

**УТВЕРЖДЕН**

**ИМЕС.00390-03 33 01-ЛУ**

**Пакет инструментальных средств ПЛК Fastwel**  
**Fastwel PLC Application Toolkit**  
**Руководство пользователя**  
**ИМЕС.00390-03 33 01**  
Версия 1.0

## АННОТАЦИЯ

Настоящий документ содержит указания по установке и применению программного обеспечения "Пакет инструментальных средств ПЛК Fastwel" (Fastwel PLC Application Toolkit) для разработки прикладного программного обеспечения для программируемых контроллеров Fastwel.

### Торговые марки

ДОЛОМАНТ™, ФАСТВЕЛ™, Fastwel™ – официально зарегистрированные торговые марки ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ», Москва, Российская Федерация.

Кроме того, настоящий документ может содержать наименования, фирменные логотипы и торговые марки, являющиеся зарегистрированными торговыми марками, а следовательно, права собственности на них принадлежат их законным владельцам.

### Права собственности

Настоящий документ содержит информацию, которая является интеллектуальной собственностью ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ».

Владельцем прав на объекты интеллектуальной собственности программного обеспечения CODESYS V3 является компания CODESYS GmbH, Memminger Straße 151, 87439 Kempten Germany.

Владельцем прав на объекты интеллектуальной собственности программного обеспечения Astra.IDE является компания ООО "ПРОСОФТ-СИСТЕМЫ", 620102, Свердловская область, г. Екатеринбург, Волгоградская ул., стр 194а.

Настоящий документ не может быть скопирован или передан с использованием известных средств, а также не может храниться в системах хранения и поиска информации без предварительного письменного согласия ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ» или одного из ее уполномоченных агентов. Информация, содержащаяся в настоящем документе, насколько нам известно, не содержит ошибок, однако, ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ» не может принять на себя ответственность за какие-либо неточности и их последствия, а также ответственность, возникающую в результате использования или применения любой схемы, продукта или примера, приведенного в настоящем документе. ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ» оставляет за собой право изменять и усовершенствовать как настоящий документ, так и представленный в нем продукт по своему усмотрению без дополнительно извещения.

### Контактная информация

Изготовитель – ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ»:

Почтовый адрес: Россия, 117342, Москва, ул. Введенского, д.3

Телефон: +7 (495) 232-2033

Факс: +7 (495) 232-1654

Электронная почта: [info@dolomant.ru](mailto:info@dolomant.ru)

Web: <http://www.dolomant.ru>

Служба технической поддержки:

Телефон: +7 (495) 232-1698

Электронная почта: [fio@fastwel.ru](mailto:fio@fastwel.ru)  
[support@fastwel.ru](mailto:support@fastwel.ru)

### Авторское право

Настоящий документ не может быть скопирован, воспроизведен, переведен или конвертирован в любую электронную или машиночитаемую форму без предварительного письменного разрешения ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ».

## СОДЕРЖАНИЕ

<b>1.</b>	<b>ВВЕДЕНИЕ</b>	<b>7</b>
1.1.	ИНФОРМАЦИЯ О ДОКУМЕНТЕ	7
1.2.	ПРИНЯТЫЕ СОКРАЩЕНИЯ И ТЕРМИНЫ	7
1.3.	ПЕРЕЧЕНЬ ССЫЛОЧНЫХ ДОКУМЕНТОВ	9
1.4.	ПЕРЕЧЕНЬ НОРМАТИВНЫХ ДОКУМЕНТОВ	9
1.5.	ЭЛЕМЕНТЫ ОФОРМЛЕНИЯ ТЕКСТА	10
1.6.	ИСПОЛЬЗУЕМЫЕ ПРЕДУПРЕЖДАЮЩИЕ И ИНФОРМАЦИОННЫЕ ЗНАКИ	10
1.7.	ТИПЫ ДАННЫХ И ПРЕДСТАВЛЕНИЕ ЧИСЛОВЫХ ЗНАЧЕНИЙ	10
<b>2.</b>	<b>ОБЩИЕ СВЕДЕНИЯ</b>	<b>12</b>
2.1.	НАЗНАЧЕНИЕ FASTWEL PLC APPLICATION TOOLKIT	12
2.2.	СТРУКТУРА И СОСТАВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	12
2.2.1.	Общие сведения	12
2.2.2.	Системные компоненты	13
2.2.3.	Драйвер сервисного порта USB	13
2.2.4.	Пакет расширения среды разработки IDE МЭК 61131-3	13
2.2.5.	Примеры программирования	13
2.2.6.	Документация	14
2.2.7.	IDE МЭК 61131-3 CODESYS V3	14
2.2.8.	Перечень файлов Fastwel PLC Application Toolkit	14
2.3.	СИСТЕМНЫЕ ТРЕБОВАНИЯ К РАБОЧЕМУ МЕСТУ РАЗРАБОТЧИКА	18
2.3.1.	Требования к аппаратным средствам	18
2.3.2.	Требования к системному программному обеспечению	19
<b>3.</b>	<b>УСТАНОВКА FASTWEL PLC APPLICATION TOOLKIT И НАСТРОЙКА IDE МЭК 61131-3</b>	<b>21</b>
3.1.	ПРЕДВАРИТЕЛЬНЫЕ ЗАМЕЧАНИЯ	21
3.2.	УСТАНОВКА FASTWEL PLC APPLICATION TOOLKIT	22
3.2.1.	Установка базового варианта FastwelPLCApplicationToolkit.exe	22
3.2.2.	Установка расширенного варианта Fastwel PLC Application Toolkit	25
3.3.	РЕКОМЕНДАЦИИ ПО УСТАНОВКЕ IDE МЭК 61131-3	27
3.3.1.	Установка CODESYS V3	27
3.3.2.	Установка Astra.IDE	29
3.4.	ПРОВЕРКА УСТАНОВКИ FASTWEL PLC APPLICATION TOOLKIT	33
3.4.1.	Общие сведения	33
3.4.2.	Проверка установки Fastwel PLC Application Toolkit с CODESYS V3	33
3.4.3.	Проверка установки Fastwel PLC Application Toolkit с Astra.IDE	35
3.5.	НАСТРОЙКА ПАРАМЕТРОВ СВЯЗИ МЕЖДУ IDE МЭК 61131-3 И КОНТРОЛЛЕРОМ	38
3.5.1.	Общие сведения	38
3.5.2.	Настройка сервиса IDE Gateway для работы через сервисный USB-порт или порт интерфейса RS-232C контроллера	39
3.5.3.	Настройка сервиса IDE Gateway для работы по сети Ethernet	46
3.5.3.1.	Общие сведения	46
3.5.3.2.	Связь с контроллерами через TCP-интерфейс	47
3.5.3.3.	Связь с контроллерами через UDP-интерфейс	49
3.6.	НАСТРОЙКА ПАРАМЕТРОВ СРЕДЫ РАЗРАБОТКИ	52
3.6.1.	Общие сведения	52
3.6.2.	Изменение языка пользовательского интерфейса	52
3.6.3.	Интеллектуальный ввод	53
3.6.4.	Мониторинг переменных	53
3.6.5.	Управление содержимым главного окна среды разработки при запуске	55
3.6.6.	Автоматическое и резервное сохранение проекта во время работы	55
3.6.7.	Размещение вложенных вкладок редактора устройств	55
3.6.8.	Управление отображением координатной сетки в формах визуализации	57
3.7.	НАСТРОЙКА ПАРАМЕТРОВ ПРОЕКТА	58
3.7.1.	Общие сведения	58
3.7.2.	Параметры компиляции	58
3.7.3.	Параметры загрузки исходного кода проекта в контроллер	59
3.7.4.	Защита файла проекта от несанкционированного доступа	60
<b>4.</b>	<b>РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ КОНТРОЛЛЕРОВ FASTWEL В IDE МЭК 61131-3</b>	<b>62</b>
4.1.	ОБЩИЕ СВЕДЕНИЯ	62
4.2.	ПРОГРАММНАЯ МОДЕЛЬ ПРИЛОЖЕНИЯ	62
4.2.1.	Общие сведения	62
4.2.2.	Проект IDE МЭК 61131-3	62

4.2.3.	Программы, функциональные блоки и функции .....	63
4.2.4.	Задачи.....	65
4.2.5.	Пользовательские типы данных (DUT) .....	67
4.2.6.	Символьная конфигурация.....	67
4.2.7.	Глобальные переменные.....	71
4.2.8.	Энергонезависимые переменные.....	72
4.2.9.	Сетевые переменные .....	75
4.2.10.	Неподдерживаемые элементы программной модели приложения.....	79
4.3.	ОБМЕН ДАННЫМИ С ВНЕШНИМИ УСТРОЙСТВАМИ.....	80
4.3.1.	Связь приложения с окружением .....	80
4.3.2.	Ввод и вывод данных.....	84
4.3.3.	Режимы ввода-вывода.....	86
4.3.4.	Мониторинг значений переменных, соотнесенных с %I-адресами во входном образе процесса.....	88
4.4.	ВЫПОЛНЕНИЕ ЗАДАЧ ПРИЛОЖЕНИЯ .....	90
4.4.1.	Общие сведения .....	90
4.4.2.	Циклические задачи.....	92
4.4.3.	Ациклические задачи типа Событие .....	93
4.4.4.	Ациклические задачи типа Статус .....	94
4.4.5.	Свободно-исполняемые задачи.....	95
4.5.	ОБРАБОТКА СИСТЕМНЫХ СОБЫТИЙ .....	96
4.5.1.	Общие сведения .....	96
4.5.2.	Перечень поддерживаемых системных событий.....	98
4.5.3.	Порядок вызова обработчиков системных событий.....	99
4.6.	ЗАГРУЗКА ПРИЛОЖЕНИЯ В КОНТРОЛЛЕР .....	100
4.6.1.	Способы загрузки приложения в контроллер.....	100
4.6.2.	Полная загрузка приложения .....	102
4.6.3.	Загрузка онлайн-изменений.....	103
4.6.4.	Множественная загрузка приложений в несколько контроллеров.....	105
4.7.	СЕРВЕР OPC UA .....	107
4.7.1.	Общие сведения .....	107
4.7.2.	Активация и настройка параметров сервера OPC UA.....	108
4.7.3.	Доступ сервера OPC UA к переменным приложения.....	109
4.7.4.	Соединение клиента с сервером OPC UA .....	111
4.7.5.	Обзор адресного пространства и обмен данными с сервером.....	114
4.7.6.	Общие сведения о безопасном взаимодействии клиентов с сервером OPC UA.....	117
4.7.7.	Аутентификация пользователя.....	117
4.7.8.	Механизм цифровой подписи и шифрования OPC UA .....	118
4.7.9.	Разрешение защищенных сеансов связи с сервером OPC UA.....	119
4.7.10.	Просмотр и генерация сертификатов безопасности в веб-конфигураторе контроллера .....	120
4.7.11.	Просмотр и генерация сертификатов безопасности в IDE МЭК 61131-3.....	122
4.7.12.	Настройка защищенного сеанса связи в клиентском приложении OPC UA .....	124
4.7.13.	Обмен сертификатами клиента с сервером и создание сеанса связи через защищенный канал .....	124
4.7.14.	Развертывание приложения с сохранением возможности безопасного подключения клиентов OPC UA.....	127
5.	СЕРВИСНЫЕ ОПЕРАЦИИ .....	129
5.1.	ОБЩИЕ СВЕДЕНИЯ .....	129
5.2.	УПРАВЛЕНИЕ СЪЕМНЫМИ ДИСКОВЫМИ НАКОПИТЕЛЯМИ .....	129
5.2.1.	Общие сведения .....	129
5.2.2.	Просмотр информации о подключенных съемных накопителях.....	130
5.2.3.	Извлечение съемных дисковых накопителей .....	130
5.3.	УПРАВЛЕНИЕ ДОСТУПОМ .....	131
5.3.1.	Учетные записи пользователей.....	131
5.3.2.	Изменение пароля учетной записи из IDE МЭК 61131-3.....	132
5.3.3.	Ограничение прав доступа для учетных записей пользователей.....	135
5.4.	ЧТЕНИЕ И ЗАПИСЬ ФАЙЛОВ.....	138
5.4.1.	Способы передачи файлов между контроллером и компьютером.....	138
5.4.2.	Передача файлов по протоколу FTP.....	138
5.4.2.1.	Передача файлов проводником Windows .....	138
5.4.2.2.	Передача файлов утилитами командной строки Cygwin .....	139
5.4.2.3.	Передача файлов по протоколу SFTP .....	140
5.4.3.	Передача файлов в IDE МЭК 61131-3.....	142
5.5.	РАЗВЕРТЫВАНИЕ ПРИЛОЖЕНИЙ.....	143
5.5.1.	Общие сведения .....	143
5.5.2.	Операции процесса развертывания.....	144
5.5.2.1.	Формирование файла развертывания на эталонном контроллере.....	144

