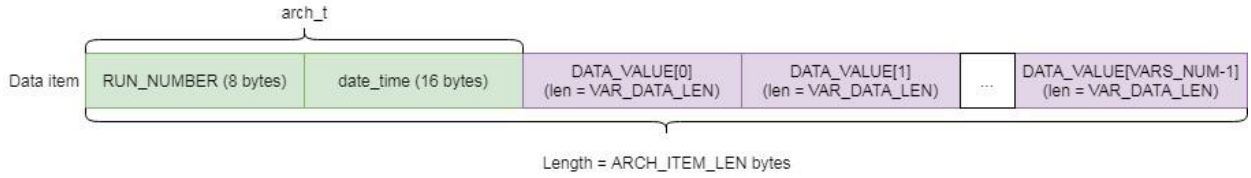


Рисунок 55 - Структура данных строки архива

Структура archiver_t описана следующим образом:

```
typedef struct {
  uint64_t RUN_NUMBER; //номер текущего цикла
  uint8_t date_time[16]; //время архивирования timespec64
} archiver_t;
```

8.2.3. Настройка системы архивирования

Добавление нового экземпляра модуля архивирования в проект производится вызовом из контекстного меню ПЛК в дереве проекта. Необходимо выбрать «Система архивирования». Изображено на рисунке (см. Рисунок 56).

В проект может быть добавлен только один модуль архивирования. Каждый модуль может содержать ряд архивов, каждый из которых будет иметь индивидуальные настройки. Таким образом, все переменные, подлежащие архивированию, могут быть логически разделены на архивы по типам. Архивам присваиваются имена, что позволяет определять их наименования в соответствии с их логическим назначением.

Окно настроек / создания системы архивирования логически разделено на два компонента: настройки – слева, добавленные переменные в архив – справа. Подразумевается возможность настройки нескольких архивов. Все переменные, добавленные в архив, будут иметь одинаковые правила архивирования.

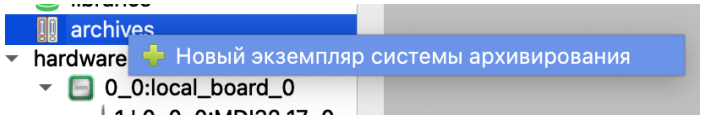


Рисунок 56 – Добавление модуля архивирования в проект

Окно системы архивирования с установленными настройками и переменными представлено на рисунке (см. Рисунок 57).

В открывшемся окне необходимо указать следующие параметры:

- имя – название архива, отображаемое в проекте;
- тип – стратегия архивирования (Триггерный/ Периодический);
- триггерная переменная – переменная, по фронту изменения которой архив будет записываться на диск;
- количество записей – размер кольцевого буфера (количество строк в базе данных);
- тип сохраняемого файла (SQLite/CSV/Бинарный файл);
- путь к файлу – путь, куда будут сохраняться файлы архивов;
- длина переменной – количество байт, занимаемое каждым элементом в системе архивирования (4 или 8).

Включить отладку – вывод диагностической информации в работе системы архивирования.

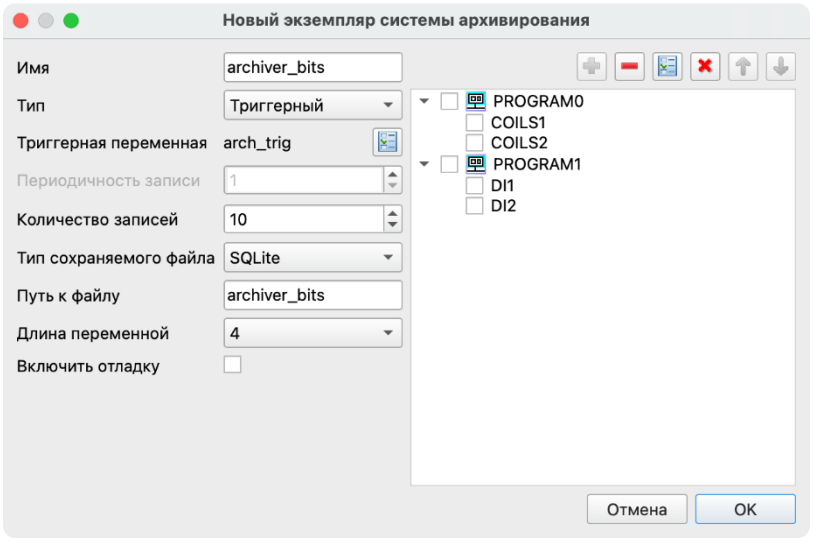


Рисунок 57 – Окно настроек системы архивирования

Элементы управления добавленными переменными:

- удаление выбранной переменной;
- добавление переменных из проекта;
- удаление всех переменных.

После заполнения полей настроек архива следует добавить необходимые переменные из программ. Отображаются только программы, участвующие в обмене (добавленные в resource). Пример добавления переменных указан на рисунке (см. Рисунок 58).

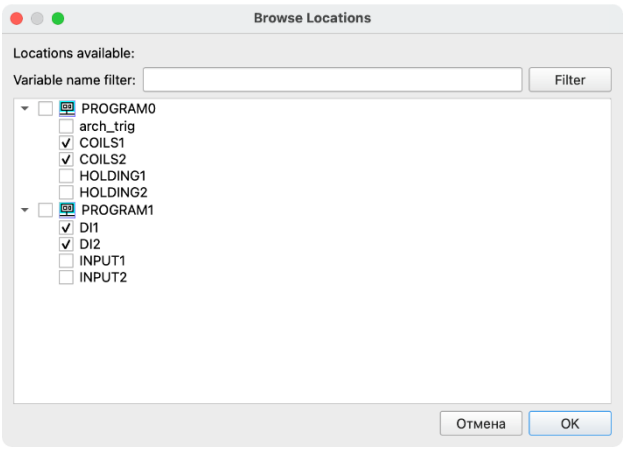


Рисунок 58 – Добавление переменных

8.3. Модуль протоколов Modbus

САПР ELPLC-LOGIC поддерживает протокол Modbus в сетях TCP/IP, а также для последовательных каналов передачи данных. Разработаны модули ModBus-TCP/RTU Master/Slave. Модуль поддерживатеся для всех платформ, на которых может работать исполнительная система ELPLC-LOGIC. Модуль может быть добавлен пользователем в проект в виде отдельных четырех компонентов: Modbus-TCP Master, Modbus-TCP Slave, Modbus-RTU Master, Modbus-RTU Slave. Каждый модуль имеет собственный набор настроек, описывающих оборудования для подключения, а также временные характеристики и настройки контроля ошибок.

8.3.1. Modbus-TCP/RTU Master

Modbus-TCP/RTU Master предназначен для организации опроса удаленных slave устройств. При этом, опрос выполняется асинхронно, в отдельном потоке, а не в основном цикле IEC программы. Работа плагина построена таким образом, что задаются параметры

организации асинхронного опроса, определяются типовые операции чтения (транзакции), выполняемые на каждом цикле, задаются временные параметры исполнения циклического опроса. Для каждого устройства для протокола TCP или последовательной линии для протокола RTU запускается отдельный поток опроса и выполняется независимо. Данные, получаемые от удаленных устройств, помещаются в специализированный буфер, который считывается IEC-программой в момент чтения входных данных (retrieve). Управляющие команды или другие выходные данные IEC программы выполняются синхронно, с блокировкой основного цикла IEC программы на стадии публикации выходных переменных с обеспечением синхронизации потоков. Таким образом, выполняется задача достоверной доставки управляющей команды на удаленное устройство. Контроль за процессом асинхронного опроса осуществляется через специализированные программно-формируемые плагином переменные, отображающие текущий статус канала опроса, а также время последней операции обмена. Показатель времени дает возможность программисту оценить загрузку коммуникационного канала, а также оценить общее время выполнения стадии публикации выходных данных для согласования циклов IEC-программы.

Важно отметить, что в отдельные потоки выносятся лишь регулярные операции чтения регистров удаленных устройств. Сформированные в системе исполнения команды записи выполняются своевременно, на такте PUBLISH системы исполнения, с обеспечением синхронизации с потоком опроса. Т.е. сформированный пул управляющих команд дожидается завершения ближайшей регулярной операции чтения, занимает линию и выдает эти команды в сеть. Временная диаграмма процесса показана на рисунке (см. Рисунок 59).

Таким образом, для каждого узла Modbus-TCP исполнительная система создаст независимый поток опроса и будет взаимодействовать с этим узлом. В случае с Modbus-RTU поток будет создан не для устройства, а для линии (последовательного порта). Соответственно, все устройства (до 247 устройств), подключенные на эту линию, будут выстраиваться в очередь последовательного обмена. При этом для обоих вариантов соединения команды управления будут встраиваться в цикл регулярного чтения в ближайшее окно.

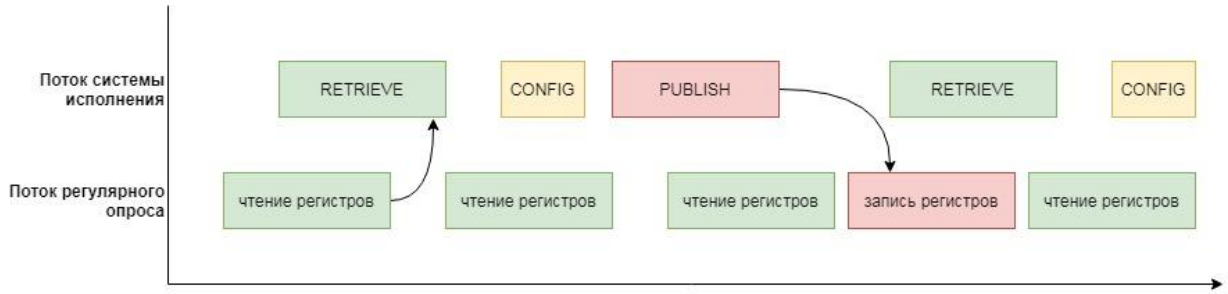


Рисунок 59 – Временная диаграмма чтения и записи Modbus

Контроль за процессом асинхронного опроса осуществляется через специализированные программно-формируемые плагином переменные, показывающие текущий статус канала опроса, а также время последней операции обмена. Показатель времени дает возможность программисту оценить качество коммуникационного канала, а также оценить общее время выполнения стадии публикации выходных данных для согласования циклов IEC-программы. С учетом специфики коммуникаций по последовательным каналам, особенно при использовании длинных линий связи, существует вероятность нарушения целостности данных при передаче. Таким образом, единичные ошибки данных не всегда могут считаться фактической ошибкой связи. Плагин Modbus RTU Master имеет соответствующие настройки, позволяющие определить количество последовательных отказов в исполнении транзакций, которое будет приводить к формированию программной ошибки связи. Такая настройка позволяет обеспечить гибкий подход к процессу формирования ошибок при использовании на линиях низкого качества. При этом, команды выполняются с соблюдением очередности, что гарантирует последовательное их поступление на устройство даже в случае нарушения связи.

8.3.1.1. Настройка протокола Modbus-TCP Master.

Редактор модуля (см. Рисунок 60) предоставляет настройки TCP соединения для работы по протоколу Modbus TCP.

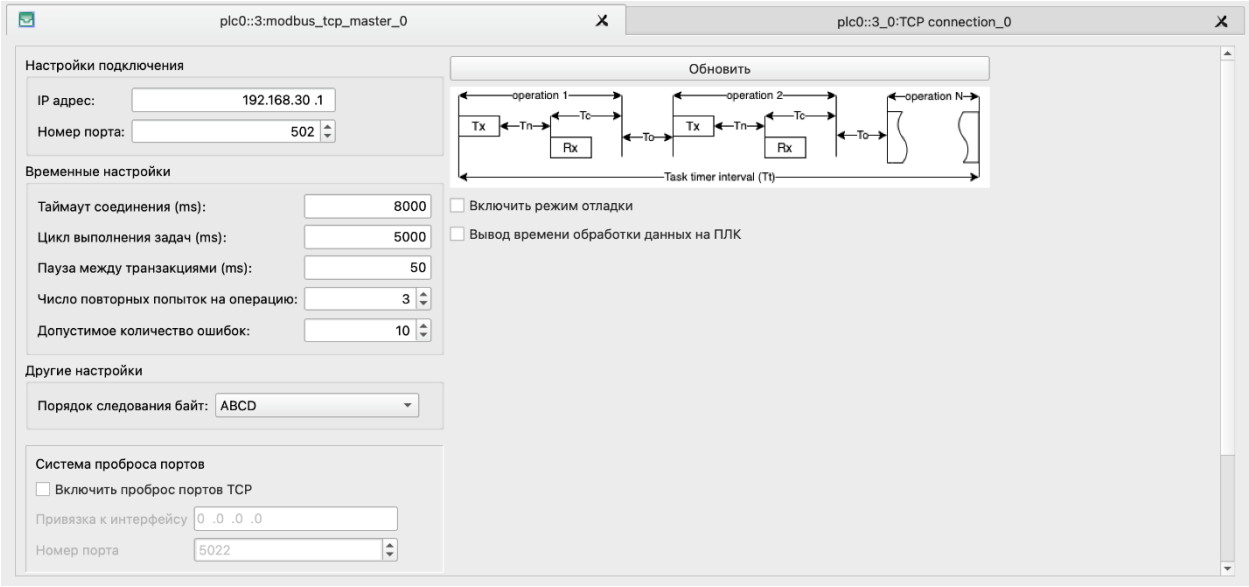


Рисунок 60 - Окно настройки Modbus-TCP Master

Окно имеет 2 вкладки: Общие настройки и Переменные.

Общие настройки плагина позволяют определить следующие параметры:

1. Настройки подключения:

- IP-адрес – адрес опрашиваемого устройства;

- номер порта – TCP порт подключения, через который происходит обмен данными. По умолчанию используется порт 502.

2. Временные настройки:

- таймаут соединения в миллисекундах (в скобках указаны рекомендованные значения) – время ожидания ответа на запрос. При превышении этого времени устройство будет считаться отказавшим, соединение будет разорвано;

- цикл выполнения задач - интервал в миллисекундах регулярного асинхронного опроса устройств (в скобках указаны рекомендованные значения);

- пауза между транзакциями – время паузы между выполнениями двух транзакций. Это время необходимо указывать отличным от 0, чтобы оставалась возможность синхронизации с другими потоками;

- число повторных попыток на операцию – количество попыток каждой транзакции в случае сбоя (доступно только для плагина modbus_rtu_slave);

- допустимое количество ошибок на операцию – количество ошибок связи для перехода в состояние отказа транзакции (доступно только для плагина modbus_rtu_slave);

- порядок следования байт – порядок байт в сообщении для регистров, имеющих тип данных REAL (4 байта).