5.5.2.2.	Развертывание с карты microSD.....	145
5.5.2.3.	Развертывание с USB-накопителя.....	145
5.5.2.4.	Развертывание с компьютера .....	146
5.6.	ОБНОВЛЕНИЕ СИСТЕМНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ .....	146
5.7.	ОЦЕНКА ЗАГРУЗКИ ПРОЦЕССОРА .....	147
<b>6.</b>	<b>СИСТЕМНЫЕ БИБЛИОТЕКИ.....</b>	<b>150</b>
6.1.	ОБЩИЕ СВЕДЕНИЯ .....	150
6.2.	БИБЛИОТЕКИ СИСТЕМНЫХ ТИПОВ CmpErrors и ISysTypes .....	152
6.3.	БИБЛИОТЕКА FASTWELBOARD .....	153
6.3.1.	Общие сведения .....	153
6.3.2.	Типы данных .....	154
6.3.2.1.	BoardLeds_t .....	154
6.3.2.2.	BootReason_t .....	154
6.3.2.3.	LedsColor_t.....	154
6.3.3.	Функции.....	155
6.3.3.1.	SysBoardReadSwitches.....	155
6.3.3.2.	SysBoardReadSwitch .....	156
6.3.3.3.	SysBoardSetUserLed.....	156
6.3.3.4.	SysBoardReboot.....	157
6.3.3.5.	SysBoardGetBootReason .....	158
6.3.4.	Функциональные блоки .....	159
6.3.4.1.	SWITCH_READER_x10.....	159
6.3.4.2.	USER_LED_CONTROL .....	160
6.3.4.3.	USER_LEDS_CONTROL.....	161
6.4.	БИБЛИОТЕКА FASTWELCORE .....	161
6.4.1.	Общие сведения .....	161
6.4.2.	Типы данных .....	161
6.4.2.1.	F_PLCTL_RESULT .....	161
6.4.2.2.	F_RESET_MODE.....	162
6.4.2.3.	F_NETWORK_IFADDR .....	162
6.4.2.4.	F_NETWORK_MODE.....	162
6.4.2.5.	F_LINK_DESCRIPTOR.....	163
6.4.2.6.	F_LINK_RESULT .....	163
6.4.2.7.	F_TASK_INFO .....	164
6.4.3.	Функции.....	164
6.4.3.1.	FwPlatformGetSerialNumber.....	164
6.4.3.2.	FwPlatformReset.....	165
6.4.3.3.	F_Net_getAdaptersCount .....	166
6.4.3.4.	F_Net_getInterfacesCount.....	167
6.4.3.5.	F_Net_getIpInfo .....	167
6.4.3.6.	F_IecTasks_getCount.....	169
6.4.3.7.	F_IecTasks_getInfo .....	169
6.4.3.8.	F_IecTasks_linkVariables .....	170
6.4.3.9.	FwCheckSum16 .....	172
6.4.3.10.	FwCheckSum32 .....	173
6.4.3.11.	FwCRC16.....	173
6.4.3.12.	FwCRC32.....	174
6.4.3.13.	FwSerialNumberToString .....	174
6.4.4.	Функциональные блоки .....	175
6.4.4.1.	CALL_PERIOD_GETTER.....	175
6.4.4.2.	RETAIN_VAR_ENGINE .....	176
6.5.	БИБЛИОТЕКИ ДОСТУПА К ФАЙЛАМ И КАТАЛОГАМ SysFile и SysDir .....	177
6.5.1.	Общие сведения .....	177
6.5.2.	Рекомендации по использованию библиотеки SysFile .....	181
6.5.2.1.	Общие сведения.....	181
6.5.2.2.	Проверка наличия или отсутствия файла по заданному пути .....	182
6.5.2.3.	Режимы доступа к файлу .....	182
6.5.2.4.	Чтение и запись данных.....	184
6.5.2.5.	Закрытие файлов .....	185
6.5.3.	Рекомендации по использованию библиотеки SysDir .....	185
6.6.	БИБЛИОТЕКА SYSCOM.....	187
6.6.1.	Общие сведения .....	187
6.6.2.	Начало и завершение работы с последовательным портом.....	187
6.6.3.	Обмен данными через последовательный порт.....	190
6.7.	БИБЛИОТЕКА FASTWELREMOVABLEMEDIA.....	194
6.7.1.	Общие сведения .....	194
6.7.2.	Типы данных .....	194
6.7.2.1.	MountDisk.....	194
6.7.2.2.	MountDiskInfo .....	194
6.7.2.3.	MountPoint.....	195

---

6.7.2.4.	EVTPARAM_BoardDeviceNotify .....	195
6.7.3.	Функции .....	197
6.7.3.1.	SysBoardDeviceGetAllDisks .....	197
6.7.3.2.	SysBoardDeviceGetDiskInfo .....	198
6.7.3.3.	SysBoardDeviceGetAllMountPoints .....	199
6.7.3.4.	SysBoardDeviceEject .....	199
<b>ПРИЛОЖЕНИЕ 1 . ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ.....</b>		<b>201</b>

## 1. Введение

### 1.1. Информация о документе

Настоящий документ содержит указания по установке, настройке и применению программного обеспечения "Пакет инструментальных средств ПЛК Fastwel" (Fastwel PLC Application Toolkit) для разработки прикладного программного обеспечения (далее – *приложений*) для контроллеров программируемых, выпускаемых под торговой маркой Fastwel, на языках программирования, определенных стандартом ГОСТ Р МЭК 61131-3 (IEC 61131-3), в интегрированных средах разработки, совместимых с Fastwel PLC Application Toolkit.

Данный документ предназначен для инженеров-программистов автоматизированных систем управления технологическими процессами (далее – АСУТП), имеющим навыки программирования на языках стандарта ГОСТ Р МЭК 61131-3 (IEC 61131-3) и знакомыми с операционной системой Windows на уровне, достаточном для квалифицированного использования.

### 1.2. Принятые сокращения и термины

В настоящем документе используются следующие сокращения и термины:

Astra.IDE	инструментальное программное обеспечение (интегрированная среда разработки) версии 1.7.0.0 и выше компании Prosoft-Systems Ltd., предназначенное для разработки приложений контроллеров, программируемых на языках стандарта IEC 61131-3 (ГОСТ Р МЭК 61131-3)
CDC	Communications Device Class, класс устройств последовательной передачи данных через интерфейс USB
CODESYS V3	инструментальное программное обеспечение (интегрированная среда разработки) версии 3.5 и выше компании CODESYS Development GmbH, предназначенное для разработки приложений контроллеров, программируемых на языках стандарта IEC 61131-3 (ГОСТ Р МЭК 61131-3)
DHCP	Dynamic Host Configuration Protocol, протокол динамического назначения IP-адресов
DNS	Domain Name System, система доменных имен, как правило используемая для получения IP-адреса узла сети по имени
FTP	File Transfer Protocol, протокол передачи файлов по сети
HTTP	Hypertext Transfer Protocol, гипертекстовый протокол передачи данных по сети
IDE МЭК 61131-3	интегрированная среда разработки на языках стандарта IEC 61131-3 (ГОСТ Р МЭК 61131-3), совместимая с CODESYS V3 и/или Astra.IDE
IDE Gateway	коммуникационный сервис, обеспечивающий взаимодействие между IDE МЭК 61131-3 и контроллерами для загрузки и отладки приложений, передачи файлов и выполнения диагностических операций
MRAM	Magnetoresistive Random-Access Memory – энергонезависимая память с произвольным доступом на основе спиновых вентилей
NTP	Network Time Protocol, протокол сетевого времени для синхронизации внутренних часов на узлах сети с переменной латентностью
OPC UA	Open Platform Communications Unified Architecture – спецификация протокола информационного обмена между устройствами в промышленных сетях, разработанная организацией OPC Foundation

POU	<p>Program Organization Unit, программная единица или программный компонент по ГОСТ Р МЭК 61131-3 – элемент программной модели, представляющий обособленный именованный набор данных и операций по их преобразованию.</p> <p>Стандартом определены четыре вида POU: программа (PROGRAM), функциональный блок (FUNCTION_BLOCK), функция (FUNCTION) и класс (CLASS).</p>
PTP	Precision Time Protocol, протокол точного времени для синхронизации внутренних часов на узлах сети
ST	Structured Text, язык структурированного текста – текстовый язык реализации приложений ПЛК согласно IEC 61131-3 (ГОСТ Р МЭК 61131-3)
LD	Ladder Diagram, язык релейно-контактных схем (диаграмм) – графический язык реализации приложений ПЛК согласно IEC 61131-3 (ГОСТ Р МЭК 61131-3), основными примитивами которого являются графические представления катушек и контактов реле.
FBD	Function Block Diagram, язык диаграмм функциональных блоков – графический язык реализации приложений ПЛК согласно IEC 61131-3 (ГОСТ Р МЭК 61131-3), основными примитивами которого являются графические представления функциональных блоков, имеющих входы, выходы, внутренние переменные состояния и, как минимум, функцию, вызываемую при вызове экземпляра блока, оперирующую значениями на входах, переменных состояния и формирующую значения выходов.
SFC	Sequential Function Chart, язык последовательных функциональных схем – графический язык реализации приложений ПЛК согласно IEC 61131-3 (ГОСТ Р МЭК 61131-3), основными примитивами которого являются графические представления шагов, действий, переходов и условий переходов.
АСУТП, АСУ ТП	автоматизированная система управления технологическим процессом
КП	контроллер программируемый универсальный – вычислительное устройство, входящее в состав программируемого (логического) контроллера с переменным составом модулей и выполняющее функции модуля центрального процессора
ПЛК, контроллер	совокупность контроллера программируемого, объединенного с периферийными модулями межмодульной шиной, образующая многофункциональный многоканальный программируемый контроллер с переменным составом модулей, который по структуре и назначению относится к программируемым контроллерам по ГОСТ Р МЭК 61131-1
ПК	персональный компьютер, портативный компьютер (ноутбук), рабочая станция
ПТК	комплекс программно-технический – изделия, не соединенные друг с другом при изготовлении, и программное обеспечение, поставляемые потребителю и относящиеся к семейству продуктов, объединяемых одним или несколькими конструктивно-техническими признаками, принципами работы и правилами применения
ППО, приложение	прикладное программное обеспечение, реализующее специфические пользовательские алгоритмы сбора, обработки данных и управления технологическим объектом управления
Профиль (среды разработки)	именованный набор программных компонентов, загружаемых при запуске среды разработки
РЭ	руководство по эксплуатации



СПО	системное программное обеспечение, микропрограмма – встроенное программное обеспечение вычислительного или периферийного устройства
Система исполнения приложений МЭК 61131-3	часть СПО контроллеров Fastwel, обеспечивающая выполнение кода ППО, загруженного в ПЛК, взаимодействие ППО с периферийными модулями ПЛК и обмен данными по сети с использованием поддерживаемых протоколов прикладного уровня
ЯП	язык программирования

### 1.3. Перечень ссылочных документов

При работе с настоящим документом следует также пользоваться указаниями документов, перечисленных в таблице 1.

Данные документы находятся на дисковом накопителе в комплекте поставки контроллеров программируемых, контроллеров узла сети и модулей интерфейсных NIM745, а также доступны для загрузки на ftp-сервере производителя.

Для загрузки наиболее актуальной редакции документа с ftp-сервера производителя воспользуйтесь любым приложением, поддерживающим протокол ftp (FileZilla, Total Commander и т.п.), или *Проводником* Windows. При использовании *Проводника* Windows следует скопировать путь к файлу документа на ftp-сервере, указанный справа от поля "путь" в соответствующей ячейке таблицы 1 в адресную строку *Проводника*, нажать Enter, после чего скопировать на диск компьютера файл с именем, указанным в поле "файл" в соответствующей ячейке таблицы 1.

**Таблица 1 – Перечень ссылочных документов**

Обозначение	Наименование, ссылка на ресурс
ИМЕС.00300-03 33 02-1	Контроллер программируемый CPM723-01. Руководство по конфигурированию и программированию
	путь: <a href="ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: CPM723-01_CDSV3_UM.pdf
ИМЕС.00300-03 33 02-2	Контроллер программируемый универсальный CPM810-03. Руководство по конфигурированию и программированию
	путь: <a href="ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: CPM810-03_CDSV3_UM.pdf
ИМЕС.00300-03 33 01	Модули ввода-вывода. Руководство программиста
	путь: <a href="ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: FBUS_CDSV3_UM.pdf
ИМЕС.00300-03 33 03	Протокол MODBUS. Руководство по конфигурированию и программированию
	путь: <a href="ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: MODBUS_CDSV3_UM.pdf
ИМЕС.00300-03 33 04	Протокол ГОСТ Р МЭК 60870-5-104. Руководство по конфигурированию и программированию
	путь: <a href="ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: IEC60870-5-104_CDSV3_UM.pdf
ФАПИ.421459.700РЭ	Распределенная система ввода-вывода Fastwel I/O. Руководство по эксплуатации
	путь: <a href="ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version2/Doc/">ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version2/Doc/</a> файл: FIO_UM.pdf
ИМЕС.421459.252РЭ	Комплекс программно-технический Fastwel I/O-2. Руководство по эксплуатации. Часть 1
	путь: <a href="ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: FIO-2_UM1.pdf
ИМЕС.421459.252РЭ1	Комплекс программно-технический Fastwel I/O-2. Руководство по эксплуатации. Часть 2
	путь: <a href="ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Doc/">ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Doc/</a> файл: FIO-2_UM2.pdf

### 1.4. Перечень нормативных документов



В настоящем документе даны ссылки на нормативные документы согласно таблице 2.

Обозначение	Наименование документа	Номер пункта
ГОСТ Р МЭК 60870-5-104-2004	УСТРОЙСТВА И СИСТЕМЫ ТЕЛЕМЕХАНИКИ. Часть 5. Протоколы передачи. Раздел 104. Доступ к сети для ГОСТ Р МЭК 870-5-101 с использованием стандартных транспортных профилей	п. 1.3, 2.2.8, 6.1
ГОСТ Р МЭК 61131-1-2016	Контроллеры программируемые. Часть 1: Общая информация	п. 1.2
ГОСТ Р МЭК 61131-3-2016 (IEC 61131-3:2013)	Контроллеры программируемые. Часть 3: Языки программирования	п. 1.1, 1.2, 1.7, 2.2.7, 4.1, 4.2.1, 4.2.2, 4.3.1, 4.7.5, 6.1, 6.3.3.2, 6.3.3.3

В настоящем документе используются следующие элементы дополнительного оформления текста:

Текст	Понятие, значение параметра, команда, информационный элемент.
Текст	Название элемента пользовательского интерфейса (заголовок элемента управления, меню, окна и т.п.), заголовок рисунка или таблицы.
Текст	Фрагмент исходного текста программы, консольной команды или результата выполнения команды.
// Текст	Комментарий в исходном тексте программы.
Текст	Текст в ячейке таблицы.

Для привлечения внимания пользователя в настоящем документе применены следующие предупреждающие и информационные знаки:

	<p>Данный знак обращает внимание на необходимость следовать предупреждаемым им указаниям во избежание повреждения оборудования, потери данных или результатов ранее выполненной работы.</p>
	<p>Данный знак отмечает важную информацию, на которую следует обратить внимание.</p>

Для представления элементов программной модели устройств и переменных величин в настоящем руководстве используются типы данных по ГОСТ Р МЭК 61131-3, перечисленные в таблице 3.

Числовые значения в десятичной, двоичной и шестнадцатеричной системах исчисления представляются в виде числовых литералов согласно ГОСТ Р МЭК 61131-3. Например, значение 4294967295 в разных системах исчисления может быть представлено следующими способами:

шестнадцатеричное представление	<i>16#FFFFFFFF или 16#FFFF_FFFF</i>
двоичное представление	<i>2#1111_1111_1111_1111_1111_1111_1111_1111</i> или <i>2#1111111111111111 1111111111111111</i>

[illegible]

**Таблица 3 – Типы данных для представления элементов программной модели**

Тип	Диапазон	Длина, бит	Описание
BOOL	FALSE (0), TRUE (1)	1 <sup>1</sup>	Логический тип
BYTE	0 – 255	8	Битовая строка длины 8
WORD	0 – 65535	16	Битовая строка длины 16
DWORD	0 – 4294967295	32	Битовая строка длины 32
INT	–32768 – 32767	16	Целое
DINT	–2147483648 – 2147483647	32	Двойное целое
UDINT	0 – 4294967295	32	Двойное целое без знака
REAL	абсолютная величина минимального значения: 1.0E-44 абсолютная величина максимального значения: 3.402823E+38	32	Действительное одинарной точности
LREAL	абсолютная величина минимального значения: 4.94065645841247E-324 абсолютная величина максимального значения: 1.7976931348623157E+308	64	Действительное двойной точности

<sup>1</sup> – в системе исполнения приложений ГОСТ Р МЭК 61131-3 для хранения значений типа BOOL используется 1 байт, младший бит которого служит для проверки и операций со значением типа BOOL.

## 2. Общие сведения

### 2.1. Назначение Fastwel PLC Application Toolkit

Fastwel PLC Application Toolkit является программным обеспечением, устанавливаемым на ПК, интегрируемым с одной или несколькими совместимыми IDE МЭК 61131-3 и обеспечивающим возможность разработки приложений сбора, обработки данных и управления для контроллеров, выпускаемых под торговой маркой Fastwel, в том числе:

- Создание и редактирование проектов приложений для контроллеров Fastwel на базе процессоров с архитектурой x86 и ARM.
- Создание и редактирование конфигурации периферийных модулей, которые предполагается иметь в составе в контроллера, в том числе типы модулей, местоположение модулей на межмодульной шине, а также настройку параметров модулей в специальных редакторах, интегрированных в IDE МЭК 61131-3, в удобной для восприятия форме.
- Создание и редактирование конфигурации сетевых сервисов, которые должны функционировать в контроллере для обмена данными реального времени с другими контроллерами и/или системами сбора данных и оперативного диспетчерского управления и другими системами верхнего уровня, а также настройку параметров коммуникационных.

### 2.2. Структура и состав программного обеспечения

#### 2.2.1. Общие сведения

Структура Fastwel PLC Application Toolkit показана на рисунке 1.



Рисунок 1 – Структура Fastwel PLC Application Toolkit

Fastwel PLC Application Toolkit поставляется в виде двух вариантов программы установки (инсталлятора) на ПК с операционной системой MS Windows, в том числе: Windows 7 Service Pack 1, Windows 8, Windows 8.1, Windows 10 или Windows 11:

1. Базовый: *FastwelPLCAplicationToolkit.exe* – предполагает наличие на ПК ранее установленной совместимой IDE МЭК 61131-3 и содержит только программные компоненты, документацию и примеры программирования для контроллеров Fastwel. Совместимыми IDE МЭК 61131-3 являются:  
CODESYS версий от V3.5 SP14 Patch 1 до V3.5 SP17 Patch 3;  
Astra.IDE версии 1.7.x.x.
2. Расширенный: *FastwelPLCAplicationToolkitFull.exe* – помимо программных компонентов, документации и примеров программирования для контроллеров Fastwel, включает в себя IDE МЭК 61131-3 CODESYS версии не ниже V3.5 SP16 Patch 7 (3.5.16.700).

Обе программы установки поставляются на компакт-диске Fastwel DVD или могут быть загружены по адресу [ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel\\_IO/Version3/Setup](ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Setup).

### 2.2.2. Системные компоненты

В состав системных компонентов Fastwel PLC Application Toolkit включены обновления операционной системы Windows 7 Service Pack 1, необходимые для установки .NET Framework v4.8, а также автономный инсталлятор .NET Framework v4.8, который должен быть установлен перед установкой IDE МЭК 61131-3 на ПК с операционными системами Windows 7 Service Pack 1, Windows 8/8.1.

Установка системных компонентов происходит автоматически только при необходимости в процессе работы инсталлятора Fastwel PLC Application Toolkit на ПК с операционными системами Windows 7 Service Pack 1 или Windows 8/8.1.

При установке обновлений Windows 7 Service Pack 1 может потребоваться перезапуск ПК, после которого работа инсталлятора Fastwel PLC Application Toolkit будет продолжена.

Лицензионное соглашение .NET Framework v4.8 предполагает наличие у пользователя корректной активной лицензии на используемую операционную систему Windows.

### 2.2.3. Драйвер сервисного порта USB

В составе Fastwel PLC Application Toolkit имеется файл установки конфигурации драйвера Windows последовательного порта, реализуемого через интерфейс USB ПК, к которому подключен сервисный порт USB контроллера, например, CPM723-01.

Данный драйвер устанавливается только в операционных системах Windows 7 Service Pack 1 и Windows 8/8.1.

В операционных системах Windows 10 и Windows 11 используется драйвер CDC со стандартной конфигурацией.

### 2.2.4. Пакет расширения среды разработки IDE МЭК 61131-3

Fastwel PLC Application Toolkit содержит пакет расширения IDE МЭК 61131-3 *FastwelTargets.package*, который включает в себя:

- Файлы описания целевых устройств, устройств ввода-вывода и сетевых сервисов, представляющих контроллеры Fastwel в IDE МЭК 61131-3 для генерации исполняемого кода целевых процессоров и бинарной конфигурации устройств ввода-вывода и сетевых сервисов.
- Набор библиотек, содержащих функциональные блоки, функции и другие программные единицы, обеспечивающие доступ к специфическим функциональным возможностям контроллеров Fastwel из приложений.
- Наборы системных библиотек, обеспечивающих совместимость системных вызовов, формируемых IDE МЭК 61131-3 при компиляции и кодогенерации пользовательского приложения, с соответствующими реализациями системных вызовов в системе исполнения приложений в контроллерах Fastwel.
- Компоненты расширения (*плагины*) IDE МЭК 61131-3, позволяющие создавать и редактировать в специальных интерактивных редакторах конфигурационную информацию для периферийных модулей и сетевых сервисов в составе контроллера, а также генерировать дополнительную информацию времени выполнения.

Данный пакет расширения устанавливается автоматически в выбранные пользователем профили IDE МЭК 61131-3 в процессе работы инсталлятора Fastwel PLC Application Toolkit.

### 2.2.5. Примеры программирования

Fastwel PLC Application Toolkit содержит проекты IDE МЭК 61131-3 с примерами программирования, демонстрирующими приемы программного доступа к подсистемам контроллера, управления ими, обработки данных периферийных модулей, диагностики и оценки вычислительной загрузки контроллера.

Для поставляемых примеров программирования в меню запуска Windows создаются ярлыки, облегчающие поиск и доступ к проектам.

Данные примеры программирования могут использоваться в качестве основы при реализации пользовательских приложений.

### 2.2.6. Документация

Fastwel PLC Application Toolkit содержит файлы эксплуатационной и программной документации для контроллеров Fastwel в формате pdf.

Для документации в меню запуска Windows создаются ярлыки, облегчающие доступ к файлам.

### 2.2.7. IDE МЭК 61131-3 CODESYS V3

В состав расширенного варианта инсталлятора Fastwel PLC Application Toolkit входит наиболее стабильная совместимая IDE МЭК 61131-3 CODESYS V3, которую предлагается установить при запуске инсталлятора Fastwel PLC Application Toolkit при отсутствии на ПК ранее установленной совместимой IDE МЭК 61131-3, а также при наличии на ПК другой совместимой IDE МЭК 61131-3 или CODESYS V3, версия которой ниже версии в составе Fastwel PLC Application Toolkit.

IDE МЭК 61131-3 CODESYS V3 разработана фирмой CODESYS GmbH и обеспечивает выполнение следующих функций:

1. Реализацию прикладного алгоритма обработки данных и управления на языках ST, IL, LD, FBD, SFC стандарта ГОСТ Р МЭК 61131-3 и трансляцию разработанного приложения в исполняемый код процессора контроллера.
2. Создание конфигурации контроллера, которая включает в себя перечень описаний периферийных модулей, входящих в его состав, параметры каждого модуля, описания коммуникационных объектов – сообщений, поступающих в контроллер по сети и передаваемых контроллером в сеть, а также параметры исполнения кода приложения.
3. Отображение входных и выходных переменных разрабатываемого приложения на сетевые сообщения и каналы периферийных модулей.
4. Загрузку приложения в контроллер.
5. Удаленную отладку и управление исполнением приложения в контроллере.
6. Сервисные функции, включая диагностирование исполнения, загрузку и выгрузку файлов, трассировку значений переменных и т.д.

Среда разработки Astra.IDE может быть загружена отдельно с веб-сервера производителя по адресу <https://reglab.ru/software/astraide> или [https://prosoftsystems.ru/catalog/show/astra\\_ide](https://prosoftsystems.ru/catalog/show/astra_ide).

### 2.2.8. Перечень файлов Fastwel PLC Application Toolkit

Перечень программных компонентов и файлов, входящих в программы установки Fastwel PLC Application Toolkit, приведен в таблице 4.

Местоположение программных компонентов и файлов указано относительно каталога, выбираемого пользователем в процессе работы мастера установки после запуска инсталлятора или относительно подкаталога, находящегося в расположенной выше строке таблицы. Каталог обозначен символом "." в столбце "Местоположение" таблицы 4.

**Таблица 4 – Состав программы установки Fastwel PLC Application Toolkit**

Компонент/файл	Местоположение	Описание
Драйвер сервисного порта USB	.\bin\FastwelCDC\	Драйвер сервисного USB-порта контроллера. Обеспечивает на ПК создание устройства CPM723 USB to Serial Converter при подключении сервисного порта контроллера к USB-порту ПК. Устанавливается автоматически при установке пакета адаптации на операционных системах Windows 7 Service Pack 1, Windows 8, Windows 8.1. Не требуется на Windows 10 и Windows 11.

Компонент/файл	Местоположение	Описание
Программа установки CODESYS V3 (setup.exe)	.\CODESYS V3 Setup	Программа установки IDE МЭК 61131-3 CODESYS V3. Входит в состав только расширенной программы установки FastwelPLCAApplicationToolkitFull.exe.
FastwelPLCAApplicationToolkit_UM.pdf	.\doc	ИМЕС.00390-03 33 01. Пакет инструментальных средств ПЛК Fastwel. Руководство пользователя
FIO_UM.pdf	.\doc	ФАПИ.421459.700РЭ. FASTWEL I/O. Распределённая система ввода–вывода. Руководство по эксплуатации.
FIO-2_UM1.pdf	.\doc	ИМЕС.421459.252РЭ. Комплекс программно-технический Fastwel I/O-2. Руководство по эксплуатации. Часть 1.
FIO-2_UM2.pdf	.\doc	ИМЕС.421459.252РЭ1. Комплекс программно-технический Fastwel I/O-2. Руководство по эксплуатации. Часть 2.
CPM810-03_CDSV3_UM.pdf	.\doc	ИМЕС.00300-03 33 02-2. Контроллер программируемый универсальный CPM810-03. Руководство по конфигурированию и программированию
CPM723-01_CDSV3_UM.pdf	.\doc	ИМЕС.00300-03 33 02-1. Контроллер программируемый CPM723-01. Руководство по конфигурированию и программированию.
FBUS_CDSV3_UM.pdf	.\doc	ИМЕС.00300-03 33 01. Модули ввода-вывода. Руководство программиста
MODBUS_CDSV3_UM.pdf	.\doc	ИМЕС.00300-03 33 03. Протокол MODBUS. Руководство по конфигурированию и программированию
IEC60870-5-104_CDSV3_UM.pdf	.\doc	ИМЕС.00300-03 33 04. Протокол ГОСТ Р МЭК 60870-5-104. Руководство по конфигурированию и программированию
NIM745_UM.pdf	.\doc	NIM745-01. Руководство пользователя
NIM74502_PM.pdf	.\doc	NIM745-02. Руководство по эксплуатации
Примеры программирования для CPM723-01	.\examples\CPM723-01	Примеры программирования для CPM723-01
DateTimeUtilities.project <sup>V</sup>	\DateTime	Использование функций для работы с временем и датой, получение статуса батареи
FastwelBoardExample.project <sup>V</sup>	\FastwelBoard	Приемы работы с функциями чтения состояния переключателей и управления светодиодом USER
FileRetainer.project <sup>VE</sup>	\FastwelCore	Реализация энергонезависимых переменных при помощи функционального блока RETAIN_VAR_ENGINE из библиотеки FastwelCore. Получение статуса карты MicroSD.
PlatformControl.project <sup>VE</sup>	\FastwelCore	Приемы работы с функциями чтения серийного номера и сброса системы исполнения. Автоматический запуск приложения после полной загрузки. Обработка системного события LegacyOnInit.
SystemEvents.project <sup>VE</sup>	\FastwelCore	Обработка системных событий. Запись информации о системных событиях в файл на карте MicroSD с меткой времени событий. Измерение коротких интервалов. Использование ациклических задач.
TasksExchange.project <sup>VE</sup>	\FastwelCore	Безопасный обмен данными между циклическими задачами. Получение статистики исполнения задач. Использование наследования функциональных блоков. Формирование команд управления. Обработка системного события LegacyOnInit.
FastwelCpuLoadExample.project <sup>V</sup>	\FastwelCpuLoad	Пример оценки загрузки процессора в приложении с использованием библиотеки FastwelCpuLoad
aim721.project	\FastwelFbuslo\AIM721	Обработка данных и диагностика AIM721
aim722.project	\FastwelFbuslo\AIM722	Обработка данных и диагностика AIM722
aim723.project	\FastwelFbuslo\AIM723	Обработка данных и диагностика AIM723
aim724.project	\FastwelFbuslo\AIM724	Обработка данных и диагностика AIM724
aim725.project	\FastwelFbuslo\AIM725	Обработка данных и диагностика AIM725
aim72503.project	\FastwelFbuslo\AIM725	Обработка данных и диагностика AIM72503