В правой части окна настроек находится отображение временной диаграммы, демонстрирующей полный цикл обмена и устанавливаемые временные характеристики.

После настройки сетевого подключения необходимо добавить modbus-устройство (см. Рисунок 61).

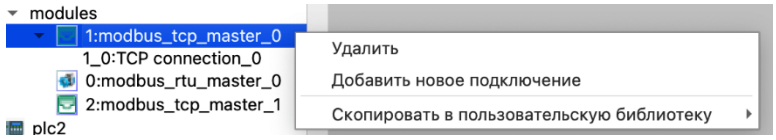


Рисунок 61 - Интерфейс добавления нового устройства к TCP подключению

В добавленном подключении (см. Рисунок 62) доступно измение поля «Адрес устройства» – UNIT_ID заголовка пакета.

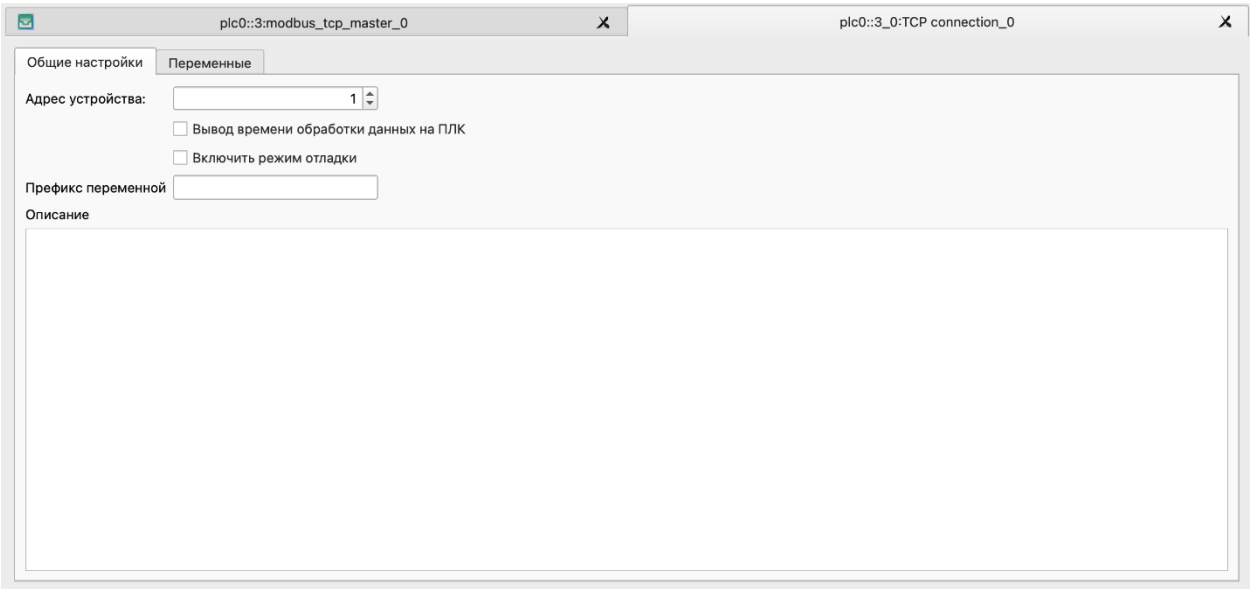


Рисунок 62 - Окно настройки Modbus-TCP Connection

8.3.1.2. Настройка протокола Modbus-RTU Master

После добавления плагина RTU Master создается линия соединения RS-485 (RTU connection). Далее на опрашиваемую линию должны быть добавлены устройства (см.

Рисунок 63). На каждой линии RS232/485 может быть расположено до 247 зависимых устройств.

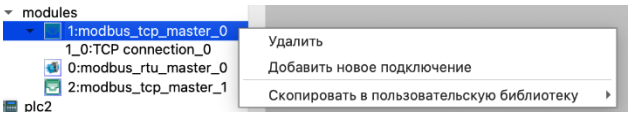


Рисунок 63 – Интерфейс добавления нового устройства на линию RTU

Редактор модуля Modbus RTU предоставляет настройки соединения.

Окно позволяет описать параметры RTU соединения. Указать устройство последовательного порта, задать скорость соединения и прочие настройки. Информацию об именах устройств последовательного порта следует получить из руководств на оборудование. Окно настройки представлено на рисунке (см. Рисунок 64).

Настройки соединения:

- устройство последовательного порта – пусть к устройству используемого последовательного порта в ОС (см. руководство на оборудование);
- скорость передачи данных – скорость (baud rate) передачи данных в линии;
- паритет – флаг использования бита паритета;
- биты данных – настройка количества бит данных последовательного порта;
- стоповые биты – количество стоповых бит;

Настройки временных интервалов. Определение временных характеристик работы соединения. Каждая операция на линии (транзакция) должна быть отделена по времени от предыдущей транзакции, в соответствии со спецификацией протокола Modbus. Для этого вводятся настройки, позволяющие оптимизировать процесс обмена и, при этом, выдержать необходимы паузы, чтобы устройства на линии могли идентифицировать пакеты, отделять их друг от друга. Следует учитывать, что если на линии подключено более одного ведомого устройства, то после ответа ведущему одного ведомого необходимо также выдержать паузу, чтобы второй ведомый смог идентифицировать начало нового пакета:

- таймаут соединения – время в мс, в течении которого ожидается ответ от ведомого. Если за указанное время ответ не получен, то транзакция считается ошибочной. В скобках указываются рекомендованные значения для выбранной скорости передачи данных:

- цикл выполнения задач - интервал времени таймера опроса, в миллисекундах (в скобках указаны рекомендованные значения);
- интервал между транзакциями – время в мс, которое будет выдерживаться тишина на линии после предыдущего обмена;
- число повторных попыток на операцию – количество попыток каждой транзакции в случае сбоя;
- допустимое количество ошибок на операцию – количество ошибок связи для перехода в состояние отказа устройства.

Система проброса портов. Исполнительная система имеет дополнительную функцию проброса портов. Это позволяет использовать проприетарное оборудование поставщиков оконечных устройств для прямого взаимодействия с ведомыми устройствами, подключенными на линию. Пакеты принимаются исполнительной системой через сеть TCP/IP и встраиваются в поток транзакций. Для этого выполняются следующие настройки:

- включить проброс портов TCP;
- привязка к интерфейсу – IP-адрес;
- номер порта – TCP порт подключения.

Дополнительная задержка RTS/DTR. В зависимости от примененного целевого устройства, существуют различные методы управления приемом-передачей для двух проводных линий RS-485. Ряд устройств имеют автоматическое управление, другие программным сигналом RTS или DTR. Блок позволяет задать эти настройки, а также установить необходимые временные параметры:

- включить задержку RTS – активация программного управления потоком;
- задержка в миллисекундах – время в мс задержки после установки сигнала управления перед началом передачи данных. Такое же время выдерживается и после снятия сигнала;
- режим RTS (Auto, RTS software control, DTR software control) – режим управления (RTS, DTR).

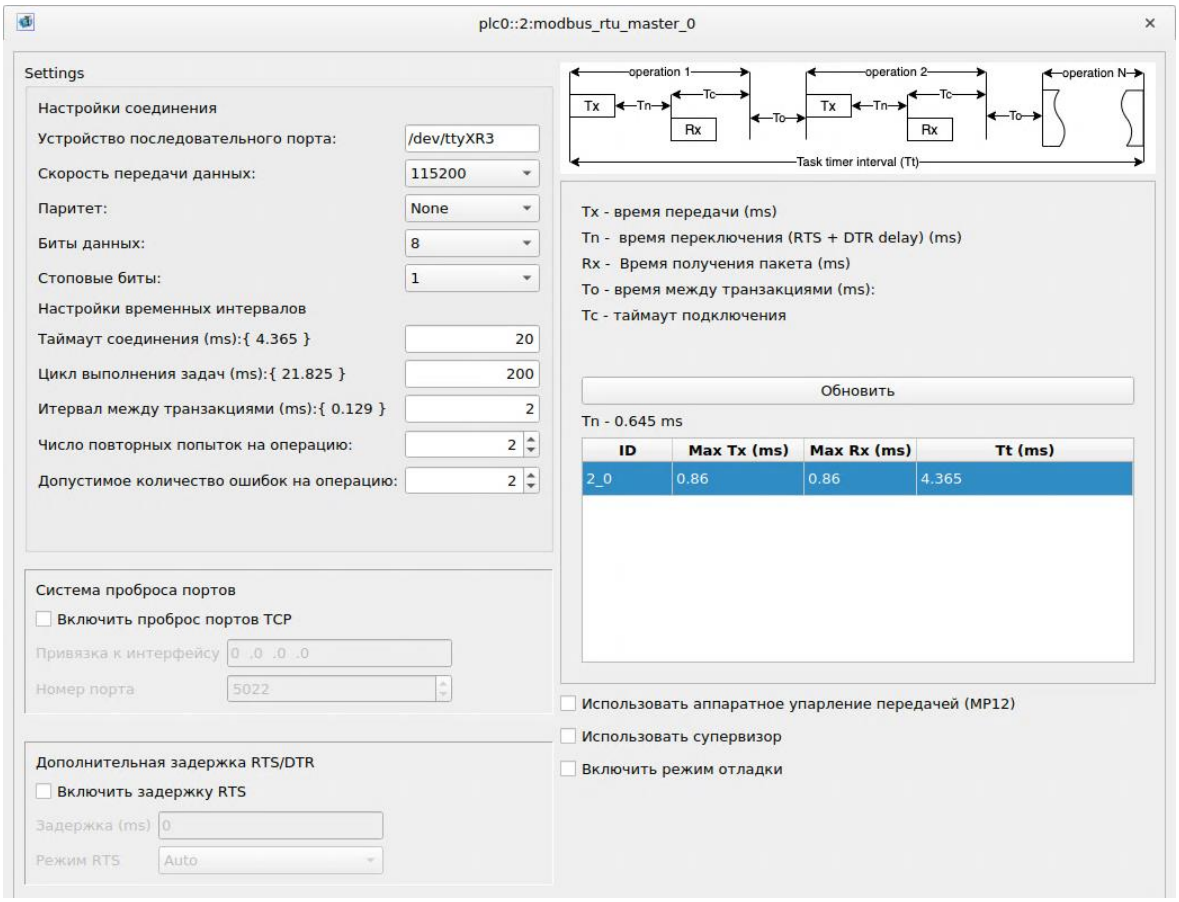


Рисунок 64 – Окно настройки Modbus-RTU

После настройки последовательного устройства к нему необходимо добавить набор узлов (ведомых устройств). Каждый узел имеет свои настройки (см. Рисунок 65).

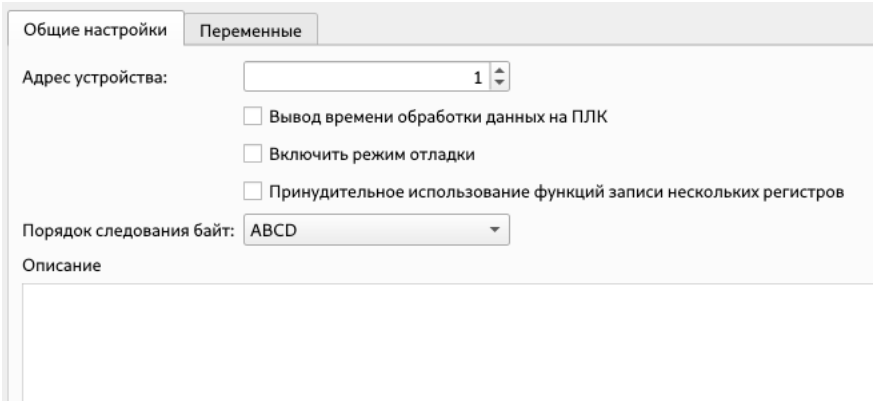


Рисунок 65 – Настройка узла Modbus-RTU

Настройки:

- адрес устройства – сетевой адрес ведомого устройства протокола Modbus-RTU;

- порядок следования байт – порядок байт в сообщении для регистров, имеющих тип данных REAL (4 байта).

Также окно содержит выключатели для отладки и оптимизации запросов.

8.3.1.3. Настройка переменных Modbus-master

Для протоколов Modbus-RTU и TCP применяется одинаковые окна описания переменных (регистров). Принцип описания заключается в группировке транзакций по типам, а затем по функциям. Так, на первом уровне опередяются группы:

- циклическое чтение;
- триггерное чтение;
- циклическая запись;
- триггерная запись.

Пример окна настройки переменных приведен на (см. Рисунок 34).

Группа «циклическое чтение» определяет набор операций чтения, которые будут выполняться в регулярном цикле потока транзакций. Т.е. с интервалом времени, определенным как «Цикл выполнения задач» для последовательной линии или для TCP узла будет выполняться последовательность операций чтения, определенных в этой группе.

Операции «циклическая запись» будут выполняться на каждом цикле ПЛК.

Группы «триггерное чтение» и «триггерная запись» используются для нерегулярного выполнения особых операций. Эти операции будут выполнены в случае выполнения условия триггера. Как правило – это целочисленная или бинарная переменная, по изменению которой операция будет исполнена однократно. Следующее выполнение будет возможно при фиксации нового фронта изменения сигнала. Сам сигнал сбрасывается системой после успешного выполнения.

Внутри каждой группы описываются функции протокола Modbus. Пользователь может определить и добавить необходимые функции. Далее, внутри функции, описываются регистры протокола и именуются в переменные. Типы данных переменных назначаются в соответствии с выбранной функцией.

Таблица настроек содержит поле «Индекс операпции». Это поле позволяет пользователю задать необходимую последовательность исполнения транзакций.

8.3.2. Modbus-TCP Slave

Модуль предназначен для создания ведомого устройства Modbus-TCP и настройки регистров, хранящихся в нем.