Компонент/файл	Местоположение	Описание
aim726.project	\\FastwelFbuslo\\AIM726	Обработка данных AIM726
aim727.project	\\FastwelFbuslo\\AIM727	Обработка данных AIM727
aim728.project	\\FastwelFbuslo\\AIM728	Обработка данных AIM728
aim729.project	\\FastwelFbuslo\\AIM729	Обработка данных AIM729
aim730.project	\\FastwelFbuslo\\AIM730	Формирование сигналов треугольной формы на выходах AIM730
aim731.project	\\FastwelFbuslo\\AIM731	Формирование сигналов треугольной формы на выходах AIM733
aim791.project	\\FastwelFbuslo\\AIM791	Обработка данных и диагностика AIM791
aim792.project	\\FastwelFbuslo\\AIM792	Обработка данных и диагностика AIM792
aim826.project <sup>v</sup>	\\FastwelFbuslo\\AIM826	Обработка данных AIM826
aim891.project <sup>v</sup>	\\FastwelFbuslo\\AIM891	Обработка данных AIM891
dim711.project	\\FastwelFbuslo\\DIM711	Формирование сигналов в виде меандра на выходах DIM711 с использованием функции ШИМ и программным способом.
dim712.project <sup>v</sup>	\\FastwelFbuslo\\DIM712	Формирование команд управления на выходах модуля DIM712.
dim713.project <sup>v</sup>	\\FastwelFbuslo\\DIM713	Формирование команд управления на выходах модуля DIM713.
dim717.project <sup>v</sup>	\\FastwelFbuslo\\DIM717	Прием дискретных сигналов модулем DIM717, программная фильтрация дребезга, программный и аппаратный счет импульсов. Формирование импульсов на выходах модуля DIM718.
dim718.project	\\FastwelFbuslo\\DIM718	Формирование сигналов в виде меандра на выходах DIM718 с использованием функции ШИМ и программным способом.
dim719.project	\\FastwelFbuslo\\DIM719	Формирование сигналов в виде меандра на выходах DIM719 с использованием функции ШИМ и программным способом.
dim760.project <sup>v</sup>	\\FastwelFbuslo\\DIM760	Прием дискретных сигналов модулем DIM760, программная фильтрация дребезга, программный и аппаратный счет импульсов. Формирование импульсов на выходах модуля DIM718.
dim762.project <sup>v</sup>	\\FastwelFbuslo\\DIM762	Прием дискретных сигналов модулем DIM762, программная фильтрация дребезга, программный и аппаратный счет импульсов. Формирование импульсов на выходах модуля DIM719.
dim763.project	\\FastwelFbuslo\\DIM763	Формирование сигналов в виде меандра на выходах DIM763 с использованием функции ШИМ и программным способом.
dim764_freq_meter.project	\\FastwelFbuslo\\DIM764	Восьмиканальный измеритель частоты на базе DIM764. Фильтрация показаний измерителя. Формирователь частотного сигнала на базе DIM719.
dim764_updown_counter.project <sup>v</sup>	\\FastwelFbuslo\\DIM764	Квадратурный счетчик на базе DIM764 для приема сигнала энкодера. Формирование квадратурного сигнала на выходах DIM719.
dim764_ext_encoder_input.project <sup>v</sup>	\\FastwelFbuslo\\DIM764	Квадратурный счетчик на базе DIM764 для приема сигнала энкодера с возможностью контроля частоты вращения менее 0,765 Гц. Формирование квадратурного сигнала на выходах DIM719.
dim766.project <sup>v</sup>	\\FastwelFbuslo\\DIM766	Прием дискретных сигналов модулем DIM766, оценка параметров цепей связи с источниками сигналов, диагностика.
ModbusControl.project <sup>v</sup>	\\MODBUS	Пример применения библиотеки FastwelModbus для программного управления сервисами протоколов MODBUS и MODBUS TCP.
ModbusDocExamples.project <sup>vE</sup>	\\MODBUS	Пример реализации сетевых команд управления и параметризации алгоритма с использованием сервисов MODBUS.



Компонент/файл	Местоположение	Описание
ModbusRetains.project <sup>VE</sup>	MODBUS	Пример реализации параметризации алгоритма с использованием сервисов MODBUS.
Networkvariables.project <sup>V</sup>	NetworkVariables	Проект для двух CPM723-01, демонстрирующий обмен данными посредством сетевых переменных. Требуется предварительного включения функции мастера PTP на одном из контроллеров.
DataSource.gvl	NetworkVariables	Экспортированный список сетевых переменных, используемый в проекте Networkvariables.project
ModbusRtuClient.project	ISysCom	Пример реализации мастера MODBUS RTU с использованием библиотеки FastwelModbusRTUClientSerial и модуля NIM742. Использование встроенного сервера MODBUS через NIM742.
SerialConsole.project <sup>E</sup>	ISysCom	Реализация последовательной консоли ввода-вывода через NIM742.
SimpleDataLogger.project	ISysFile	Применение функций библиотек SysFile и SysDir для сохранения данных в файле на карте MicroSD с метками времени. Получение статуса карты MicroSD. Измерение коротких интервалов.
TCP_UDP_Sockets.project <sup>VE</sup>	ISysSocket	Применение функций библиотеки SysSocket для программирования обмена данными с использованием протоколов TCP и UDP. Клиенты и серверы функционируют на одном контроллере.
2xPLCs_Sockets.project <sup>VE</sup>	ISysSocket	Применение функций библиотеки SysSocket для программирования обмена данными с использованием протокола TCP. Клиенты и сервер функционируют на двух контроллерах.
tutorial.project	ITutorial	Учебный проект.
Примеры программирования для CPM810-03	examples\CPM810-03	Примеры программирования для CPM810-03
DateTimeUtilities.project <sup>V</sup>	IDateAndTime	Использование функций для работы с временем и датой, получение статуса батареи
FastwelBoardExample.project <sup>V</sup>	IFastwelBoard	Приемы работы с функциями чтения состояния переключателей и управления светодиодом USER
FileRetainer.project <sup>VE</sup>	IFastwelCore	Реализация энергонезависимых переменных при помощи функционального блока RETAIN_VAR_ENGINE из библиотеки FastwelCore. Получение статуса карты MicroSD.
PlatformControl.project <sup>VE</sup>	IFastwelCore	Приемы работы с функциями чтения серийного номера и сброса системы исполнения. Автоматический запуск приложения после полной загрузки. Обработка системного события LegacyOnInit.
SystemEvents.project <sup>VE</sup>	IFastwelCore	Обработка системных событий. Запись информации о системных событиях в файл на карте microSD с меткой времени событий. Измерение коротких интервалов. Использование ациклических задач.
TasksExchange.project <sup>VE</sup>	IFastwelCore	Безопасный обмен данными между циклическими задачами. Получение статистики исполнения задач. Использование наследования функциональных блоков. Формирование команд управления. Обработка системного события LegacyOnInit.
FastwelCpuLoadExample.project <sup>V</sup>	IFastwelCpuLoad	Пример оценки загрузки процессора в приложении с использованием библиотеки FastwelCpuLoad
aim826.project <sup>V</sup>	IFastwelFbus\AIM826	Обработка данных AIM826
aim891.project <sup>V</sup>	IFastwelFbus\AIM891	Обработка данных AIM891
dim812.project <sup>V</sup>	IFastwelFbus\DIM812	Формирование команд управления на выходах модуля DIM812.

Компонент/файл	Местоположение	Описание
dim813.project <sup>V</sup>	\FastwelFbuslo\DIM813	Формирование команд управления на выходах модуля DIM813.
dim713.project <sup>V</sup>	\FastwelFbuslo\DIM713	Формирование команд управления на выходах модуля DIM713.
dim717.project <sup>V</sup>	\FastwelFbuslo\DIM717	Прием дискретных сигналов модулем DIM717, программная фильтрация дребезга, программный и аппаратный счет импульсов. Формирование импульсов на выходах модуля DIM718.
Networkvariables.project <sup>V</sup>	\NetworkVariables	Проект для двух СРМ810-03, демонстрирующий обмен данными посредством сетевых переменных. Требуется предварительного включения функции мастера РТР на одном из контроллеров.
SerialConsole.project <sup>E</sup>	\SysCom	Реализация последовательной консоли ввода-вывода через порт COM1.
TCP_UDP_Sockets.project <sup>VE</sup>	\SysSocket	Применение функций библиотеки SysSocket для программирования обмена данными с использованием протоколов TCP и UDP. Клиенты и серверы функционируют на одном контроллере.
tutorial.project	\Tutorial	Учебный проект.
FastwelTargets.package	.\packages	Пакет расширения среды разработки IDE МЭК 61131-3 для контроллеров Fastwel Targets Extension Package. Устанавливается автоматически в процессе установки Fastwel PLC Application Toolkit. Включает в себя файлы описания устройств и сетевых сервисов, набор поддерживаемых библиотек и редакторы конфигурации устройств и сетевых сервисов Fastwel.
ПРИМЕЧАНИЕ. Проекты IDE МЭК 61131-3, имена которых отмечены символом <sup>V</sup> , содержат примеры использования форм визуализации, функционирующих в среде разработки на ПК. Проекты с именами, отмеченными символом <sup>E</sup> , содержат примеры обработки системных событий.		

### 2.3. Системные требования к рабочему месту разработчика

#### 2.3.1. Требования к аппаратным средствам

ПК, на котором предполагается использовать IDE МЭК 61131-3 с установленным Fastwel PLC Application Toolkit, должен иметь аппаратную конфигурацию, достаточную для функционирования операционной системы Windows 7 Service Pack 1, Windows 8, Windows 8.1, Windows 10 или Windows 11, а также:

- не менее 8 Гбайт оперативной памяти;
- не менее 25 Гбайт свободного пространства на диске;
- адаптер интерфейса Ethernet.

Рекомендуемое разрешение монитора 1680×1050, 1920×1080 и более.

При необходимости установки Fastwel PLC Application Toolkit с Fastwel DVD ПК должен быть оснащен устройством чтения DVD.

При необходимости организации взаимодействия между IDE МЭК 61131-3 и контроллерами Fastwel с использованием интерфейса USB ПК должен иметь в своем составе порт интерфейса USB 1.1/2.0. Для подключения порта USB контроллера к ПК следует использовать кабель ACS00092-02 или аналогичный.

При необходимости организации взаимодействия между IDE МЭК 61131-3 и контроллерами Fastwel с использованием интерфейса RS-232C ПК должен иметь в своем составе порт интерфейса RS-232C либо адаптер USB-to-COM, подключенный к свободному порту интерфейса USB. Для подключения порта RS-232C контроллера к ПК следует использовать кабель ACS00092-01 или аналогичный.

Более подробная информация о сервисных кабелях приведена в документах ФАПИ.421459.700РЭ и ИМЕС.421459.252РЭ.

### 2.3.2. Требования к системному программному обеспечению


Персональный компьютер, на котором предполагается использовать IDE МЭК 61131-3 с установленным Fastwel PLC Application Toolkit, должен иметь конфигурацию программных средств не хуже:

- операционная система Windows 7 Service Pack 1, Windows 8/8.1, Windows 10 или Windows 11;
- программа чтения документов в формате Adobe PDF (Adobe Acrobat Reader®, Foxit® Reader и др.).

Для настройки системных параметров контроллеров через имеющийся в составе СПО контроллеров встроенный веб-сервер на ПК должен быть установлен хотя бы один из следующих веб-браузеров:

- Mozilla Firefox® версии не ниже 52;
- Google Chrome® версии не ниже 56;
- Opera® версии не ниже 44;
- Microsoft Edge® версии не ниже 44.

При использовании операционной системы Windows 8, Windows 8.1, Windows 10 или Windows 11 рекомендуется оснастить пользовательский интерфейс операционной системы классическим меню запуска. Для этого можно установить программу расширения интерфейса пользователя Windows Classic Shell™ или аналогичную либо обеспечить доступ к меню запуска путем создания дополнительной панели инструментов в панели задач. Для создания панели инструментов с доступом к меню запуска:

1. При работе в операционной системе Windows 8/8.1 переключитесь из режима отображения "плитки" в режим рабочего стола (обычно нажатием кнопки ).
2. Щелкните правой кнопкой мыши над значком скрытых ярлыков в области уведомлений панели задач, как показано на рисунке 2, и выберите команду **Панели–Создать панель инструментов...** в контекстном меню.

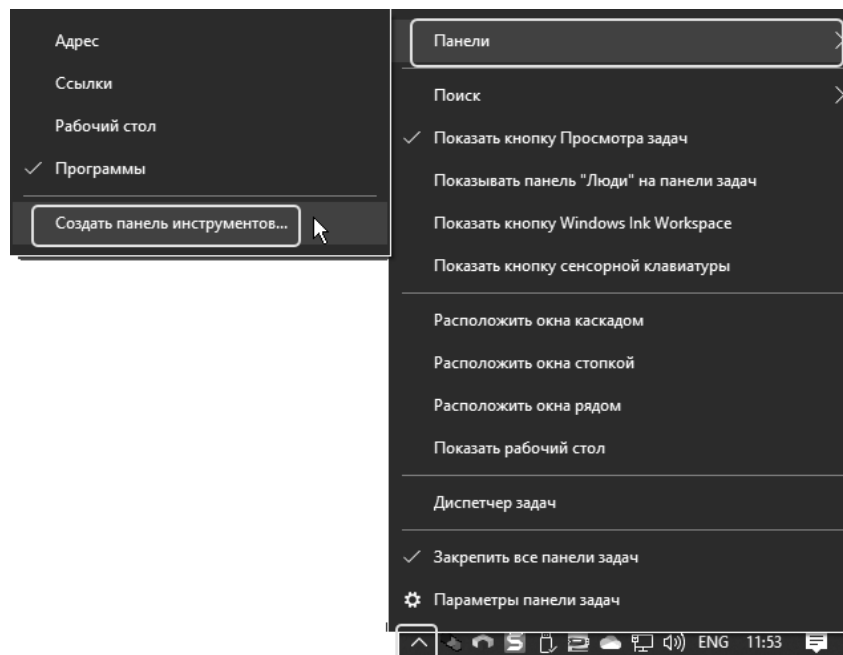
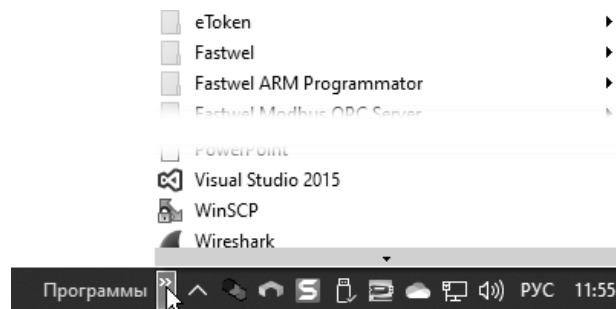



Рисунок 2 – Создание новой панели инструментов Windows 8/ 8.1/10/11

3. В появившейся диалоговой панели **Новая панель инструментов – Выбор папки** выберите папку C:\ProgramData\Microsoft\Windows\Start Menu\Programs и нажмите **Выбор папки**. Панель инструментов **Programs (Программы)** появится на панели задач, как показано на рисунке 3.



**Рисунок 3. Панель доступа к меню запуска Windows**

4. Для получения доступа к программным группам щелкните на значке  справа от имени только что созданной панели инструментов **Programs (Программы)**, как показано на рисунке 3.

### 3. Установка Fastwel PLC Application Toolkit и настройка IDE МЭК 61131-3

#### 3.1. Предварительные замечания

Перед началом установки убедитесь, что аппаратно-программная конфигурация ПК, на который предполагается установить Fastwel PLC Application Toolkit, соответствует требованиям, приведенным в п. 2.3 настоящего документа.

Для установки Fastwel PLC Application Toolkit и, если требуется, IDE МЭК 61131-3 пользователь должен иметь права администратора ПК.

Если на ПК ранее была установлена совместимая IDE МЭК 61131-3 согласно п. 2.2.1, то для установки Fastwel PLC Application Toolkit возможно использовать базовую программу установки FastwelPLCApplcationToolkit.exe.

Если до установки Fastwel PLC Application Toolkit на ПК не была установлена ни одна из совместимых IDE МЭК 61131-3, то возможны следующие варианты установки:

1. Использование расширенной программы установки FastwelPLCApplcationToolkitFull.exe, которая идентична базовой в части состава (см. п. 2.2), но дополнена программой установки IDE МЭК 61131-3 CODESYS V3 версии не ниже V3.5 SP16 Patch 7.
2. Загрузка программы установки IDE МЭК 61131-3 Astra.IDE по адресу <https://reglab.ru/software/astraide> или [https://prosoftsystems.ru/catalog/show/astra\\_ide](https://prosoftsystems.ru/catalog/show/astra_ide), установка ее на ПК, а затем запуск базовой программы установки FastwelPLCApplcationToolkit.exe.



В системных требованиях IDE МЭК 61131-3 Astra.IDE указана необходимость использования операционной системы Windows 10 и выше. Уточнение данного требования: установка Astra.IDE версии 1.7.0.0 возможна на ПК с 64-разрядным выпуском операционной системы Windows 10 и выше.

Если для установки Astra.IDE и Fastwel PLC Application Toolkit требуется использовать 64-разрядный выпуск операционных систем Windows 7 Service Pack 1, Windows 8 или Windows 8.1, следует предварительно запустить базовую программу установки (инсталлятор) FastwelPLCApplcationToolkit.exe, установить дополнительные системные компоненты Windows, включая .NET Framework v4.8, установить Astra.IDE, после чего повторно запустить инсталлятор FastwelPLCApplcationToolkit.exe.





Инсталлятор Fastwel PLC Application Toolkit выполняет следующие действия:

1. Проверяет тип и версию операционной системы, в которой выполняется установка, а также наличие прав администратора учетной записи пользователя, от имени которой запущен инсталлятор. Если установка выполняется с правами администратора в операционной системе Windows 7 Service Pack 1 или выше, то процесс установки будет продолжен. При отрицательном результате проверки установка будет завершена.
2. Проверяет наличие установленной на ПК совместимой IDE МЭК 61131-3 подходящей версии.  
Если используется базовый вариант инсталлятора (FastwelPLCApplcationToolkit.exe) и совместимая IDE МЭК 61131-3 не установлена на ПК, то работа инсталлятора будет завершена с предложением предварительно установить совместимую IDE МЭК 61131-3.  
Если установка выполняется на ПК с Windows 7 Service Pack 1 или Windows 8/8.1, то перед завершением работы инсталлятора будет предложено установить .NET Framework v4.8, поскольку он должен быть установлен на ПК для последующей успешной установки IDE МЭК 61131-3.  
Если IDE МЭК 61131-3 подходящей версии обнаружена на ПК, то будет выполнена установка Fastwel PLC Application Toolkit.
3. Если используется расширенный вариант инсталлятора (FastwelPLCApplcationToolkitFull.exe), и совместимая IDE МЭК 61131-3 подходящей версии не найдена на ПК, пользователю будет предложено выполнить установку среды

разработки IDE МЭК 61131-3, входящей в состав инсталлятора. В случае отказа работа инсталлятора будет завершена.

Если версия IDE МЭК 61131-3 CODESYS V3, обнаруженной на ПК, ниже версии IDE МЭК 61131-3 CODESYS V3, входящей в инсталлятор, либо если на ПК установлена IDE МЭК 61131-3 Astra.IDE, пользователю будет предложено установить IDE МЭК 61131-3 из состава инсталлятора. В случае отказа инсталлятор продолжит свою работу и установит на ПК Fastwel PLC Application Toolkit в профили обнаруженных совместимых IDE МЭК 61131-3 CODESYS V3, выбранных пользователем.

4. Если на ПК установлена только IDE МЭК 61131-3 Astra.IDE, то установка Fastwel PLC Application Toolkit будет выполнена в специальный профиль "Fastwel PLC Application Toolkit – Astra.IDE V1.7.0.0", созданный инсталлятором и содержащий только компоненты Astra.IDE, необходимые для работы с контроллерами Fastwel и исключающий компоненты для работы с другими контроллерами.
5. Если все предварительные условия для установки удовлетворены, то будет выполнена установка Fastwel PLC Application Toolkit.

	<p>Инсталлятор Fastwel PLC Application Toolkit не требует наличия работоспособного соединения ПК с сетью Интернет.</p> <p>После автоматической установки системных компонентов из состава инсталлятора Fastwel PLC Application Toolkit в операционной системе Windows 7 Service Pack 1 или Windows 8/8.1 для последующей установки IDE МЭК 61131-3 и Fastwel PLC Application Toolkit не требуется соединение ПК с сетью Интернет.</p>
	<p>В зависимости от производительности ПК процесс установки расширенного варианта Fastwel PLC Application Toolkit может занимать до 30-40 минут.</p> <p>В течение этого времени нежелательно запускать на ПК какие-либо программы и/или завершать работу ПК.</p>
	<p>При установке Fastwel PLC Application Toolkit на ПК с операционной системой Windows, для которой установлен русский язык интерфейса пользователя, будет автоматически выбран русский язык мастера установки и созданы русскоязычные названия ярлыков Fastwel PLC Application Toolkit в меню запуска Windows ("Пакет инструментальных средств ПЛК Fastwel").</p> <p>При установке на ПК с операционной системой Windows, для которой выбран язык интерфейса пользователя, отличный от русского, установка будет выполнена с автоматически выбранным английским языком мастера установки и созданы названия ярлыков Fastwel PLC Application Toolkit в меню запуска Windows на английском языке.</p>
	<p>При установке IDE МЭК 61131-3 CODESYS V3 в операционной системе Windows, для которой выбран русский язык интерфейса пользователя, будет автоматически установлен русский язык интерфейса пользователя среды разработки и справочной системы.</p> <p>Указания о смене языка интерфейса пользователя среды разработки приведены в п. 3.6.2 настоящего документа.</p>


### 3.2. Установка Fastwel PLC Application Toolkit

#### 3.2.1. Установка базового варианта FastwelPLCAApplicationToolkit.exe

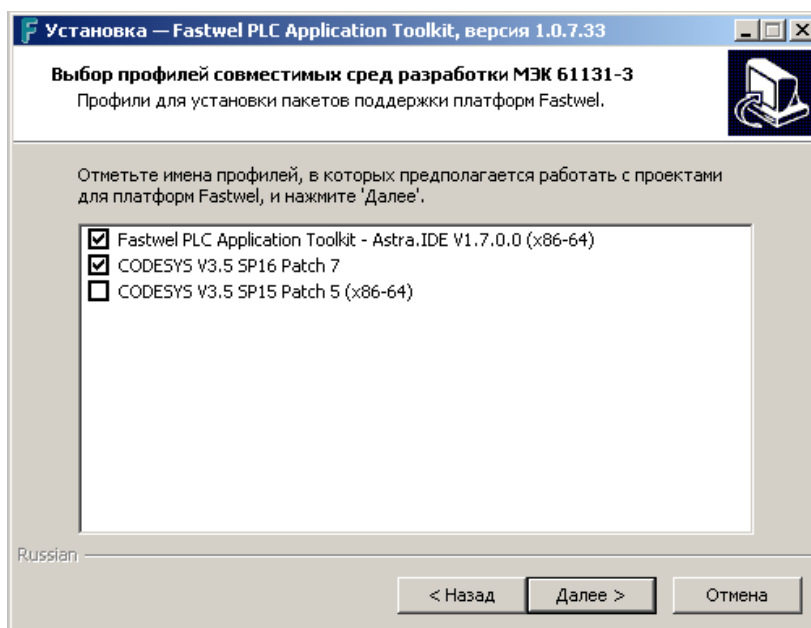
Для установки базового варианта Fastwel PLC Application Toolkit:

1. Если к порту USB ПК подключен контроллер Fastwel, имеющий сервисный порт интерфейса USB, необходимо выключить питание контроллера или отсоединить кабель ACS00092-02 или аналогичный, используемый для связи контроллера с ПК.

2. Если на ПК запущена среда разработки IDE МЭК 61131-3 или менеджер пакетов среды разработки, следует завершить работу среды разработки и/или менеджера пакетов.
3. Запустить программу FastwelPLCApplicationToolkit.exe.  
Если на ПК был ранее установлен Fastwel PLC Application Toolkit меньшей версии, чем в настоящий момент находится в процессе установки, на экран монитора будет выведена диалоговая панель **Установка** с сообщением **Обнаружен установленный пакет инструментальных средств ПЛК Fastwel версии <номер версии> Вы хотите удалить предыдущую установку?**. Для удаления предыдущей версии следует нажать **Да**, а для продолжения установки поверх ранее установленной версии – **Нет**.  
Если на ПК был ранее установлен Fastwel PLC Application Toolkit той же версии, что в настоящий момент находится в процессе установки, то установка будет автоматически продолжена.
4. Через некоторое время на экран монитора будет выведено окно мастера установки **Установка – Fastwel PLC Application Toolkit, версия <номер версии>** с текстом лицензионного соглашения.  
После ознакомления с информацией лицензионного соглашения в появившемся окне мастера установки **Лицензионное соглашение** для продолжения установки необходимо выбрать **Я принимаю условия соглашения** и нажать **Далее**.  
Для прекращения установки следует выбрать **Я не принимаю условия соглашения** и нажать **Отмена**.
5. Если установка продолжена, на экран будет выведено окно мастера установки **Выбор папки установки** с предложением выбрать каталог, в котором будут размещены файлы Fastwel PLC Application Toolkit (см. таблицу 4 п. 2.2.8).

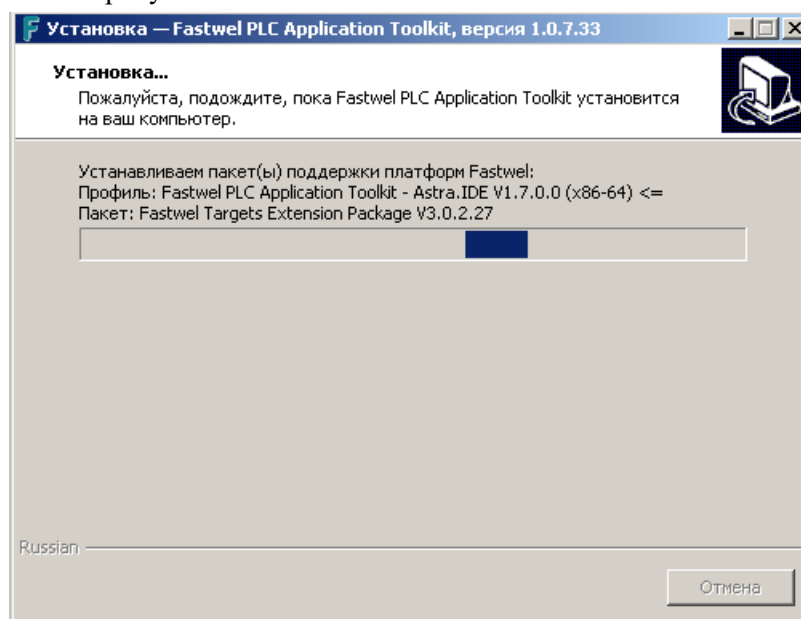
	<p>По умолчанию предлагается каталог установки в локальном профиле Windows текущей учетной записи пользователя: <i>C:\Users\&lt;Имя пользователя&gt;\AppData\Local\Programs\Fastwel\Fastwel PLC Application Toolkit</i>.</p> <p>При необходимости изменить каталог установки следует нажать кнопку <b>Обзор</b> и выбрать каталог, отличный от <i>C:\Program Files (x86)</i> и <i>C:\Program Files</i>, поскольку иначе при работе с проектами примеров программирования в IDE МЭК 61131-3 не будет возможности сохранять изменения и в файлах настроек среды разработки с пользовательскими предпочтениями.</p>
---	--

6. После выбора каталога установки следует нажать **Далее** и в окне **Выбор профилей совместимых сред разработки МЭК 61131-3** выбрать профили IDE МЭК 61131-3, в которые предполагается установить пакеты расширения среды разработки из состава Fastwel PLC Application Toolkit, как показано на рисунке 4.



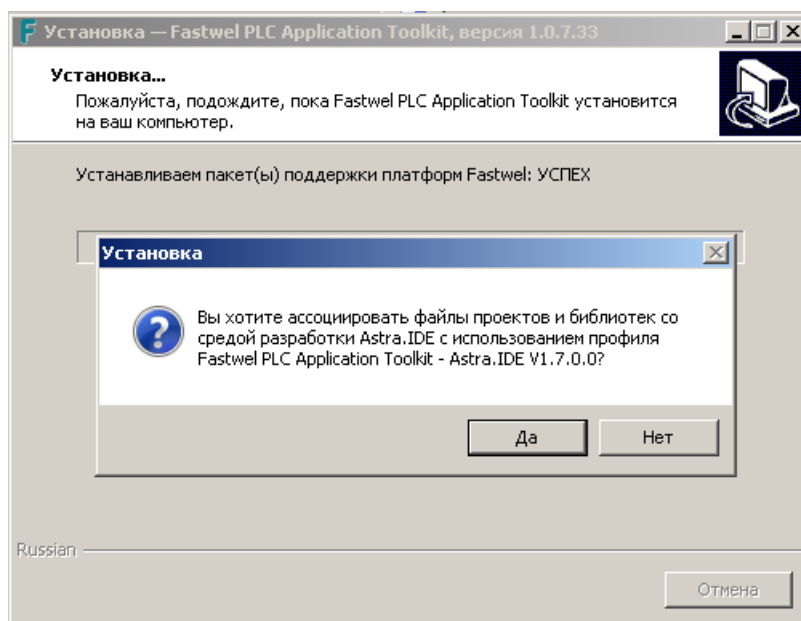
**Рисунок 4 – Профили IDE МЭК 61131-3 для установки пакетов расширения среды разработки из состава Fastwel PLC Application Toolkit**

После выбора имен профилей и нажатия кнопки **Далее** будет выполнена установка пакетов расширения IDE МЭК 61131-3 в выбранные профили IDE МЭК 61131-3, как показано на рисунке 5.



**Рисунок 5 – Установка пакетов расширения среды разработки в выбранные профили**


7. Если на ПК установлена IDE МЭК 61131-3 Astra.IDE, то после установки пакетов расширения в выбранные профили на экран будет выведено окно **Вы хотите ассоциировать файлы проектов и библиотек со средой разработки Astra.IDE с использованием профиля Fastwel PLC Application Toolkit – Astra.IDE V1.7.0.0**, показанная на рисунке 6.