В настройках (см. Рисунок 66) необходимо указать IP-адрес и номер порта сервера для привязки к сетевому интерфейсу исполнительного устройства, а также порядок следования байт. В качестве IP адреса привязки возможно использование неопределенного адреса (0.0.0.0) для запуска сервера для всех сетевых интерфейсов ПЛК.

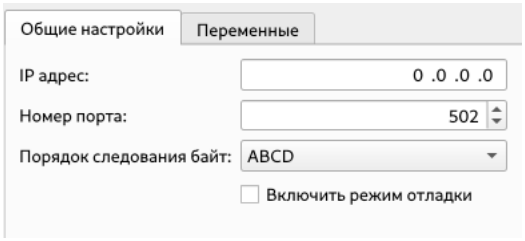


Рисунок 66 - Окно настройки сервера Modbus-TCP

Добавление переменных выполняется на вкладке «Переменные» (см. Рисунок 67). Здесь представлена таблица, содержащая разделы для дискретного ввода, катушек, входных регистров и хранимых регистров. Добавление переменных осуществляется с помощью кнопок «плюс» и «множественный плюс».

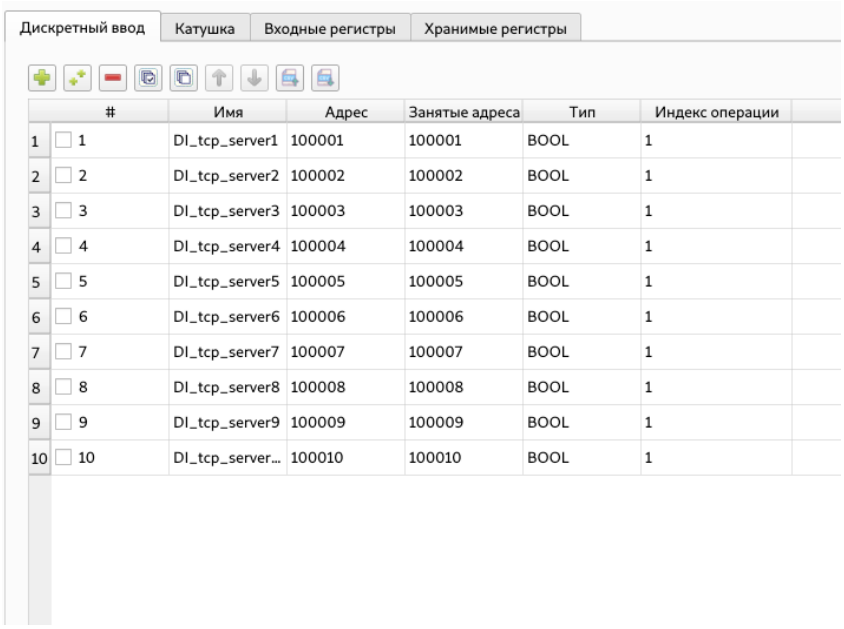


Рисунок 67 - Интерфейс добавления переменных для модуля Modbus-TCP Slave

Панель редактирования одиночного сигнала представлена на рисунке (см. Рисунок 68).

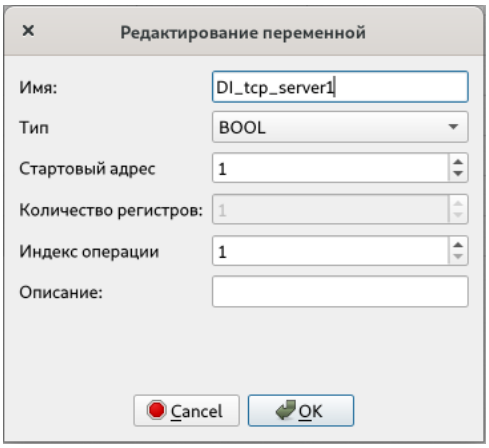


Рисунок 68 - Интерфейс добавления одиночного сигнала в Modbus Slave

Пользователь задает название переменной, тип сигнала и стартовый адрес. Редактирование переменной происходит по двойному щелчку левой кнопки мыши на строке редактируемого сигнала.

Вызвать форму добавления (см. Рисунок 69) нескольких переменных можно с помощью кнопки «множественное добавление». Форма содержит шаблон имени переменной, стартовый адрес, количество переменных, а также тип сигнала. Формирование имени переменной осуществляется с помощью добавления начального адреса к шаблону имени. Следующая переменная просто увеличивает порядковый номер переменной на 1.

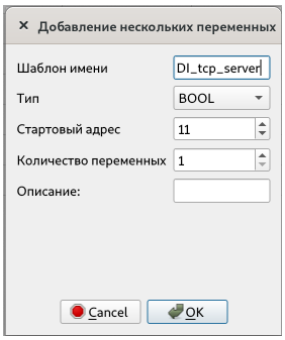


Рисунок 69 - Интерфейс добавления нескольких сигналов в Modbus Slave

Для удаления переменной из списка необходимо выделить чекбокс слева в строке добавленного сигнала и нажать кнопку «минус». Очистка списка переменных происходит последовательным нажатием кнопок «выбрать все» и «минус».

8.3.2.1. Настройка модуля Modbus-RTU slave

Модуль предназначен для настройки регистров, хранящихся на ведомом устройстве Modbus RTU.

Настройки модуля (см. Рисунок 70) позволяют пользователю задать устройство последовательного порта, с помощью которого будет вестись обмен информацией, сетевой адрес ПЛК, скорость передачи данных, паритет, указать биты данных, стоповые биты, а также выбрать порядок следования байт в сообщении.

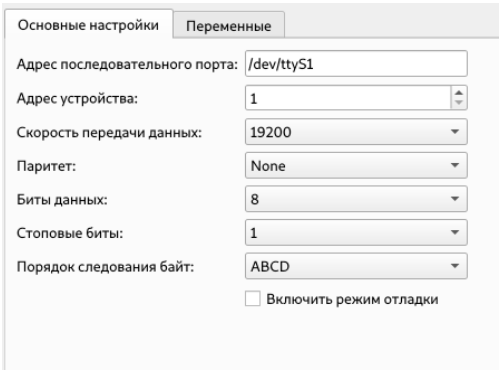


Рисунок 70 - Интерфейс настройки Modbus RTU slave

Добавление и работа с переменными переменных происходит как и в плагине “Modbus TCP сервер”.

8.3.3. Специализированные устройства Modbus

Для ряда устройств, часто применяющихся в проектах, созданы специальные шаблонные наборы регистров, позволяющие быстро и наглядно подключать такие устройства к программе. Интерфейс описания регистров таких устроств идентичен с устройствами общего назначения. Однако в окне настройки переменных есть возможность визуального выбора предустановленных регистров (см. Рисунок 62).

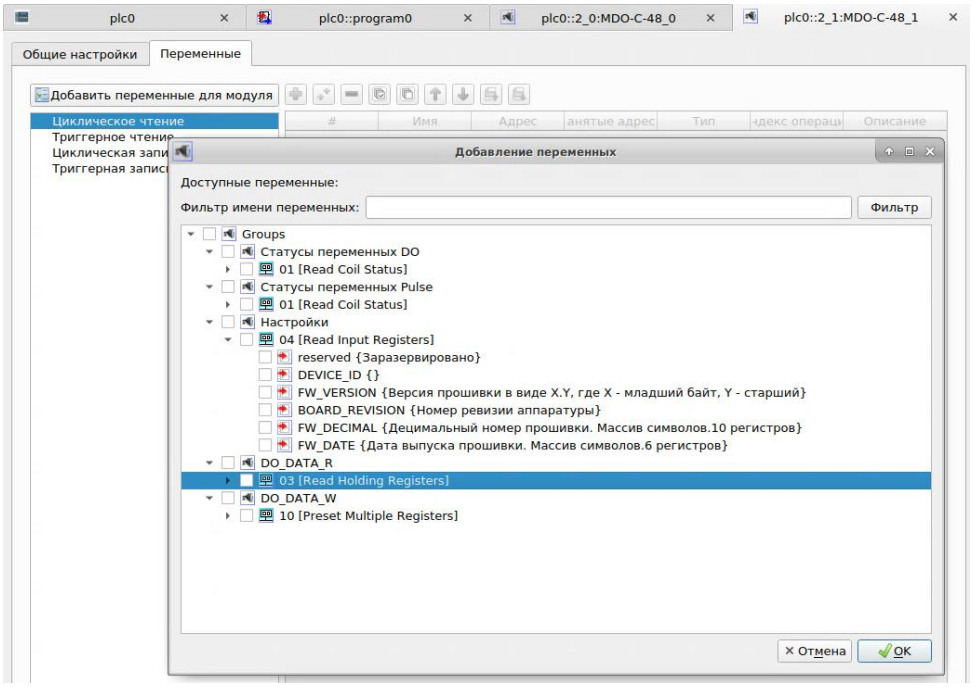


Рисунок 71 – Окно выбора регистров

Нажмите кнопку «Добавить переменные для модуля» и в открывшемся диалоге отметьте те регистры, которые необходимо считывать из модуля.

Такой подход значительно ускоряет разработку для известных типовых устройств.

8.3.4. Диагностика функционирования канала Modbus

Система исполнения ELPLC-LOGIC формирует ряд программных переменных, отражающих состояние и статистику подключений Modbus. Эти данные доступны в IEC программах. Для их получения необходимо перейти в программу, открыть окно выбора внешних переменных модулей (плагинов), в дереве обнаружить требуемое Modbus соединение. Переменные Modbus соединения разбиты на несколько логических групп (см. Рисунок 72).

Так, в дереве будут отображаться непосредственно перменные, сформированные по регистрам подключенных устройств, а также присутствует группа «Status variables». Эта группа содержит ряд служебных переменных. Рассмотрим их.

Переменная TASK_X_POLLS – программный счетчик проведенных циклов опроса. Автоматически инкрементируется по ходу работы программы.

Переменная TASK_X_RECONNS – счетчик переподключений. Переменная отображает количество переподключений для протокола TCP (после обрыва связи) или количество переинициализации последовательных устройств обмена.

Переменная TASK_X_FAULTS – счетчик сбоев. Переменная увеличивается в случае, когда на один из запросов не поступил ответ со стороны ведомого устройства и ведущий направляет повторный запрос.

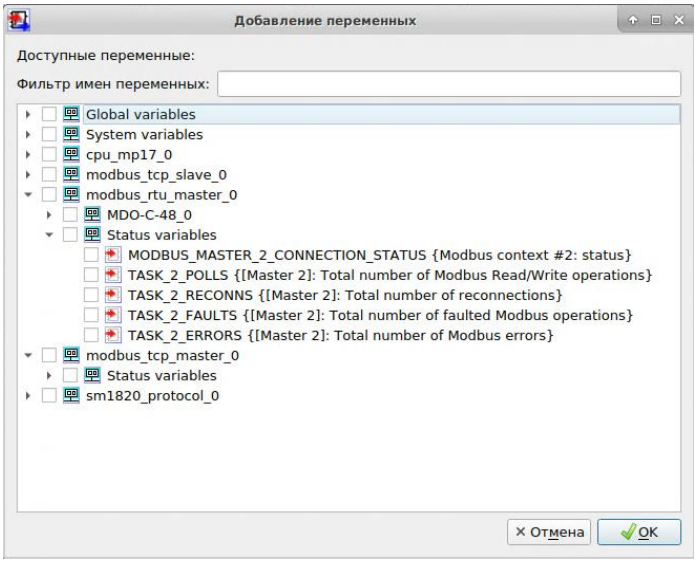


Рисунок 72 – Окно выбора переменных Modbus в программе

Переменная TASK_X_ERRORS – счетчик ошибок. В настройках соединения мы задавали допустимое количество повторов для транзакций. В случае с линиями RS485 есть вероятность возникновения помехи и, как следствие, отбраковки пакета. В этом случае ведущий направляет повторный запрос. Несколько ошибок подряд говорят об отказе устройства. Количество последовательных ошибок, после которых устройство будет считаться отказавшим задается в настройках линии. Этот параметр показывает количество уже установленных отказов устройства.

Также статистические переменные доступны для каждого устройства (см. Рисунок 73).

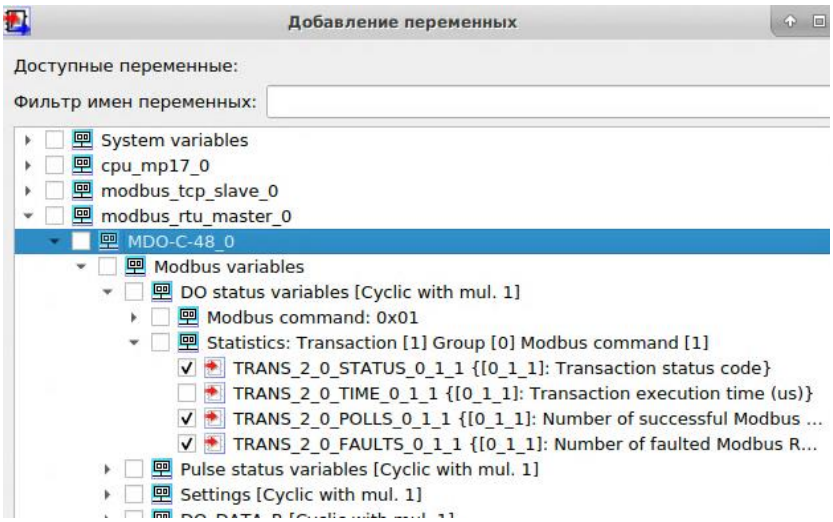


Рисунок 73 - Диагностические переменные устройства

Набор таких переменных доступен внутри устройства, в группе «Statistics».

TRANS_X_X_STATUS_XXX – статусная переменная исполнения транзакции. 0 – транзакция выполнена успешно. Статус транзакции присваивается по факту ее выполнения. Возможные значения указаны в таблице (см. Таблица 4):

Таблица 4 – Коды статуса транзакции

Описание	**Значение**
Неизвестный статус MB_TRANS_STATUS_UNKNOWN	0x0
Транзакция выполнена успешно MB_TRANS_STATUS_GOOD	0x1
Ошибка выполнения транзакции MB_TRANS_STATUS_ERROR	0x2
Недопустимая функция MB_EXCEPT_ILLEGAL_FUNCTION	0x3
Недопустимый адрес данных MB_EXCEPT_ILLEGAL_DATA_ADDRESS	0x4
Недопустимое значение MB_EXCEPT_ILLEGAL_DATA_VALUE	0x5
Исключение на ведомом MB_EXCEPT_SLAVE_OR_SERVER_FAILURE	0x6
Исключение при подтверждении MB_EXCEPT_ACKNOWLEDGE	0x7
Ведомый занят MB_EXCEPT_SLAVE_OR_SERVER_BUSY	0x8
Отрицательное подтверждение MB_EXCEPT_NEGATIVE_ACKNOWLEDGE	0x9
Ошибка памяти MB_EXCEPT_MEMORY_PARITY	0xA
Неизвестная ошибка	0xB

MB_EXCEPT_NOT_DEFINED	
Ошибка пути маршрутизации MB_EXCEPT_GATEWAY_PATH	0xC
Ошибка целевого устройства при маршрутизации MB_EXCEPT_GATEWAY_TARGET	0xD

Переменная TRANS_X_X_TIME_XXX содержит время в мкс. исполнения последней транзакции.