**Рисунок 6 – Запрос установки ассоциаций файлов .project, .library, .projectarchive с Astra.IDE, запускаемой с профилем Fastwel PLC Application Toolkit – Astra.IDE V1.7.0.0**

При необходимости открывать двойным щелчком мышью файлы с расширениями .project, .library и .projectarchive в среде разработки Astra.IDE с профилем Fastwel PLC Application Toolkit следует нажать **Да**.



	<p>Установленная ассоциация приложения Astra.IDE с файлами .project, .library и .projectarchive будет действовать только в случае, если Astra.IDE была установлена последней из всех совместимых IDE МЭК 61131-3 на данном ПК или является единственной установленной IDE МЭК 61131-3.</p> <p>Для изменения данного поведения следует использовать приложение IDE Selector, установленное на ПК при установке Astra.IDE.</p>
---	--

8. Если на ПК установлена операционная система Windows 10 или Windows 11, для работы с сервисным USB-портом контроллеров Fastwel будет использоваться универсальный драйвер для преобразователей USB в COM с конфигурацией, входящей в состав операционной системы.  
Если на ПК установлена операционная система Windows 7 Service Pack 1, Windows 8 или Windows 8.1, незадолго до завершения установки на экран монитора будет выведено окно мастера установки драйвера сервисного USB-порта контроллера **Fastwel CDC Driver Installer**, входящего в Fastwel PLC Application Toolkit. При необходимости организации взаимодействия между IDE МЭК 61131-3 с контроллерами Fastwel через сервисный порт USB контроллеров потребуются выполнить указания мастера установки драйвера для продолжения установки. Если в процессе будет предложено выполнить перезагрузку ПК, следует выбрать вариант с отложенной перезагрузкой (диалоговая панель **Чтобы изменения вступили в силу, нужно перезагрузить компьютер: Перезагрузить позже**).
9. Окончание процесса установки предваряется выводом на экран окна **Завершение Мастера установки Fastwel PLC Application Toolkit, версия <номер версии>**. Для завершения установки следует нажать кнопку **Завершить**.

### 3.2.2. Установка расширенного варианта Fastwel PLC Application Toolkit

В расширенный вариант программы установки пакета адаптации CODESYS V3 для контроллеров Fastwel входит программа установки среды разработки CODESYS V3 одной из последних версий на момент выпуска текущей версии пакета адаптации.

Для установки расширенного варианта пакета адаптации выполните следующие действия:

1. Если к порту USB ПК подключен контроллер Fastwel, имеющий сервисный порт интерфейса USB, необходимо выключить питание контроллера или отсоединить используемый для связи контроллера с ПК кабель ACS00092-02 или аналогичный.
2. Если на ПК запущена среда разработки IDE МЭК 61131-3 или менеджер пакетов среды разработки, следует завершить работу среды разработки и/или менеджера пакетов.
3. Запустите программу FastwelPLCApplicationToolkitFull.exe.  
Если на ПК был ранее установлен Fastwel PLC Application Toolkit меньшей версии, чем в настоящий момент находится в процессе установки, на экран монитора будет выведена диалоговая панель **Установка** с сообщением **Обнаружен установленный пакет инструментальных средств ПЛК Fastwel версии <номер версии> Вы хотите удалить предыдущую установку?**  
Для удаления предыдущей версии следует нажать **Да**, а для продолжения установки поверх ранее установленной версии – **Нет**.  
Если на ПК был ранее установлен Fastwel PLC Application Toolkit той же версии, что в настоящий момент находится в процессе установки, то установка будет автоматически продолжена.  
Если на ПК установлена совместимая среда разработки IDE МЭК 61131-3, отличная от входящей в инсталлятор, на экран будет выведена диалоговая панель со списком обнаруженных совместимых IDE МЭК 61131-3 и предложением установить IDE МЭК 61131-3 из инсталлятора, как показано на рисунке 7.

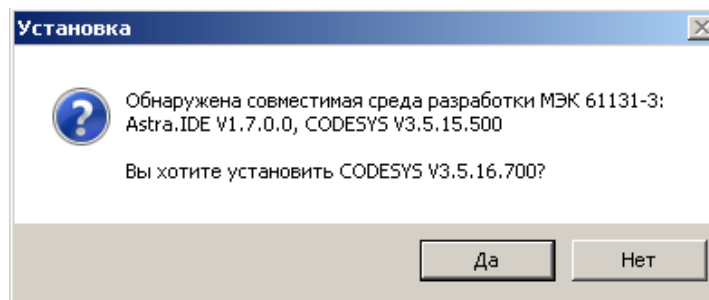



Рисунок 7. Предложение обновить среду разработки CODESYS V3

Если нажать **Нет**, установка будет продолжена без установки среды разработки.


Если нажать **Да**, в процессе установки будет также установлена IDE МЭК 61131-3, входящая в инсталлятор.

После нажатия **Да** или **Нет** на экран монитора будет выведено начальное окно мастера установки **Установка – Fastwel PLC Application Toolkit, версия <номер версии>** с текстом лицензионного соглашения.

4. После ознакомления с информацией лицензионного соглашения в появившемся окне мастера установки **Лицензионное соглашение** для продолжения установки необходимо выбрать **Я принимаю условия соглашения** и нажать **Далее**.  
Для прекращения установки следует выбрать **Я не принимаю условия соглашения** и нажать **Отмена**.
5. Если установка продолжена, на экран будет выведено окно мастера установки **Выбор папки установки** с предложением выбрать каталог, в котором будут размещены файлы Fastwel PLC Application Toolkit (см. таблицу 4 п. 2.2.8).

	<p>По умолчанию предлагается каталог установки в локальном профиле Windows текущей учетной записи пользователя: <i>C:\Users\&lt;Имя пользователя&gt;\AppData\Local\Programs\Fastwel\Fastwel PLC Application Toolkit</i>.</p> <p>При необходимости изменить каталог установки следует нажать кнопку <b>Обзор</b> и выбрать каталог, отличный от <i>C:\Program Files (x86)</i> и <i>C:\Program Files</i>, поскольку иначе при работе с проектами примеров программирования в IDE МЭК 61131-3 не будет возможности сохранять изменения и в файлах настроек среды разработки с пользовательскими предпочтениями.</p>
---	--

6. После выбора каталога установки следует нажать **Далее**, и, при положительном выборе **Да** в ответе на запрос мастера установки об установке IDE МЭК 61131-3 из состава инсталлятора на шаге 3 настоящих указаний, будет произведена установка IDE МЭК 61131-3 из инсталлятора.

	<p>На ПК с операционной системой Windows 7 Service Pack 1 или Windows 8/8.1 при отсутствии ранее установленной совместимой IDE МЭК 61131-3, перед установкой IDE МЭК 61131-3 будет произведена установка системных компонентов Windows, включая .NET Framework v4.8 (см. п. 3.1).</p>
---	---

Более подобная информация об установке IDE МЭК 61131-3 приведена в п. 3.3.

7. По завершении установки IDE МЭК 61131-3 будет продолжена работа мастера установки Fastwel PLC Application Toolkit, во время которой на экран монитора будет выведено окно выбора профилей совместимых IDE МЭК 61131-3 для установки пакетов расширения среды разработки, а затем окно, отображающее процесс установки пакетов расширения, как показано на рисунке 5.
8. Если на ПК установлена операционная система Windows 10 или Windows 11, для работы с сервисным USB-портом контроллеров Fastwel будет использоваться универсальный драйвер для преобразователей USB в COM с конфигурацией, входящей в состав операционной системы.

Если на ПК установлена операционная система Windows 7 Service Pack 1, Windows 8 или Windows 8.1, незадолго до завершения установки на экран монитора будет выведено окно мастера установки драйвера сервисного USB-порта контроллера **Fastwel CDC Driver Installer**, входящего в Fastwel PLC Application Toolkit. При необходимости

организации взаимодействия между средой разработки и контроллерами Fastwel через сервисный порт USB контроллеров потребуются выполнить указания мастера установки драйвера для продолжения установки. Если в процессе будет предложено выполнить перезагрузку ПК, следует выбрать вариант с отложенной перезагрузкой (диалоговая панель **Чтобы изменения вступили в силу, нужно перезагрузить компьютер: (Перезагрузить позже).**

9. Окончание процесса установки предваряется выводом на экран окна **Завершение Мастера установки Fastwel PLC Application Toolkit, версия <номер версии>**. Для завершения установки следует нажать кнопку **Завершить**.

### 3.3. Рекомендации по установке IDE МЭК 61131-3

#### 3.3.1. Установка CODESYS V3

Установка IDE МЭК 61131-3 CODESYS V3 выполняется мастером установки инсталлятора CODESYS V3, запускаемого автоматически из расширенной программы установки Fastwel PLC Application Toolkit.

В начале процесса мастер установки может предложить установить дополнительные системные компоненты, включенные в инсталлятор CODESYS V3, как показано на рисунке 8.

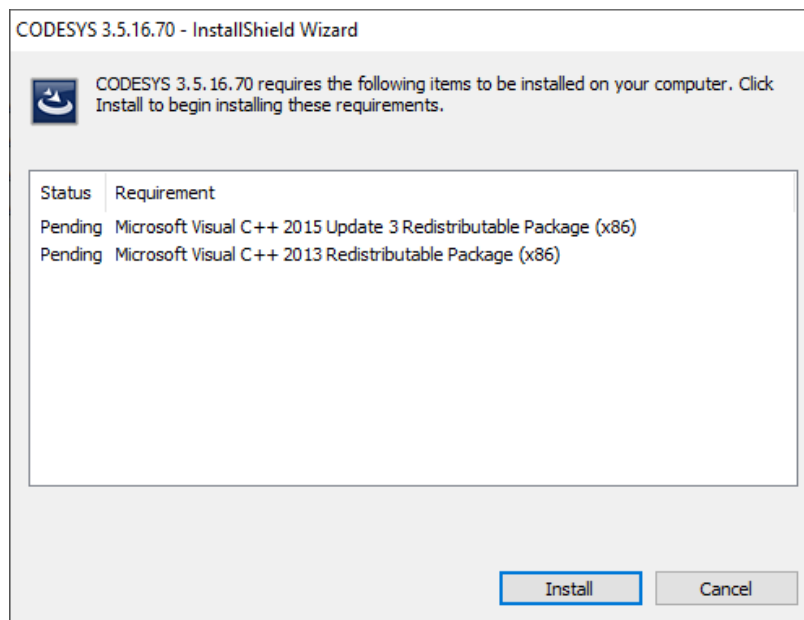


Рисунок 8 – Предложение установки дополнительных системных компонентов IDE МЭК 61131-3

Для продолжения установки необходимо нажать **Install**.

По завершении установки дополнительных системных компонентов и общих компонентов CODESYS V3 будут последовательно выведены окна мастера установки с текстом лицензионного соглашения и информацией о совместимости с предыдущими версиями.

Для продолжения установки следует выбрать **I accept the terms in the license agreement / I have read the information** и нажать **Next** в каждом окне, после чего выбрать каталог установки CODESYS V3 в окне **Destintation folder** и нажать **Next** для выбора полной (*Complete*) или выборочной (*Custom*) установки в окне **Setup Type**, показанном на рисунке 9.



Для ускорения процесса установки и уменьшения количества зависимостей в проектах IDE МЭК 61131-3 рекомендуется использовать вариант выборочной (*Custom*) установки с выключенными опциями *CODESYS SoftMotion*, *CODESYS Automation Server Connector* и *CODESYS Gateway V2.3*.

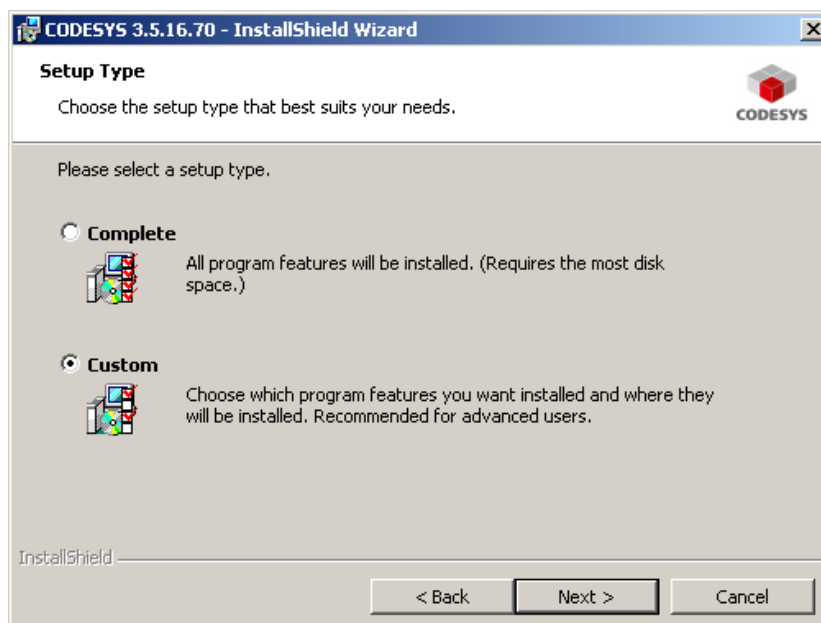


Рисунок 9 – Окно выбора варианта установки CODESYS V3

Рекомендуемый состав устанавливаемых компонентов CODESYS V3 показан на рисунке 10.

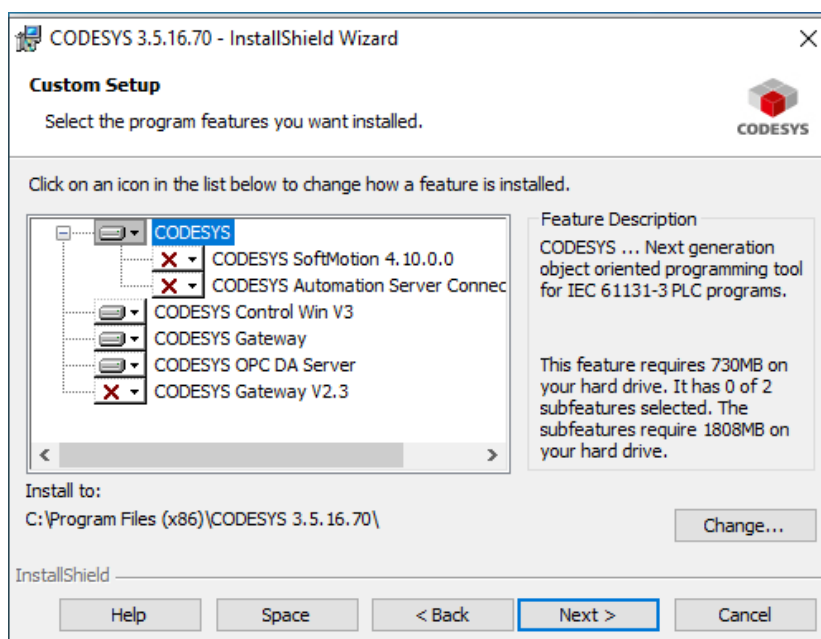


Рисунок 10 – Окно выбора компонентов CODESYS V3 с рекомендуемым перечнем устанавливаемых компонентов

После выбора устанавливаемых компонентов следует нажать **Next**, а затем **Next** в окне **Ready to Install the Program**.