Переменная TRANS_X_X_POLLS_XXX содержит количество успешных выполнений транзакции.

Переменная TRANS_X_X_FAULTS_XXX содержит количество ошибок транзакции.

8.4. Работа с модулями ELPLC-BUS

САПР ELPLC-LOGIC полностью поддерживает функционал работы с модулями ввода-вывода на шине ELPLC-BUS. Конфигурация ПЛК описывается в редакторе с указанием мест расположения модулей в монтажном каркасе. Также указываются подключения корзин расширения и их состав модулей.

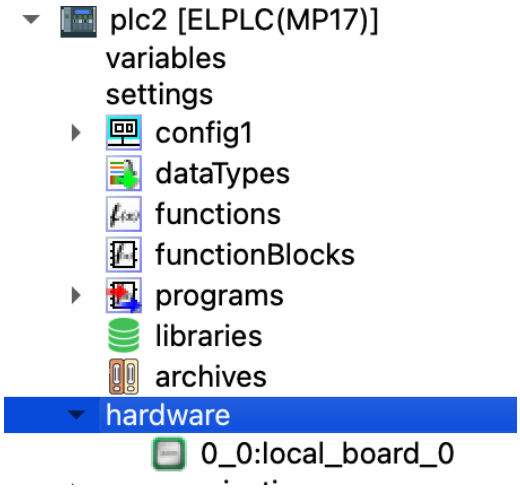


Рисунок 74 - Добавление модулей УСО ELPLC-BUS

Для добавления модулей откройте контекстное меню узла plc в дереве проекта и выберите аппартный обработчик (*hardware*) в дереве (см. Рисунок 74).

Модуль имеет ряд общесистемных настроек, позволяющих определить дисциплину работы шины, а также получить диагностическую информацию. Настройки размещены на трех вкладках (см.Рисунок 75).

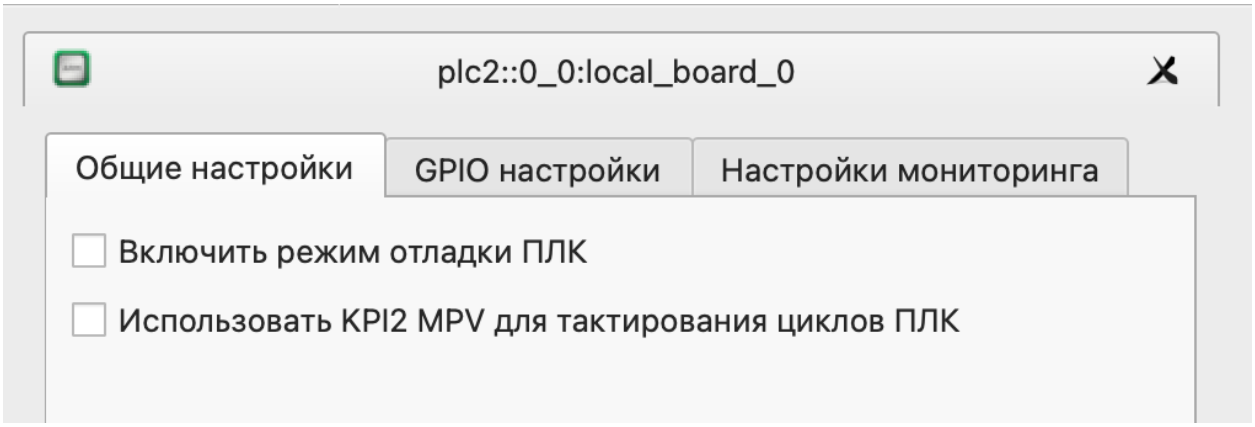


Рисунок 75 - Настройки модуля

На вкладке «Общие настройки» есть возможность активировать отладочный вывод для отображения диагностических сообщений в консоли ПЛК.

Также присутствует флажок активации устройства МПВ КПИ2. Флажок определяет дисциплину тактирования системы исполнения. Система исполнения ELPLC-LOGIC работает циклически, по таймеру. Период срабатывания таймера задается в окне настройки ресурсов проекта. Таймер настраивается на работу с заданным циклом и формирует последовательные события, по которым запускаются циклы ПЛК. В общем случае, для всех процессоров, используется POSIX таймер. Однако, в составе аппаратуры с архитектурой Эльбрус присутствует аппаратная система меток точного времени (МПВ), входящая в состав контроллера периферийных интерфейсов КПИ-2, примененного на процессорном модуле МП17. Эта система может быть активирована вместо POSIX таймеров в исполнительной системе ELPLC-LOGIC, что позволяет повысить точность циклов ПЛК при исполнении. Рекомендуется активировать подсистему МПВ, установив флажок ее использования.

На вкладке «Настройки мониторинга» существует возможность активировать ряд параметров системного мониторинга шины ELPLC-BUS и процессорного модуля. Это показатели напряжения питания, потребляемого тока, температуры и т.д. Окно настройки представлено на рисунке (см.Рисунок 76).

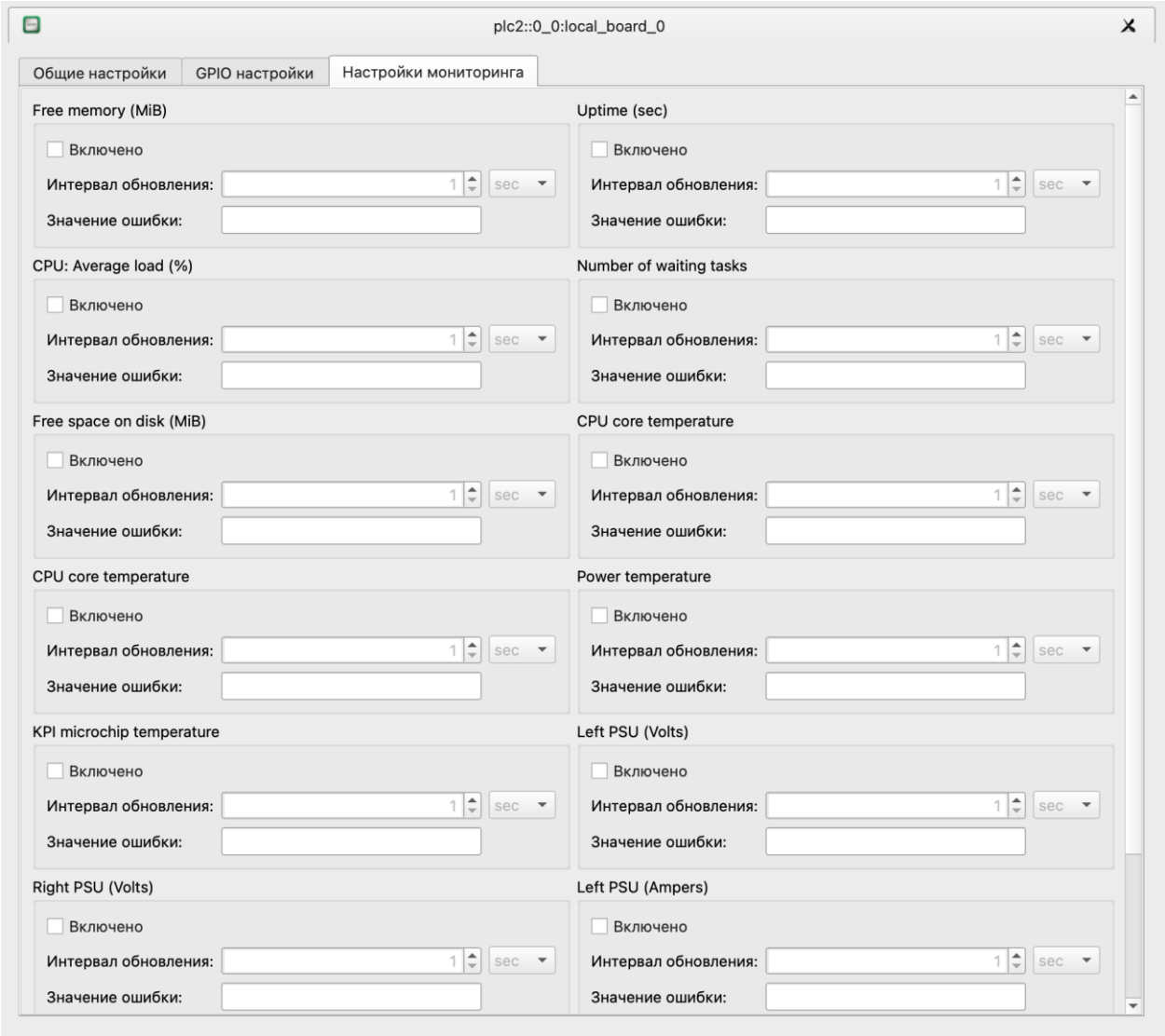


Рисунок 76 - Окно настройки мониторинга

Доступны к выбору следующие параметры, которые могут быть использованы в программе пользователем в виде IEC переменных:

- средняя нагрузка – загрузка центрального процессора;
- свободная память – размер свободной оперативной памяти процессорного модуля;
- свободное место на диске – объем свободного места на встроенном жестком диске процессорного модуля ПЛК;
- температура PWR – температура на датчике, размещенном на процессорном модуле вблизи с микросхемой вторичного электропитания;
- температура KPI – температура микросхемы КПИ-2;

- температура CPU1 – температура центрального процессора;
- блок питания – общее напряжение питания шины ELPLC-BUS;
- напряжение питания CPU1 – напряжение питания левого источника;
- напряжение питания CPU2 – напряжение питания правого источника;
- ток блока питания CPU1 – потребляемый ток от левого источника;
- ток блока питания CPU2 – потребляемый ток от правого источника.

Каждый из параметров может быть выбран для опроса, также может быть задан интервал опроса и значение ошибки, которое будет использовано для присвоения Condition переменной, соответствующей переменной параметра.

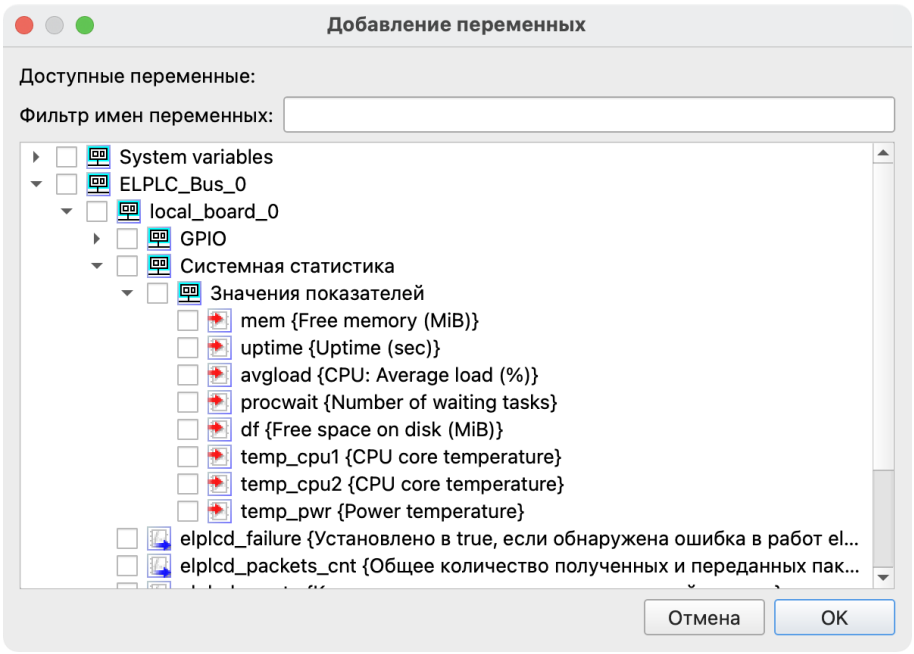


Рисунок 77 - Параметры мониторинга в программе

Все выбранные параметры могут быть доступны в IEC программе в виде переменных (см.Рисунок 77). Сформированные ошибки выделяются также в отдельный блок.

Добавив основной процессорный модуль пользователь получает возможность добавления модулей ввода-вывода, установленных в монтажный каркас ПЛК. Обеспечивается жесткая привязка модуля к слоту в монтажном каркасе.

Внимание! В случае физической перестановки модуля в другой слот эти изменения необходимо отразить в настройках. В противном случае данные будут присваиваться другим переменным.

Добавление модулей производится через контекстное меню узла hardware.

В контекстном меню локальной корзины (local_board) (см. Рисунок 78) доступны поддерживаемые модули ELPLC_Bus.

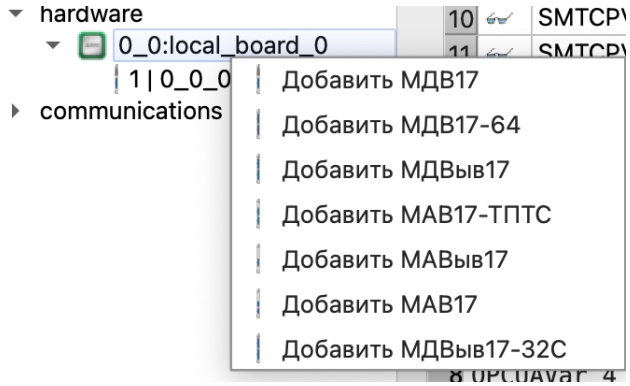


Рисунок 78 - Контекстное меню добавления модулей УСО

Меню hardware также содержит перечень пункт «Добавить “Удаленная корзина”», что позволяет описать подключение корзин расширения ПЛК.

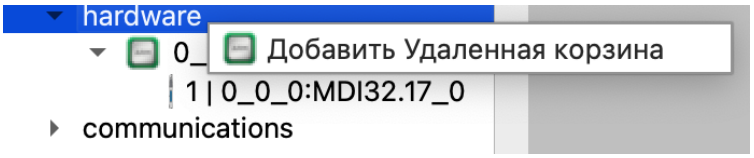


Рисунок 79 - Дерево с модулями УСО

Добавленные модули отображаются в виде дерева (см.Рисунок 79). При этом перед наименованием модуля указывается номер слота модуля, а уже затем его код в системе.

Внимание! Добавление более одного модуля на один и тот же слот не допускается. САПР выдаст ошибку при сохранении программы.

8.4.1. Настройки модулей и горячая замена

Модули шины ELPLC-BUS поддерживают «горячую замену». Т.е. неисправный модуль может быть извлечен из монтажного каркаса без выключения ПЛК и остановки программы, а также может быть подключен обратно. При этом важно отметить, что в момент добавления при работающем ПЛК (hotplug), модуль, подключаясь в систему, проходит проверку настроек и эти настройки меняются, если необходимо. Конфигурационное пространство каждого модуля УСО на шине ELPLC-BUS имеет несколько регистров, по данным которых высчитывается код CRC. Это позволяет системе

контролировать соответствие конфигурации модуля тому, что было задано разработчиком. В случае несовпадения CRC будет выполнена настройка с сохранением параметров в EEPROM модуля. Таким образом, модули не настраиваются каждый раз при запуске программы ПЛК. Происходит лишь верификация настроек.

Рассмотрим описание настроек модулей ввода-вывода.

8.4.2. Модуль MAB17

Модуль MAB17 может функционировать в различных режимах и иметь ряд настроек для четырех групп каналов (по 4 канала в каждой группе). Это выбор типа сигнала, выбор времени преобразования и т.д. Конфигурация модуля может быть задана с помощью тестового программного обеспечения, а также в системе программирования ELPLC-LOGIC.

Измерение значения входного сигнала, его преобразование в цифровой код производится автоматически по заложенной в модуле программе. Выходные данные модуля передаются в процессорный модуль через системный интерфейс контроллера.

Модуль MAB17 САПР ELPLC-LOGIC обеспечивает функционирование 16 каналов измерения и реализует следующие функции:

- настройку режима ввода и диапазон измерений;
- настройку градуировок и обеспечение связи с переменными IDE Эльбрус;
- установку сигналов качества каналов и обеспечение связи с переменными IDE Эльбрус.

На вкладке «Общие настройки» пользователь задает слот модуля в корзине ПЛК, а также конфигурирует режимы работы групп каналов модуля в соответствии с рисунком (см. Рисунок 80).

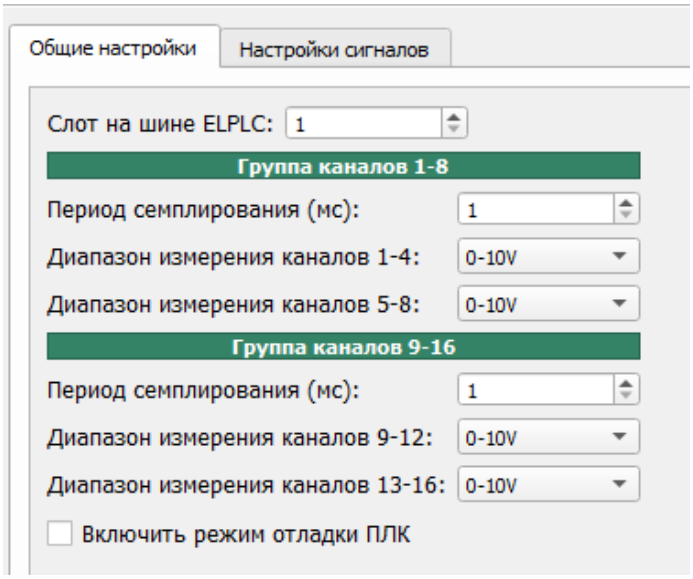


Рисунок 80 - Окно настроек модуля MAB17

Модуль MAB17 имеет в своем составе 2 АЦП по 8 каналов, для каждого из АЦП функционирует 2 группы каналов, в каждой группе содержится 4 канала аналогового ввода. Каждая группа может быть настроена на соответствующий режим ввода.

Для каждого из 8 каналов можно установить время преобразования и работу программного фильтра. АЦП модуля выполняет преобразования каждую 1 мс. Настройки допускают установку множителя от 1 до 10 для определения параметров работы программного фильтра. При указании значения 10 модуль будет выполнять 10 преобразований, по результатам будет производить усреднение данных с отбросом крайних значений и результат передавать процессорному устройству. В этом случае, обновление данных будет происходить 1 раз в 10 мс. При указании значения 1 новые данные будут каждую 1 мс.

Модуль поддерживает два режима ввода аналоговых сигналов: ток и напряжение. Поддерживаемый диапазон напряжения: от 0 до 5 В, от 0 до 10 В. Для режима тока вычисления выполняются исходя из применения нагрузочного резистора номиналом 240 Ом при опорном напряжении 3,3 В.

Вкладка "Настройка сигналов" отображает полный список каналов аналогового ввода. Щелчок на канале выводит текущие настройки сигнала на панели справа в соответствии с рисунком (см. Рисунок 81).

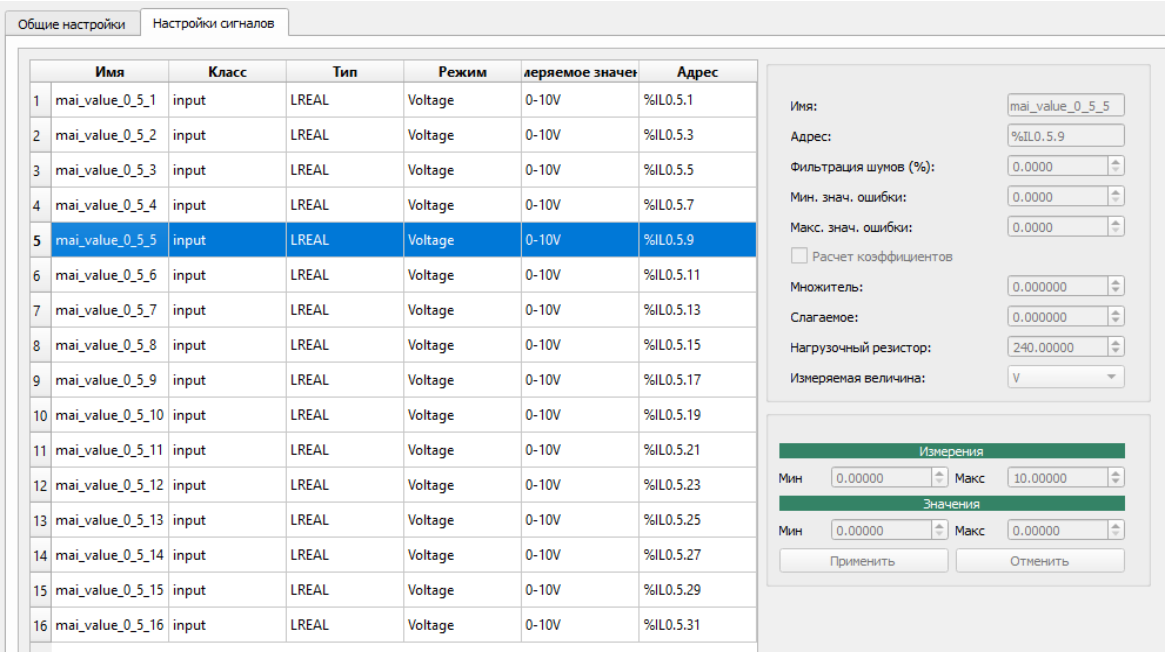


Рисунок 81 - Настройки каналов модуля MAB17

Каждому каналу аналогового ввода по умолчанию присваивается автоматически сгенерированное имя, которое может быть изменено пользователем. Тип данных сигнала аналогового ввода - LREAL. Каждому каналу соответствует дополнительный сигнал качества (condition), отражающий текущий статус переменной аналогового ввода. Тип данных сигнала качества - DINT. Сигнал качества может принимать различные значения, отличные от 0. 0 – сигнал достоверен. Все прочие значения сигнализируют о наличии той или иной ошибки. Эти ошибки могут быть присвоены на уровне модуля при диагностике аппаратной проблемы (ошибка АЦП, выход за пределы измерения) или на уровне драйвера модуля, также по факту возникновения проблем с оборудованием или границами измерения сигнала. Таким образом, сигнал считается достоверным только в случае если соответствующая ему переменная condition равна 0.

Значения кодов качества:

- 0x0 – нет ошибок;
- 0xF0 – сигнал недостоверен из-за ошибки модуля;
- 0x1 – сигнал недостоверен;
- 0x2 – физическое значение вне верхней границы;
- 0x3 – физическое значение вне нижней границы.

Модуль MAB17 выдает в программу значение в виде цифрового кода (кванты) длиной 16 бит. Далее этот код преобразуется и присваивается переменной с плавающей точкой. При этом вычисления производятся по формуле (1):

$$\mathrm{V} = \frac{q * V_{max} * div}{q_{max}} \quad (1)$$

где q – измеренные кванты;

V_max – опорное напряжение АЦП;

div - делитель, выбранный в соответствии с режимом измерения канала;

q_max – максимальное количество квантов (4096).

Поскольку модуль оснащен 12-разрядным АЦП, то диапазон выдаваемых цифровых кодов получается от 0 до 4095 квантов, итого 4096.

Получение токовых значений выполняется посредством формулы (2):

$$I = \frac{V}{R} \quad (2)$$

где V – напряжение;

R – нагрузочный резистор.

Например. С АПЦ поступает 1000 квантов. Необходимо вычислить получаемое значение тока.

Расчет. Поскольку, для режима тока вычисления выполняются исходя из применения нагрузочного резистора номиналом 240 Ом при напряжении 3,3 В и делителе 1,6 на 20 мА воспользуемся формулой (1), и вычислим напряжение:

$$V = \frac{q * V_{max} * div}{q_{max}} = \frac{1000 * 3.3 * 1{,}6}{4096} = 1{,}289 \text{ В}$$

По формуле (2) вычислим значение тока:

$$I = \frac{V}{R} = \frac{1{,}289}{240} = 5{,}371 \text{ мА}$$

Получим значение тока 5,371 мА.

Форма настройки сигнала вызывается по двойному щелчку на канале из списка сигналов в соответствии с рисунком (см. Рисунок 82). Множитель и слагаемое используются для указания градуировки датчика. Пользователь может самостоятельно задать коэффициент множителя и слагаемого или предоставить расчет этих коэффициентов среде разработки.

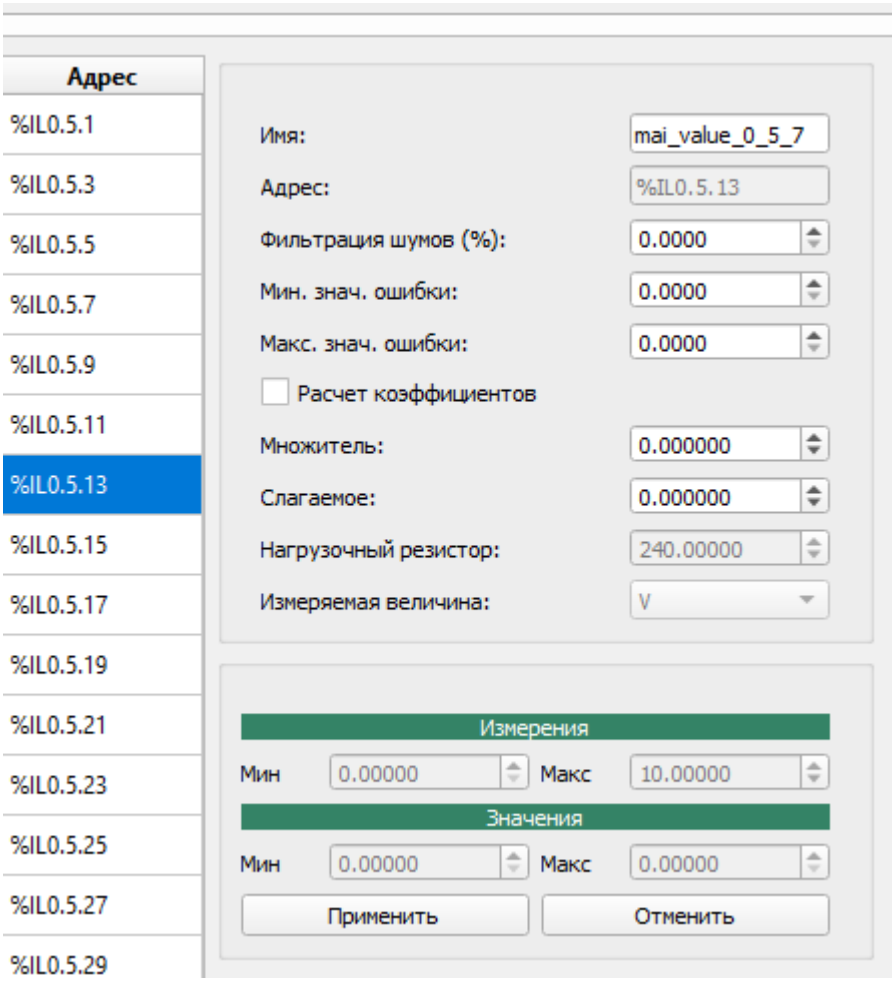


Рисунок 82 - Диалог настройки сигнала аналогового ввода

Нормирование сигнала производится по следующей формуле (3):

$$y = kx + b \tag{3}$$

где y – значение, получаемое по шкале датчика устройства, которое необходимо найти;

x – физическое значение сигнала, полученное на модуле аналогового ввода;

к – множитель;

b – слагаемое.