Процесс установки CODESYS V3 займет около 30 – 40 минут в зависимости от производительности ПК и типа используемого системного диска (HDD или SSD).


По завершении установки CODESYS V3 на экран будет выведено окно мастера установки **InstallShield Wizard Completed**. После нажатия **Finish** будет продолжена установка Fastwel PLC Application Toolkit.

### 3.3.2. Установка Astra.IDE

Установка IDE МЭК 61131-3 Astra.IDE должна быть выполнена до установки Fastwel PLC Application Toolkit. Программа установки (инсталлятор) Astra.IDE может быть загружена по адресу <https://reglab.ru/software/astraide> или [https://prosoftsystems.ru/catalog/show/astra\\_ide](https://prosoftsystems.ru/catalog/show/astra_ide).

Указания по установке IDE МЭК 61131-3 Astra.IDE приведены в руководстве пользователя Astra.IDE, которое также может быть загружено по указанным выше адресам.

Настоящий подраздел содержит дополнительные сведения, в том числе относящиеся к работе Astra.IDE совместно с Fastwel PLC Application Toolkit.

	<p>Установка Astra.IDE возможна только на ПК с 64-разрядным вариантом операционной системы Windows.</p> <p>При необходимости установки Astra.IDE на ПК с операционной системой Windows 7 Service Pack 1 или Windows 8/8.1 следует предварительно запустить базовый вариант программы установки Fastwel PLC Application Toolkit и установить системные компоненты Windows, включая .NET Framework v4.8.</p>
---	--

Установка Astra.IDE выполняется мастером установки инсталлятора Astra.IDE (Astra.IDE 64 1.x.x.x.exe, где x.x.x – числовые компоненты версии), запускаемого вручную.

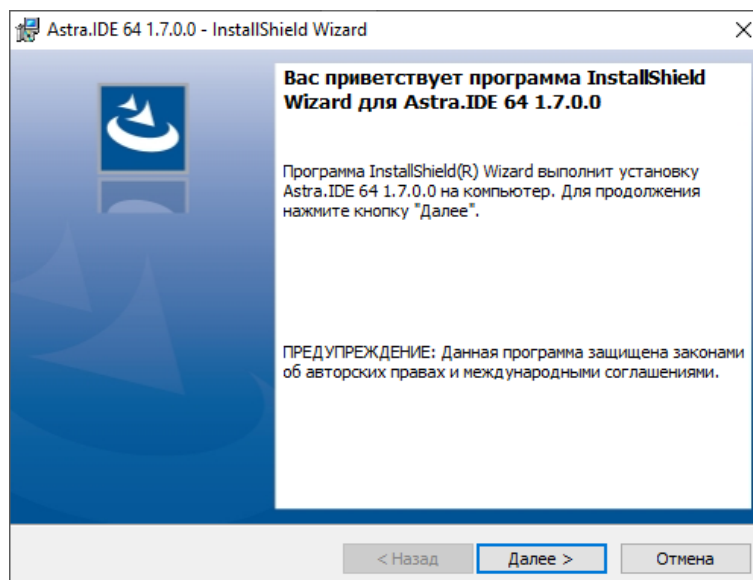



Рисунок 11 – Окно приветствия мастера установки Astra.IDE

	<p>После запуска программы установки Astra.IDE до появления окна приветствия мастера установки может потребоваться подождать несколько десятков секунд.</p> <p>До появления окна приветствия мастера установки, показанного на рисунке 11, не следует завершать работу Windows, перезапускать или выключать ПК.</p>
---	---

После нажатия кнопки **Далее** в окне приветствия мастера установки Astra.IDE, показанном на рисунке 11, на экран будет выведено окно **Лицензионное соглашение** с текстом лицензионного соглашения Astra.IDE, показанное на рисунке 12.

Для продолжения установки следует выбрать **Я принимаю условия лицензионного соглашения** и нажать **Далее**, после чего в окне **Папка назначения**, показанном на рисунке 13, будет возможность выбора каталога установки Astra.IDE либо оставления без изменений предложенного по умолчанию каталога *C:\Program Files\AstraRegul\Astra.IDE 64 1.x.x.x*.

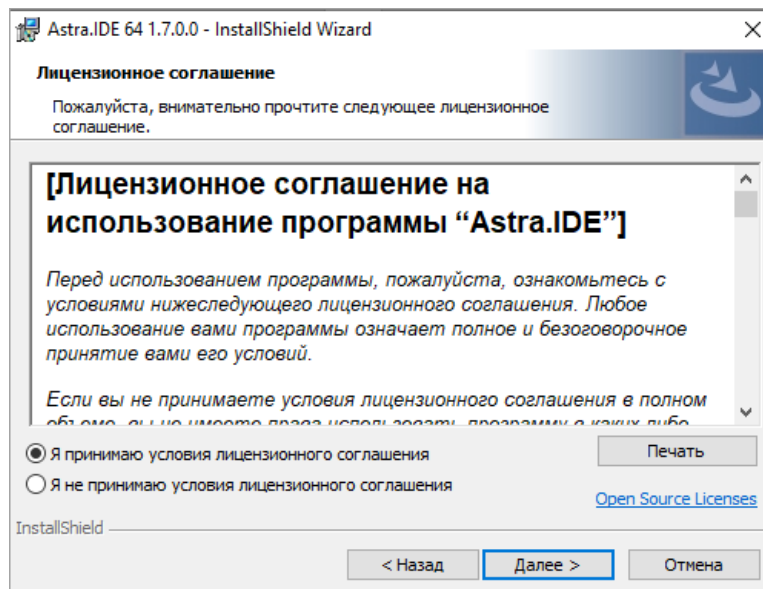


Рисунок 12 – Окно "Лицензионное соглашение" мастера установки Astra.IDE

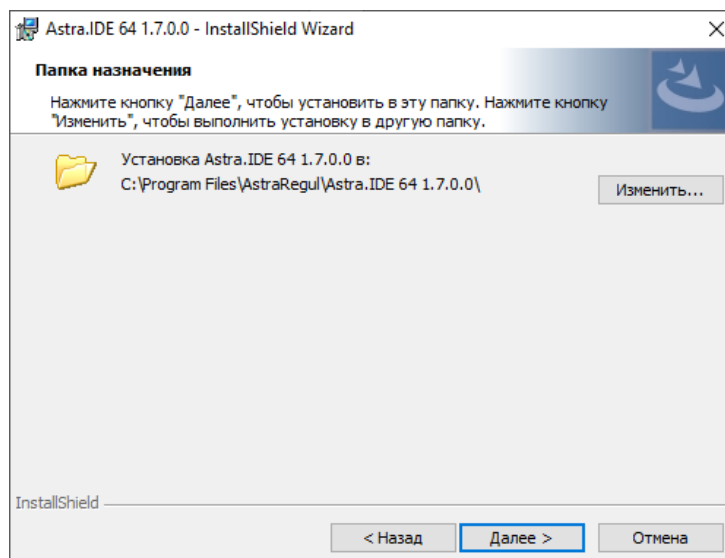




Рисунок 13 – Окно "Папка назначения" мастера установки Astra.IDE

После нажатия кнопки **Далее** в окне **Папка назначения** на экран будет выведено окно **Вид установки**, показанное на рисунке 14, для выбора варианта установки: *Полная* или *Выборочная*.

	<p>При установке Astra.IDE для ускорения процесса установки рекомендуется выбрать вариант установки <i>Выборочная</i>.</p>
	<p>При установке Astra.IDE на ПК, на который ранее была установлена другая совместимая IDE МЭК 61131-3, возможно не устанавливать компоненты Astra.IDE Control Win 64 и/или Astra.IDE Gateway 64, а пользоваться соответствующими компонентами из состава ранее установленной IDE МЭК 61131-3.</p>

После выбора варианта установки *Выборочная* в выведенном окне **Вид установки** на экран будет выведено окно **Выборочная установка**, показанное на рисунке 15, позволяющее выбрать компоненты Astra.IDE, которые не будут установлена на ПК в процессе установки.



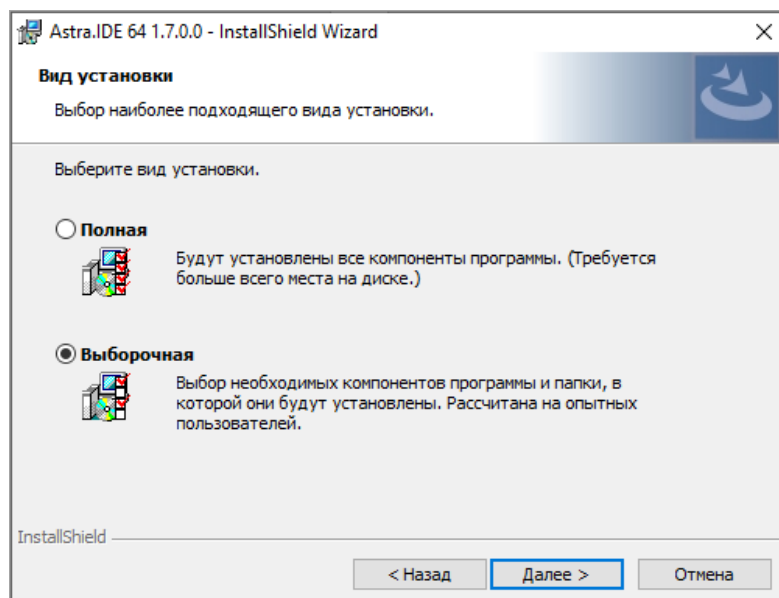


Рисунок 14 – Окно "Вид установки" мастера установки Astra.IDE

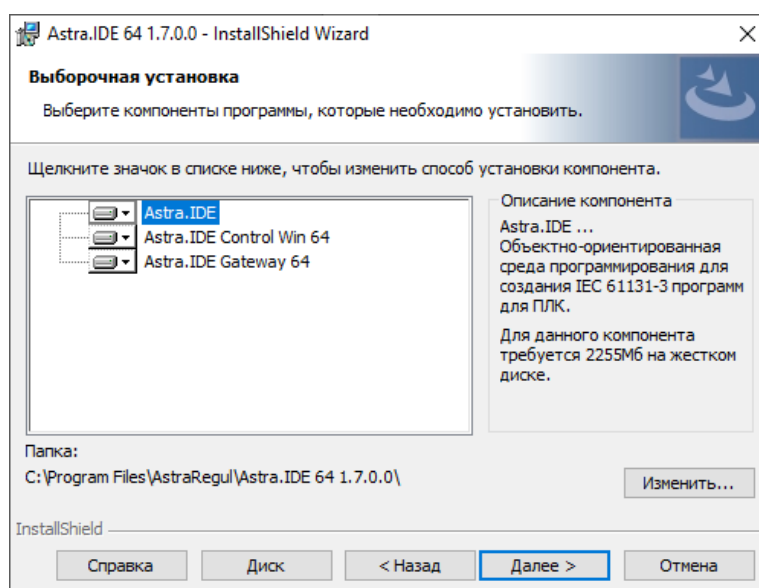


Рисунок 15 – Окно "Выборочная установка" мастера установки Astra.IDE

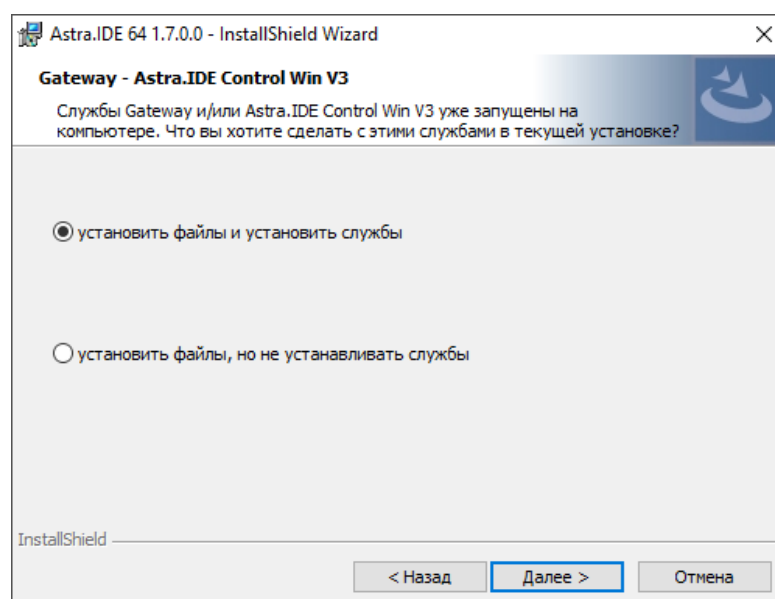



Рисунок 16 – Предложение об установке системных сервисов Astra.IDE


После выбора компонентов Astra.IDE, которые требуется или не требуется устанавливать в процессе установки, следует нажать кнопку **Далее**. Если в предыдущем окне мастера установки было предписано

установить эмулятор ПЛК Astra.IDE Control Win 64 и/или коммуникационный сервис взаимодействия среды разработки с ПЛК Astra.IDE Gateway 64, то при наличии на ПК ранее установленной IDE МЭК 61131-3 на экран будет выведено окно мастера установки **Gateway – Astra.IDE Control...** с предложением сделать выбор **установить файлы и установить службы** или **установить файлы, но не устанавливая службы**, показанное на рисунке 16.

При выборе варианта, предлагаемого по умолчанию, эмулятор ПЛК Astra.IDE Control Win 64 (наименование в оснастке services.msc – *CODESYS Control Win V3 – x64 Version 3.5.17.30*) и/или коммуникационный сервис взаимодействия среды разработки с ПЛК Astra.IDE Gateway 64 (наименование в оснастке services.msc – *CODESYS Gateway V3 Version 3.5.x.x*) после установки будут зарегистрированы в виде сервисных процессов Windows, при этом Astra.IDE Control Win 64 будет зарегистрирован в качестве сервиса, запускаемого по инициативе среды разработки Astra.IDE, а Astra.IDE Gateway 64 – в качестве сервиса, запускаемого автоматически при запуске Windows.

	<p>Рекомендуется использовать эмулятор ПЛК Astra.IDE Control Win 64 и/или коммуникационный сервис взаимодействия среды разработки с ПЛК Astra.IDE Gateway 64 в виде сервисов Windows.</p>
---	---

После нажатия **Далее** в предыдущем окне мастера установки на экран будет выведено окно **Настройки брандмауэра**, позволяющее отметить компоненты Astra.IDE, к которым требуется доступ с других ПК по сети.