Поскольку, мы имеем координаты точек линейного уравнения, то можем восстановить формулу самого уравнения. Мы имеем 2 набора координат (XminMav, YminVal), (XmaxMav, YmaxVal). Строим систему уравнений, формула (4):

$$\begin{cases} K * XminMAV + B = YminVal \\ K * XmaxMAV + B = YmaxVal \end{cases} \tag{4}$$

Путем решения системы уравнений находим коэффициенты множителя и слагаемого по формуле (5) и формуле (6):

$$K = \frac{YmaxVal - YminVal}{XmaxMAV - XminMAV} \quad (5)$$

$$B = \frac{YminVal * XmaxMAV - YmaxVal * XminMAV}{XmaxMAV - XminMAV} \quad (6)$$

Или воспользоваться формулой (7):

$$B = -1 * \frac{YmaxVal * XminMAV - YmaxValXminMAV}{XmaxMAV - XminMAV} \quad (7)$$

Таким образом, нормирование сигнала происходит с помощью следующего выражения, формула (8):

$$Y = \frac{YmaxVal - YminVal}{XmaxMAV - XminMAV} * X - \frac{YmaxVal * XminMAV - YmaxValXminMAV}{XmaxMAV - XminMAV} \quad (8)$$

В верхней части формы пользователь задает минимальное и максимальное значение ошибки для автоматического присвоения кодов качества, а также фильтрацию шумов (гистерезис или чувствительность входа).

Поля, находящиеся в области «Измерения» относятся к модулю аналогового ввода и недоступны для редактирования и просто информируют пользователя о границах выбранного сигнала. Тип измерений соответствует типу модуля. Область «Значения» предназначена для введения границ измерений датчиков. Для области значений тип измерений не устанавливается.

Если пользователь отмечает чекбокс «Расчет коэффициентов», то поле «Значения» становится активны. Сохранение результатов осуществляется по нажатию кнопки "Применить".

Для взаимодействия с сигналами аналогового ввода, а также получения статусных сигналов необходимо добавить сигналы в программный модуль ELPLC-LOGIC.

8.4.3. Модуль МДВ17

Модуль МДВ17 имеет 32 канала дискретного ввода. Окно настройки модуля позволяет выполнить:

- настройку режима работы каналов;
- настройку периода самодиагностики модуля и антидребезговой схемы;
- обеспечение связи с переменными ELPLC-LOGIC;

- установку сигналов качества каналов и обеспечение связи с переменными ELPLC-LOGIC.

На вкладке «Общие настройки» пользователь задает слот модуля в корзине контроллера, а также конфигурирует режим работы модуля в соответствии с рисунком (см. Рисунок 83).

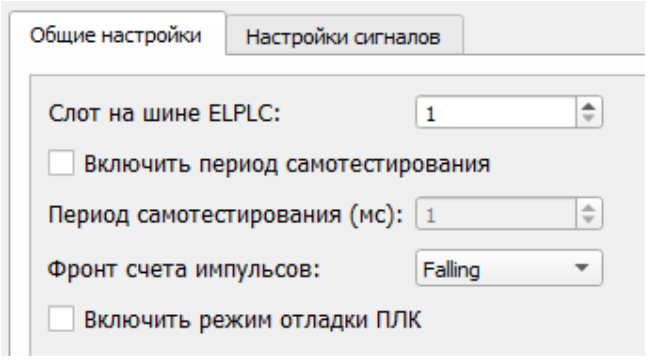


Рисунок 83 – Общие настройки модуля МДВ17

Модуль имеет в своем составе микроконтроллер, что позволяет возложить на него часть функций по фильтрации дребезга контактов, самодиагностике и т.д. В результате, взаимодействие по системному интерфейсу с модулем заключается в чтении подготовленных данных ввода с автоматически сформированными признаками качества на основании работы системы самодиагностики, а также в записи управляющих команд, позволяющих задать корректные параметры работы модуля. Также, за счет встроенного микроконтроллера, некоторые каналы могут быть перенастроены в режим программного 32-х разрядного счетчика.

Самодиагностика позволяет производить тестовые включения на аппаратном уровне всех каналов в уровни логического нуля и логической единицы. В результате тестированию подвергается большая часть электронных компонентов модуля, что позволяет выявлять сбои в работе аппаратуры.

Каждому каналу дискретного ввода по умолчанию присваиваются автоматически сгенерированные имена, которые могут изменяться пользователем. Тип данных сигнала дискретного ввода – BOOL, если канал сконфигурирован в режиме счетного ввода - UDINT. Каждому каналу соответствует дополнительный сигнал качества, отражающий текущий статус переменной дискретного ввода. Тип данных сигнала качества - DINT.

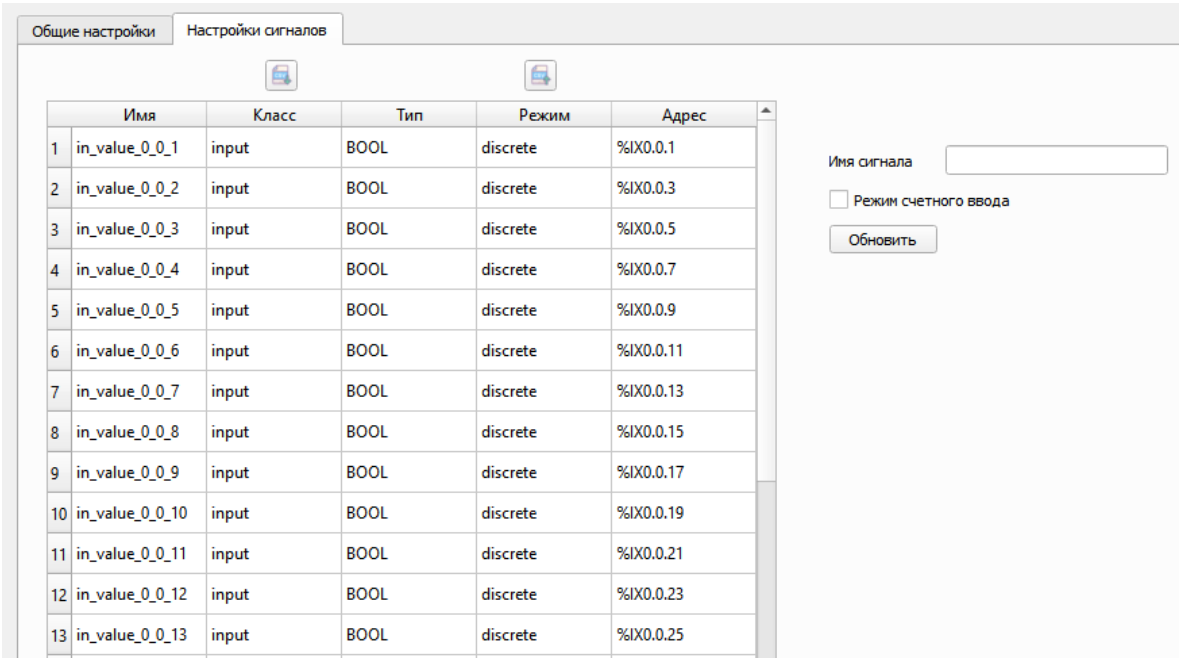


Рисунок 84 – Окно настроек сигналов МДВ17

Настройка канала производится посредством двойного щелчка по выбранному из списка каналу, представленном на рисунке (см. Рисунок 84). Для каждого канала пользователь может настроить антидребезговую защиту и указать длительность фильтра антидребезга. Доступные значения фильтра антидребезга – от 3 до 65535 мс. В указанной форме пользователь может задать новое имя сигнала и выбрать режим работы канала. Применение изменений осуществляется по кнопке «Обновить» (см. Рисунок 76).

Каналы с 4 по 11 способны функционировать в 2-х режимах:

- как дискретный ввод;
- как счетчик импульсов.

Рисунок 85 – Настройки каналов МДВ17

Обработка сигналов, сконфигурированных как счетчики дискретных импульсов, производится по одному из фронтов счета дискретных импульсов, задаваемых в поле «Фронт счета импульсов». Поддерживаемые режимы:

– падающий фронт (используется по умолчанию). Представляет сигнал типа 0-1-0;

– возрастающий фронт. Представляет сигнал типа 1-0-1.

Каналы 7 и 8 поддерживают работу в режиме счетчиков быстрых импульсов, при условии, что остальные каналы функционируют в качестве дискретного ввода.

Если хотя бы один канал работает в режиме «счетчик», то периодическое внутреннее тестирование запрещается. Также, в этом случае, пользователю запрещено активировать антидребезговую защиту канала. Для режима счетного ввода каналов поддерживается фильтр счетчика импульсов, измеряемый в микросекундах. Доступные значения от 1 до 8 мк.

Статусные сигналы каждого канала могут иметь следующие значения:

- 0x0 – нет ошибок;
- 0xF0 – сигнал недостоверен из-за ошибки модуля.

Также, статусный сигнал представляется комбинацией бит, где:

- Бит 0 – продолжительность дребезга более двукратного значения параметра фильтра антидребезга;
- Бит 1 – ошибки теста каналов на размыкание;
- Бит 2 – ошибки теста канала на замыкание.

8.4.4. Модуль МАВыв17

Модуль аналогового вывода имеет 8 каналов с настраиваемым диапазоном и типом (ток или напряжение). Модуль также имеет встроенную самодиагностику, позволяющую определить перегрузку канала, короткое замыкание, отказ питания.

Окно настройки модуля позволяет задавать режимы работы каналов и диапазоны вывода, связать управляющие каналы с переменными программы, получать информацию о состоянии каналов самодиагностики.

На вкладке «Общие настройки» пользователь задает слот модуля в корзине ПЛК.

Каждому каналу аналогового вывода по умолчанию присваиваются автоматически сгенерированные имена, которые могут изменяться пользователем. Тип данных сигнала аналогового ввода - LREAL.

Каждому каналу соответствует дополнительный сигнал качества, отражающий текущий статус переменной аналогового вывода. Тип данных сигнала качества - DINT.

Разрядность ЦАП составляет 12 бит, которое будет пересчитано в физическую величину в соответствии с установленными диапазонами измерений.

Каждый канал поддерживает три режима работы: ток от 0 до 20 мА, напряжение от 0 до 5 В, напряжение от 0 до 10 В.

Преобразование физической величины в цифровой код для генерации ЦАП выполняется по следующей формуле (9):

$$q = \left(\frac{V}{V_{max}} * q_{max}\right) + 0{,}5 \quad (9)$$

где:

V – генерируемый физический сигнал;

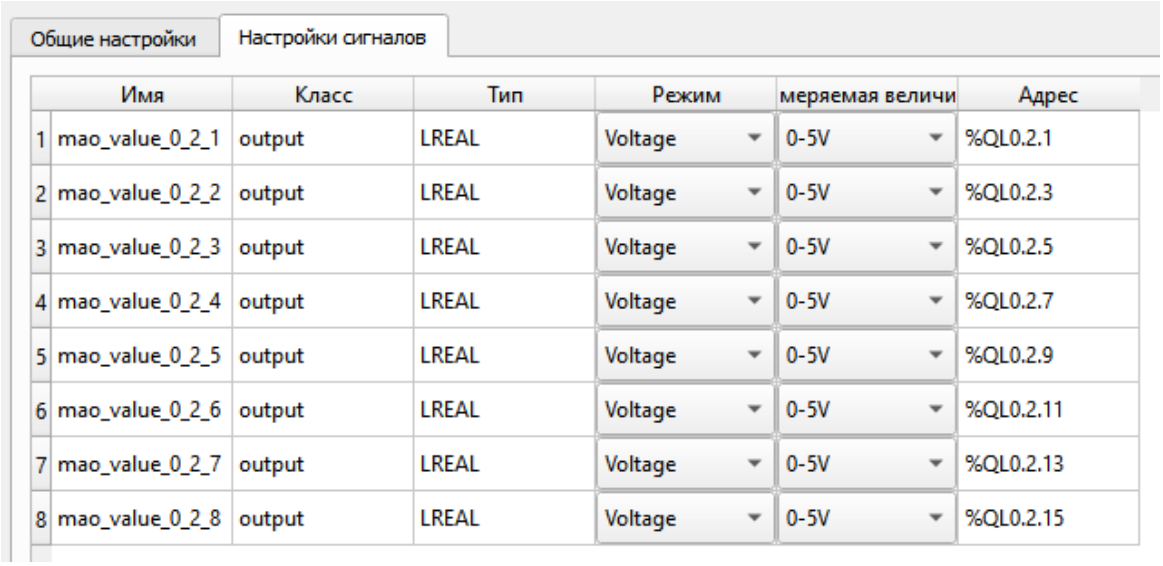
V_max – максимальное значение физической величины в соответствии с выбранным режимом канала в таблице (см. Таблица 5);

q_max – целое число, максимальное количество квантов при работе (4095);

Таблица 5 – Максимальное значение физической величины модуля МАВыв17

Наименование	Значение
V_{max} для режима от 0 до 5 В	5
V_{max} для режима от 0 до 10 В	10
V_{max} для режима от 0 до 20 мА	20

Сигнал может быть настроен индивидуально на соответствующий режим вывода в соответствии с рисунком (см. Рисунок 86). По двойному щелчку по каналу в столбце «Имя» пользователь может изменить имя сигнала. Режим работы канала и измеряемая величина доступны для редактирования путем выбора желаемого значения в соответствующих выпадающих списках.



Общие настройки | Настройки сигналов

	Имя	Класс	Тип	Режим	меряемая величи	Адрес
1	mao_value_0_2_1	output	LREAL	Voltage	0-5V	%QL0.2.1
2	mao_value_0_2_2	output	LREAL	Voltage	0-5V	%QL0.2.3
3	mao_value_0_2_3	output	LREAL	Voltage	0-5V	%QL0.2.5
4	mao_value_0_2_4	output	LREAL	Voltage	0-5V	%QL0.2.7
5	mao_value_0_2_5	output	LREAL	Voltage	0-5V	%QL0.2.9
6	mao_value_0_2_6	output	LREAL	Voltage	0-5V	%QL0.2.11
7	mao_value_0_2_7	output	LREAL	Voltage	0-5V	%QL0.2.13
8	mao_value_0_2_8	output	LREAL	Voltage	0-5V	%QL0.2.15

Рисунок 86 - Список каналов модуля вывода аналоговых сигналов

Для отладки и взаимодействия с сигналами аналогово ввода, а также получения статусных сигналов необходимо добавить сигналы в программный модуль IDE Эльбрус.

Статусные сигналы каждого канала могут иметь следующие значения:

- 0x0 – нет ошибок
- 0xF0 – сигнал недостоверен из-за ошибки модуля

8.4.5. Модуль МДВыв17

Модуль дискретного вывода МДВыв17 является управляющим модулем и имеет 32 канала.

На вкладке «Общие настройки» пользователь задает слот модуля в монтажном каркасе контроллера.

Каждому каналу дискретного вывода по умолчанию присваиваются автоматически сгенерированные имена, которые могут изменяться пользователем. Тип данных сигнала дискретного вывода – BOOL. Каждому каналу соответствует дополнительный сигнал качества, отражающий текущий статус переменной дискретного вывода. Тип данных сигнала качества - DINT.

Пользователь может задать новое имя сигнала по двойному щелчку на требуемом канале из списка.

Для взаимодействия с сигналами дискретного вывода, а также получения статусных сигналов необходимо добавить сигналы в программный модуль среды разработки IDE Эльбрус.

Статусные сигналы каждого канала могут иметь следующие значения:

- 0x0 – нет ошибок;
- 0xF0 – сигнал недостоверен из-за ошибки модуля.

8.4.6. Модуль МДВ17-64

Модуль дискретного ввода МДВ17-64 имеет 64 канала ввода.

Модуль функционирует в режиме непрерывного опроса каналов дискретного ввода по сигналам встроенного таймера с пользовательской поканальной частотой выборки.

Модуль осуществляет фильтрацию входных данных по заданному алгоритму. Выбор алгоритма, а также активация фильтра доступна пользователю.

Окно настройки модуля МДВ17-64 имеет 64 канала дискретного ввода и обеспечивает следующие функции:

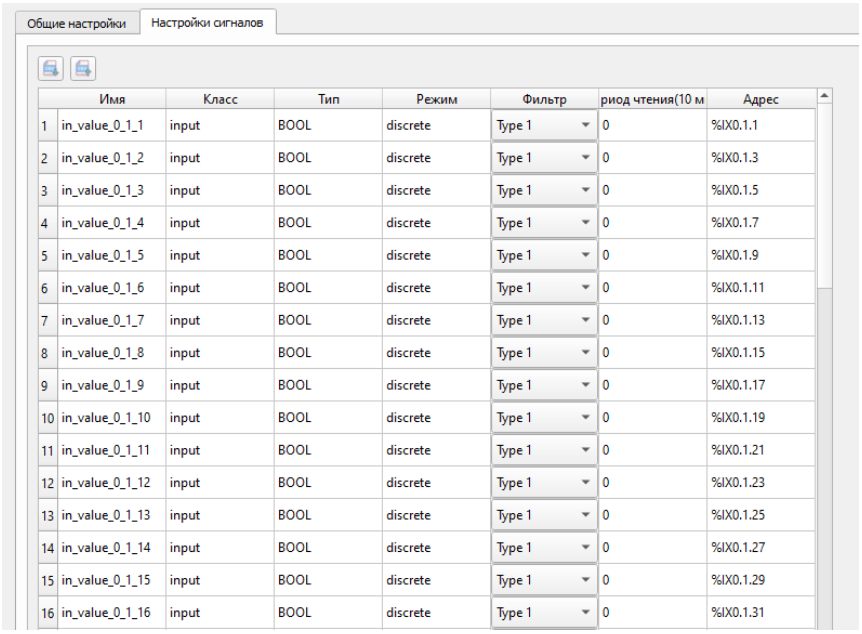
- настройку режима работы каналов;
- настройку синхронного ввода модуля;
- обеспечение связи с переменными IDE Эльбрус;
- установку сигналов качества каналов и обеспечение связи с переменными IDE Эльбрус.

На вкладке «Общие настройки» пользователь задает слот модуля в корзине контроллера.

Каждому каналу дискретного ввода по умолчанию присваиваются автоматически сгенерированные имена, которые могут изменяться пользователем. Тип данных сигнала дискретного ввода – BOOL . Каждому каналу соответствует дополнительный сигнал качества, отражающий текущий статус переменной дискретного ввода. Тип данных сигнала качества – DINT.

Сигнал может быть настроен индивидуально в соответствии с рисунком (см. Рисунок 87). По двойному щелчку по каналу в столбце «Имя» пользователь может изменить имя сигнала. Тип фильтра канала доступен для редактирования путем выбора желаемого значения в соответствующем выпадающем списке. Доступные фильтры: «No filter», «Type 1».

Для отладки и взаимодействия с сигналами дискретного ввода, а также получения статусных сигналов необходимо добавить сигналы в программный модуль IDE Эльбрус.



	Имя	Класс	Тип	Режим	Фильтр	риод чтения(10 м	Адрес
1	in_value_0_1_1	input	BOOL	discrete	Type 1	0	%IX0.1.1
2	in_value_0_1_2	input	BOOL	discrete	Type 1	0	%IX0.1.3
3	in_value_0_1_3	input	BOOL	discrete	Type 1	0	%IX0.1.5
4	in_value_0_1_4	input	BOOL	discrete	Type 1	0	%IX0.1.7
5	in_value_0_1_5	input	BOOL	discrete	Type 1	0	%IX0.1.9
6	in_value_0_1_6	input	BOOL	discrete	Type 1	0	%IX0.1.11
7	in_value_0_1_7	input	BOOL	discrete	Type 1	0	%IX0.1.13
8	in_value_0_1_8	input	BOOL	discrete	Type 1	0	%IX0.1.15
9	in_value_0_1_9	input	BOOL	discrete	Type 1	0	%IX0.1.17
10	in_value_0_1_10	input	BOOL	discrete	Type 1	0	%IX0.1.19
11	in_value_0_1_11	input	BOOL	discrete	Type 1	0	%IX0.1.21
12	in_value_0_1_12	input	BOOL	discrete	Type 1	0	%IX0.1.23
13	in_value_0_1_13	input	BOOL	discrete	Type 1	0	%IX0.1.25
14	in_value_0_1_14	input	BOOL	discrete	Type 1	0	%IX0.1.27
15	in_value_0_1_15	input	BOOL	discrete	Type 1	0	%IX0.1.29
16	in_value_0_1_16	input	BOOL	discrete	Type 1	0	%IX0.1.31

Рисунок 87 – Окно настройки модуля МДВ17-64

- Статусные сигналы каждого канала могут иметь следующие значения:
- 0x0 – нет ошибок;
- 0xF0 – сигнал недостоверен из-за ошибки модуля.

8.4.7. Модуль МАВ17-ТПТС

Модуль МАВ17-ТПТС предназначен для приема информации от 8 термопар и 8 термометров сопротивления. Модуль имеет встроенные градуировки для ряда подключаемых датчиков. Настройки режима работы каналов выполняются через среду разработки ELPLC-LOGIC.

На вкладке «Общие настройки» пользователь задает слот модуля в корзине контроллера, а также конфигурирует режим работы модуля в соответствие с рисунком (см. Рисунок 88).

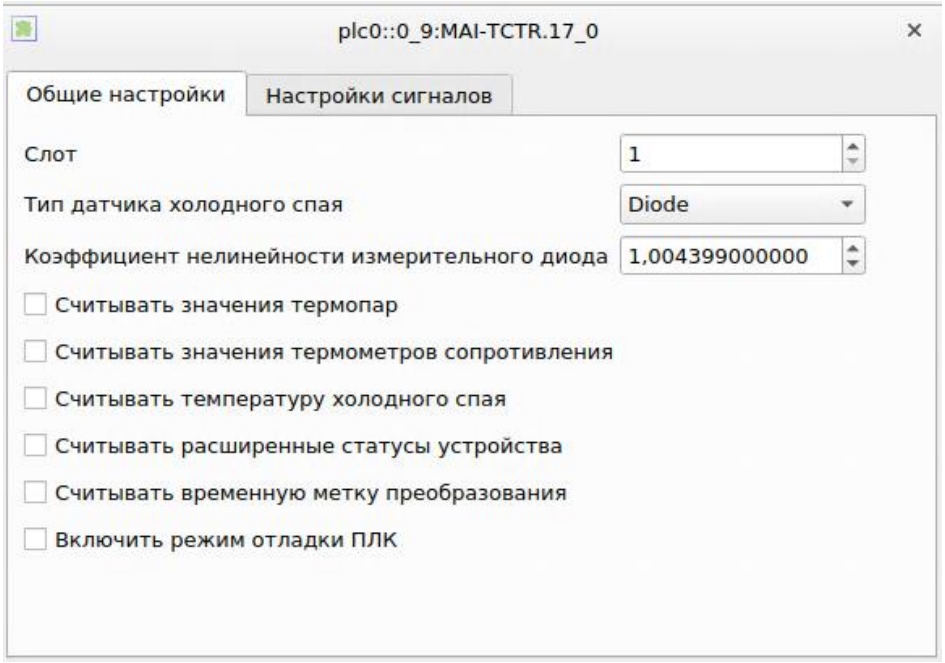


Рисунок 88 – Настройка модуля MAB17-TПТС

Окно настройки содержит следующие параметры:

Слот на шине – номер слота, в который установлен модуль.

Тип датчика холодного спая – указывается тип примененного датчика компенсации холодного спая.

Коэффициент нелинейности измерительного диода – указывается коэффициент нелинейности. Для компенсации холодного спая может быть применен измерительный диод. Каждый измерительный диод имеет коэффициент нелинейности, позволяющий повысить точность его показаний. Информацию о коэффициенте нелинейности должен предоставлять изготовитель измерительного диода. Информацию о нем также можно найти в этикетке на кроссовый модуль, поставляемый вместе с модулем MAB17-TПТС. В случае, если информация о коэффициенте недоступна, рекомендуется оставить значение по умолчанию: 1,004399.

Считывать значение термопар – этот пункт нужно отметить, если предполагается работа с термопарами. В противном случае измерения не будут считываться из модуля.

Считывать значения термометров сопротивления – этот пункт нужно отметить, если предполагается работа с термометрами сопротивления. В противном случае измерения не будут считываться из модуля.

Считывать температуру холодного спая – этот пункт нужно отметить, если требуется считывать в IEC переменные измеренную модулем температуру холодного спая, в соответствии с выбранным типом датчика.

Для получения детальной информации о статусе устройства необходимо отметить пункт «Считывать расширенные статусы устройства». В результате будет считываться 32-х разрядное число, содержащее следующие ошибки:

0 – нет ошибок

B0 – ошибка интерфейса 1

B1 – ошибка интерфейса 2

B2 – ошибка АЦП1

B3 – ошибка АЦП2

B4 – ошибка датчика ХС

B5-13 - зарезервированы

B14 – АЦП1 не инициализировано

B15 – АЦП2 не инициализировано

B16-31 – зарезервированы

Каждому измерению присваивается временная метка в тиках по 10 мкс. Отметив пункт «Считывать временную метку…» можно получить эти значения в переменные IEC.

Внимание! Важно отметить, что данные модуля MAB17-ТПТС достаточно объемные. Чтение полного комплекта данных потребует несколько тактов шины ELPLC-BUS. В связи с этим не рекомендуется отмечать все данные, если в них нет реальной необходимости в коде программы.

На вкладке «Настройка сигналов» указываются параметры для каждого измерительного канала. Окно представлено на рисунке (см. Рисунок 89).

Окно разделено на две части. В верхней части описываются настройки каналов термопар, в нижней настройки каналов термометров сопротивления.

Для каждой термопары настраиваются:

- тип датчика – градуировка по типу (режим прямого АЦП, J, K, E, N, R, S, T, B, L);
- контроль линии – аппаратный контроль присутствия датчика;

- способ компенсации холодного спая – HW – аппаратный, от датчика, подключенного к модулю или SW_CJ – программный, от переменной, задаваемой по другому датчику.

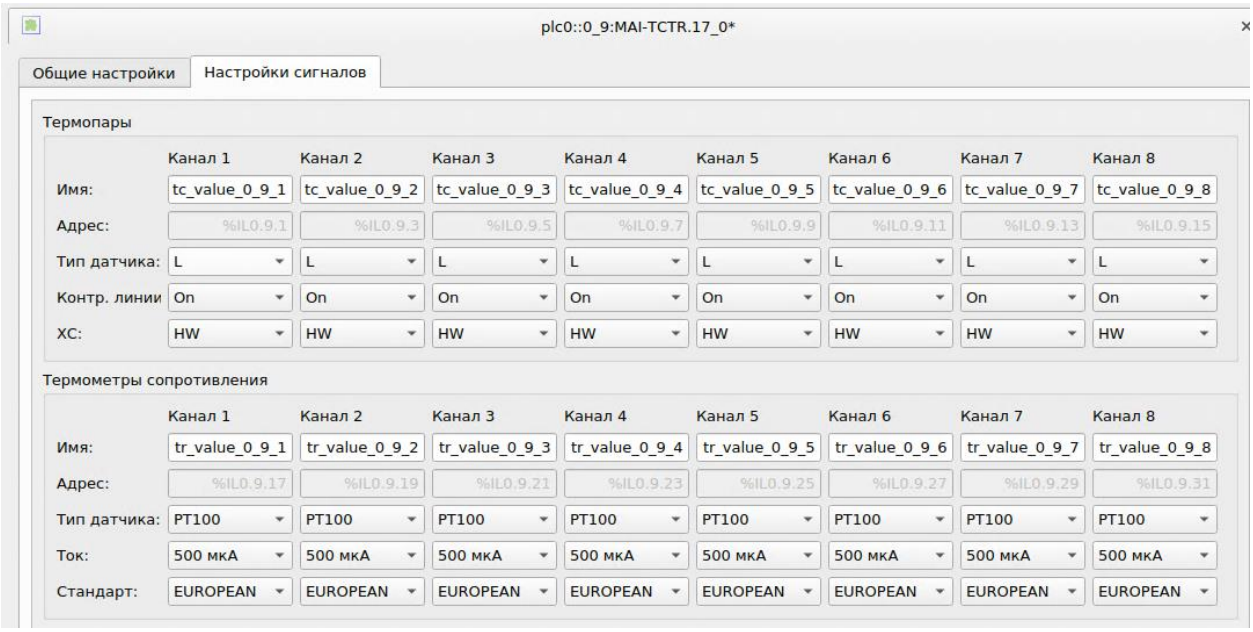


Рисунок 89 – Окно настройки каналов модуля MAB17-ТПТС

Для каналов термометров сопротивления пользователю доступны следующие настройки:

- тип датчика: PT10, PT50, PT100, PT200, PT500, PT1000, PT1000_375, NI1200, CUSTOM;
- ток: 5мкА, 10мкА, 25мкА, 50мкА, 100мкА, 250мкА, 500мкА, 1мА;
- стандарт: EUROPEAN, AMERICAN, JAPANESE, ITS_90.

Детальную информацию о настройках можно также получить из руководства на модуль.

8.5. Модуль протокола OPC UA (Server)

OPC UA – платформо независимая спецификация, выпущенная в 2008 году, определяющая передачу данных и взаимодействие устройств в промышленных сетях. Спецификация OPC UA объединяет все функции отдельных спецификаций OPC Classic в одну расширяемую структуру.

Основными особенностями реализации являюся:

- кроссплатформенность – сервер OPC UA может функционировать на любых устройствах, поддерживающих среду исполнения brztrte-runtime;
- безопасность – плагин имеет функции защиты сервера OPC UA от несанкционированного доступа с помощью логина\пароля и\или шифрованного соединения между клиентом и сервером с использованием сертификатов безопасности;
- функциональность – в рамках плагина реализованы следующие модели работы с данными: доступ и мониторинг оперативных данных, чтение архивов, резервированное подключение сервера в рамках общей системы резервирования ПЛК;
- удаленный доступ – сервер и клиент могут располагаться в разных сетях. Клиент OPC UA при подключении предоставляет доступные параметры сервера и список сигналов для управления и мониторинга пользователю.

В САПР ELPLC-LOGIC реализован OPC UA сервер.

Поддерживаются режимы работы:

- доступ к оперативным данным (профиль OPC DA);
- доступ к архивным данным (профиль OPC HA).

Реализована защита подключения к серверу с использованием логина\пароля и\или шифрованного подключения с использованием сертификатов безопасности.

Добавление модуля OPC UA сервера в проект производится в дереве проекта, в контекстном меню ПЛК (см. Рисунок 90).

После добавления модуля в проект соответсвующий узел появится в дереве. Для его настройки откройте окно этого узла.

Окно имеет две вкладки: «Общие настройки» и «Сигналы».

На вкладке «Общие настройки» указываются настройки модуля сервера и правила его исполнения. Окно настроек представлено на рисунке (см. Рисунок 91).

В окне имеются следующие элементы управления:

- архивировать значения – включение/выключение профиля архивирования;
- блок настроек системы архивирования;
- использовать аутентификацию – включение/выключение функции проверки логина/пароля при подключении;
- настройки логина и пароля;

- использовать шифрованное соединение – включение/выключение системы шифрования соединения.

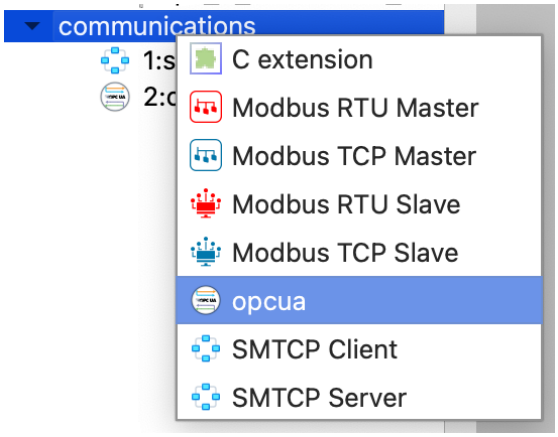


Рисунок 90 - Добавление OPC UA

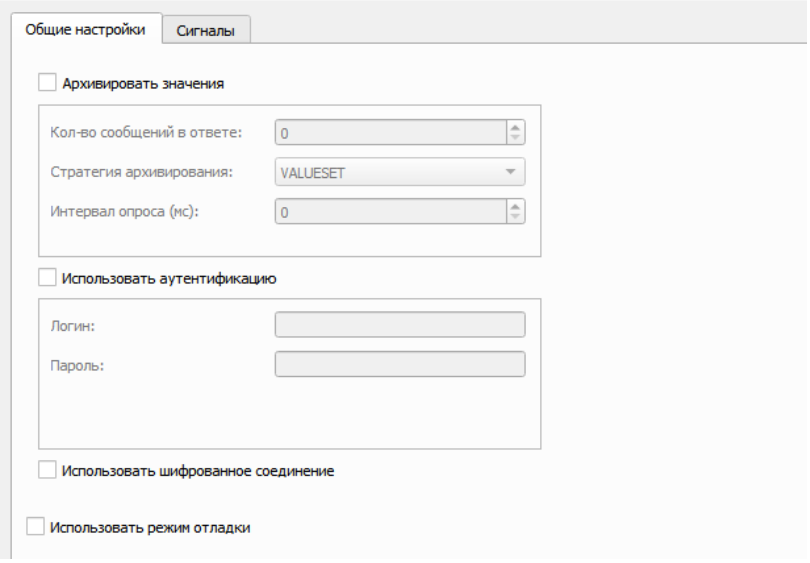


Рисунок 91 - Общие настройки OPC UA

8.5.1. Конфигурация сигналов OPC UA

На вкладке «Сигналы» пользователь задает конфигурацию сервера. В таблице слева пользователь добавляет узлы сервера, таблица справа отображает сигналы, добавленные в выбранный узел. Каждый добавленный сигнал, имеет уникальный идентификатор в среде исполнения на сервере. После выбора активного узла доступны следующие опции работы с переменными:

- добавление одного сигнала;

- добавление группы сигналов по шаблону;
- выбор всех сигналов;
- снятие выбора со всех сигналов;
- удаление выбранных сигналов;
- импорт сигналов из csv;
- экспорт сигналов в csv.

Для сигналов доступны следующие типы данных: BOOL, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL.

Сигналы могут быть импортированы из csv файла. Файл имеет следующие поля:

- Node – номер узла;
- Index - номер сигнала в узле;
- Name – название сигнала;
- DataType – номер типа данных (доступные номера: 0 -"BOOL", 1- "SINT", 2- "USINT", 3- "INT",4 - "UINT", 5- "DINT", 6- "UDINT",7- "LINT", 8- "ULINT", 9- "REAL", 10- "LREAL");
- Class – номер класса сигнала (0- input, 1- inout, 2- output);
- Description – описание сигнала. Необязательное поле;

При неверном файле конфигурации узла пользователю будут выдаваться ошибки.

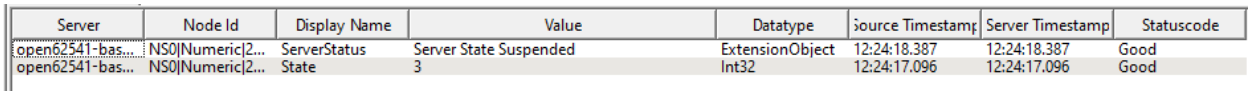
После конфигурации сервера при подключении клиент автоматически получает список сигналов доступных на сервере.

8.5.2. Признак статуса резервирования для OPC UA

Поскольку ПЛК под управлением САПР ELPLC-LOGIC может функционировать в резервированном режиме, для спецификации OPC UA реализован режим резервированного подключения. При работе в резервированном режиме, сервер одновременно запускает дубликат на резервном процессорном модуле. Если процессор, на котором запущен сервер, в настоящий момент имеет статус «PRIMARY» (используется как основной контроллер), то мониторинг узла «Server», в частности составной переменной «ServerStatus» показывает обобщенное состояние «Server State Running» и вложенный параметр «State» равняется 0, что означает, что сервер имеет достоверные актуальные данные (см. Рисунок 92).

Рисунок 92 - Отображение статуса основного сервера

При переходе ПЛК в состояние «Standby» (используется как резервный контроллер), OPC UA сервер изменяет обобщённое состояние составной переменной «ServerStatus» на «Server State Suspended» и параметр «State» становится равен 3 (Рисунок 92). В этом случае данные, приходящие с сервера считаются недостоверными.



Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode
open62541-bas...	NS0|Numeric|2...	ServerStatus	Server State Suspended	ExtensionObject	12:24:18.387	12:24:18.387	Good
open62541-bas...	NS0|Numeric|2...	State	3	Int32	12:24:17.096	12:24:17.096	Good

Рисунок 93 - Статус резервного модуля

8.5.3. Архивирование. Профиль OPC HA.

Архивирование значений (профиль OPC HA) позволяет настроить сохранение значений сигналов на период работы сервера. Группа настроек включает максимальное число сообщений для отображения пользователю и стратегию архивирования. Число сообщений в ответе может быть задано в диапазоне от 0 до 999999999. Это значит, что пользователю одновременно для мониторинга доступны заданное число сигналов в отслеживаемый момент времени. Доступные стратегии архивирования:

- VALUESET – значения сохраняются при установке узла;
- POLL – периодическая архивация.

Архивирование в режиме «VALUESET» осуществляется в режиме установки значений средой ELPCL-LOGIC. На каждом цикле publish средой исполнения осуществляется запись всех сигналов в архив. При использовании стратегии архивирования «VALUESET» архивирование будет происходить на каждом цикле среды исполнения вне зависимости от изменения значения, в такте программы ПЛК.

В режиме «POLL» отбор данных будет осуществляться автоматически, по таймеру, с циклом, заданным в общих настройках.

8.5.4. Безопасность, аутентификация пользователя и шифрование

Аутентификация пользователя активируется с использованием чекбокса «использовать аутентификацию». Для редактирования доступны текстовые поля «логин» и «пароль», которые используются клиентом для подключения к серверу. При подключении клиента к серверу в этом случае необходимо будет вводить логин/пароль указанный на сервере.

Чекбокс «Использовать шифрованное соединение» позволяет шифровать передаваемые сообщения по следующим политикам безопасности:

- Basic128Rsa15;
- Basic256;
- Basic256Sha265.

Для использования шифрованного соединения пользователю необходимо в папке со средой исполнения иметь файлы «server_cert.der» и «server_key.der». Генерация и передача сертификата и ключа будет реализована в последующих версиях ELPLC-LOGIC.

9. УКАЗАТЕЛЬ СОКРАЩЁННЫХ НАИМЕНОВАНИЙ И ОБОЗНАЧЕНИЙ

МЭК 61131-3 – раздел международного стандарта МЭК 61131 (также существует соответствующий европейский стандарт EN 61131), описывающий языки программирования для программируемых логических контроллеров.

Среда разработки для языков стандарта МЭК 61131-3 – система программных средств, используемая инженерами по автоматизации, для разработки прикладного программного обеспечения на высокоуровневых языках стандарта МЭК 61131-3 под различные целевые платформы, которая включает в себя:

- текстовые и графические редакторы языков стандарта IEC 61131-3;
- транслятор диаграмм графических языков в текстовый язык;
- транслятор текстового языка в язык С;
- механизмы плагинов для взаимодействия с модулями УСО;
- механизмы добавления компиляторов под целевую платформу;
- механизмы соединений с целевыми устройствами;
- отладчик.

Модули УСО – модули ввода/вывода, обеспечивающие подключение датчиков и исполнительных механизмов.

Целевое устройство – аппаратное средство с определённой архитектурой процессора, на котором могут исполняться различные исполняемые файлы, обращающиеся с помощью него к модулям УСО.

Прикладная программа (исполняемый файл) для целевого устройства – скомпилированный и скомпонованный so-файл, который будет выполняться на целевом устройстве.

Плагин для модуля УСО – интерфейс, состоящий из специальных драйверов и элементов пользовательского интерфейса для ПО ELPLC-LOGIC, позволяющий связывать переменные модулей УСО с переменными программных модулей, из которых состоит проект.

Проект – совокупность программных модулей (программ, функциональных блоков, функций), плагинов внешних модулей УСО, ресурсов, пользовательских типов данных, сборка (компиляция и компоновка) которых, представляет собой прикладную программу для целевого устройства. Каждый проект сохраняется в отдельном файле.

Переменная – область памяти, в которой находятся данные, с которыми оперирует программный модуль.

Ресурс – элемент, отвечающий за конфигурацию проекта: глобальные переменные и экземпляры проекта, связываемыми с программными модулями типа «Программа» и задачами.

Программный модуль – элемент, представляющий собой функцию, функциональный блок или программу. Каждый программный модуль состоит из раздела объявлений и кода. Для написания всего кода программного используется только один из языков программирования стандарта МЭК 61131-3.

Функция – программный модуль, который возвращает только единственное значение, которое может состоять из одного и нескольких элементов (если это битовое поле или структура).

Функциональный блок – программный модуль, который принимает и возвращает произвольное число значений, а также позволяет сохранять своё состояние (подобно классу в различных объектно-ориентированных языках). В отличие от функции функциональный блок не формирует возвращаемое значение.

Программа – программный модуль, представляющий собой единицу исполнения, как правило, связывается (ассоциируется) с задачей.

Задача – элемент, представляющий время и приоритет выполнения программного модуля типа «Программа» в рамках экземпляра проекта.

Экземпляр – представляет собой программу, как единицу исполнения, связанную (ассоциированную) с определённой задачей. Так же, как экземпляр, рассматриваются переменные, определённые в программных модулях: программа и функциональный блок.

Пользовательский тип данных – тип данных, добавленный в проект и представляющий собой: псевдоним существующего типа, поддиапазон существующего типа, перечисление, массив или структуру.

FBD - Function Block Diagram (Функциональный блоковые диаграммы) – Графический язык; программа создаётся путём соединения множества функциональных блоков

IL - Instruction List (Список инструкций) – Аппаратно-независимый низкоуровневый ассемблероподобный язык

LD - Ladder Diagram (Релейно-контактные схемы) – Графический язык; представляет собой программную реализацию электрических схем на базе электромагнитных реле

SFC - Sequential Function Chart (Последовательностные функциональные диаграммы) – Графический высокоуровневый язык; создан на базе математического аппарата сетей Петри описывает последовательность состояний и условий переходов

ST - Structured Text (Структурированный текст) – Текстовый Паскалеподобный язык программирования

ВУ - Верхний уровень

НУ - Нижний уровень

ОС - Операционная система

ПЛК - Программируемый логический контроллер

ПО - Программное обеспечение

ППО - Прикладное программное обеспечение

ПТК - Программно-технический комплекс

САПР - Система автоматизированного проектирования

ТС - Технические средства

ПРИЛОЖЕНИЕ

Редактор кода САПР ELPLC-LOGIC (ЛЯЮИ.00707-01 92 01).

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц) в документе	Номер документа	Входящий номер сопроводительного документа и дата	Подпись	Дата
Изм.	измененных	замененных	новых	аннулированных					