	<p>Удаленный доступ с ПК по сети если и требуется, то, как правило, только для коммуникационного сервиса взаимодействия среды разработки с ПЛК Astra.IDE Gateway 64.</p>
--	--

После нажатия **Далее** в окне **Настройки брандмауэра** следует нажать **Установить** в окне **Готовность к установке программы** мастера установки.

Процесс установки Astra.IDE может занять 30 – 60 минут в зависимости от производительности ПК и типа используемого системного диска (HDD или SSD).


При успешном завершении установки на экран будет выведено окно **Программа InstallShield Wizard завершена**, в котором следует нажать **Готово**.

По завершении установки на рабочем столе Windows будут созданы два ярлыка: *Astra.IDE 1.7.0.0* и *IDE Selector*.

Ярлык *Astra.IDE 1.7.0.0* предназначен для запуска IDE МЭК 61131-3 Astra.IDE с профилем, предназначенным для работы с проектами ППО для контроллеров семейства REGUL.

Ярлык *IDE Selector* служит для ассоциации расширений файлов IDE МЭК 61131-3 с Astra.IDE и выбора профиля Astra.IDE, с которым будет запускаться Astra.IDE по двойному щелчку на файлах IDE МЭК 61131-3 с расширениями .project, .projectarchive и .library, а также для запуска Astra.IDE с требуемым профилем среды разработки.

После успешного завершения установки Astra.IDE следует запустить базовый или расширенный инсталлятор Fastwel PLC Application Toolkit.


	<p>После установки Fastwel PLC Application Toolkit в профиль <i>Fastwel PLC Application Toolkit – Astra.IDE V1.7.0.0</i> не следует использовать ярлык <i>Astra.IDE 1.m.n.k</i> на рабочем столе для запуска Astra.IDE с целью работы над проектами для контроллеров Fastwel.</p>
---	---



### 3.4. Проверка установки Fastwel PLC Application Toolkit

#### 3.4.1. Общие сведения

После завершения установки Fastwel PLC Application Toolkit для проверки правильности установки следует открыть один из примеров программирования, входящих в состав Fastwel PLC Application Toolkit, и выполнить компиляцию и/или генерацию кода в среде разработки IDE МЭК 61131-3.

	<p>При открытии проекта, созданного ранее и загруженного в контроллер, в среде разработки IDE МЭК 61131-3 более новой версии, чем использовалась при создании проекта, а также при отличии версий описаний целевых устройств и библиотек в открываемом проекте на экран выводится окно <b>Среда проекта</b>, аналогичное показанному на рисунке 18 или 23, с предложением сделать новейшими версии компилятора, устройств, библиотек и профиля визуализации.</p> <p>При нажатии кнопки <b>Сделать все новейшими</b> и генерации кода последующая установка связи между средой разработки и контроллером, в который ранее загружен данный проект, будет сопровождаться запросом среды разработки, как минимум, выполнить загрузку обновленного приложения методом онлайн изменения, либо, при существенных отличиях, полную загрузку обновленного приложения.</p> <p>В такой ситуации при необходимости соединения с контроллером с ранее загруженным приложением без загрузки обновленного приложения в IDE МЭК 61131-3 нажмите кнопку <b>Отмена</b> в окне <b>Среда проекта</b>.</p> <p>В IDE МЭК 61131-3 Astra.IDE или CODESYS версии от 3.5.17.0 и выше при наличии в проекте элементов визуализации, в том числе не целевой, потребуется обновить версии компилятора и профиля визуализации до текущих из состава используемой IDE МЭК 61131-3, что приведет к необходимости, как минимум, выполнить загрузку приложения онлайн-изменением при установлении соединения с контроллером с ранее загруженным приложением.</p>
---	--

#### 3.4.2. Проверка установки Fastwel PLC Application Toolkit с CODESYS V3

Для проверки установки Fastwel PLC Application Toolkit в IDE МЭК 61131-3 CODESYS V3 следует выполнить следующие действия:

1. В меню запуска программ выбрать **Fastwel – Пакет инструментальных средств ПЛК Fastwel – Примеры программирования – CPM723-01 – Учебный проект**, как показано на рисунке 17.

Если установка Fastwel PLC Application Toolkit выполнена успешно, на экране монитора будет отображено главное окно CODESYS V3 с открытым проектом tutorial.project и окном **Среда проекта**, как показано на рисунке 18.



Рисунок 17 – Открытие примера программирования в IDE МЭК 61131-3 CODESYS V3

2. Для обновления версий компилятора, описаний устройств, профиля визуализации и, возможно, библиотек в открытом проекте следует нажать кнопку **Сделать все новейшими**, а затем дважды **ОК**: для закрытия окна сообщения об обновлении версий различных частей проектной информации и для закрытия окна **Среда проекта**. Для сохранения без изменений версий компилятора, описаний устройств и профиля визуализации требуется нажать **Отмена** в окне **Среда проекта**.



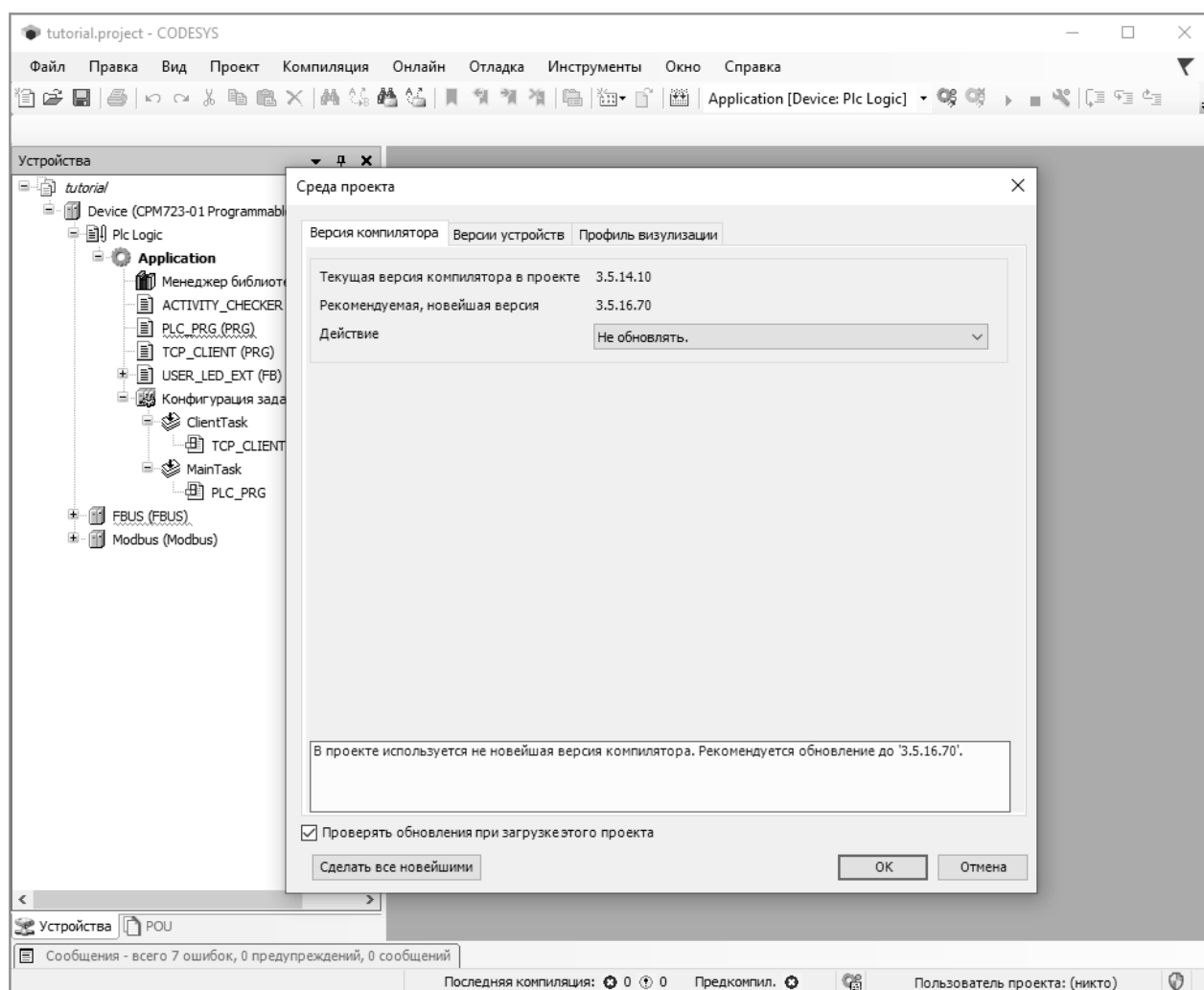
Версия профиля визуализации должна соответствовать версии компилятора.

При обновлении версии компилятора до новейшей в ранее созданном проекте требуется, как минимум, обновить до новейшей версию профиля визуализации.

При обновлении версии профиля визуализации до новейшей в ранее созданном проекте требуется, как минимум, обновить до новейшей версию компилятора.

3. В главном меню IDE МЭК 61131-3 выполнить команду **Компиляция – Генерировать код**. В панели **Сообщения**, расположенной в нижней части главного окна CODESYS V3, должно быть отображено порядка 7 сообщений, показано на рисунке 19, завершающихся строкой:

**Компиляция завершена – 0 ошибок, 0 предупреждений: готово к загрузке**



**Рисунок 18 – Главное окно IDE МЭК 61131-3 CODESYS V3 с открытым учебным проектом**

4. Для завершения работы со средой разработки IDE МЭК 61131-3 CODESYS V3 без сохранения изменений следует выполнить команду меню **Файл – Выход** и нажать **Нет** в диалоговой панели с предложением о сохранении изменений.

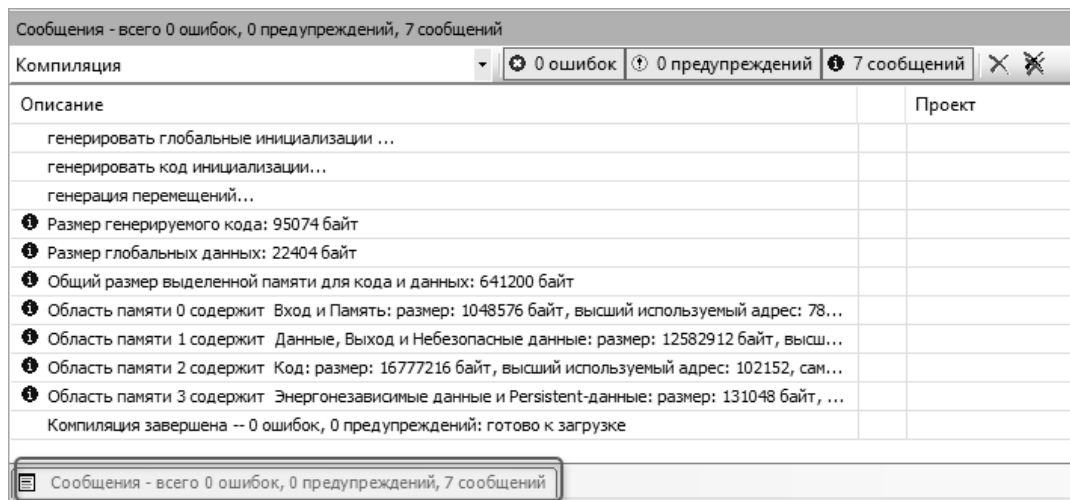


Рисунок 19 – Сообщения о выполнении и успешном завершении генерации кода

### 3.4.3. Проверка установки Fastwel PLC Application Toolkit с Astra.IDE

Если на ПК, помимо Astra.IDE, установлена IDE МЭК 61131-3 CODESYS V3, и при этом установка CODESYS V3 была произведена после (позднее) Astra.IDE, то станет невозможным открытие проектов в Astra.IDE по двойному щелчку мышью на ярлыках, созданных инсталлятором Fastwel PLC Application Toolkit в меню запуска Windows, т.к. проекты будут открываться в IDE МЭК 61131-3 CODESYS V3, установленной последней и выполнившей ассоциацию с файлами проектов и библиотек.

Для проверки установки Fastwel PLC Application Toolkit с Astra.IDE следует выполнить следующие действия:

1. На рабочем столе Windows дважды щелкнуть на ярлыке *IDE Selector* и в появившейся диалоговой панели **IDE Selector**, показанной на рисунке 20, нажать **Да**.

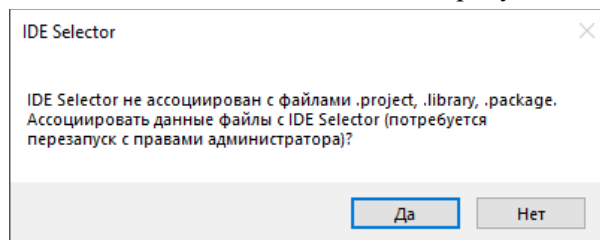


Рисунок 20 – Диалоговая панель при запуске IDE Selector

2. После подтверждения возможности запуска IDE Selector с правами администратора в появившемся окне **IDE Selector (Администратор)** в выпадающем списке **Выберите IDE** установить *Astra.IDE 1.x.x.x (1.x.x.x)*, в выпадающем списке **Профиль** выбрать *Fastwel PLC Application Toolkit – Astra.IDE V1.7.0.0*, отметить флажок **Использовать выбранное IDE по умолчанию и больше не спрашивать** и нажать **Заккрыть**, как показано на рисунке 21.

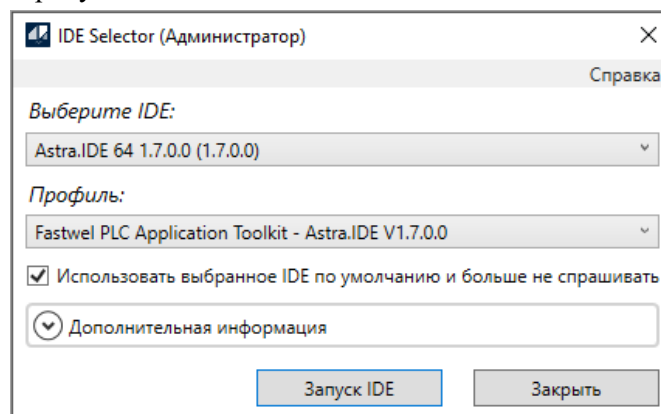


Рисунок 21 – Настройка запуска Astra.IDE по заданным ассоциациям с файлами IDE МЭК 61131-3

3. В меню запуска программ выбрать **Fastwel – Пакет инструментальных средств ПЛК Fastwel – Примеры программирования – CPM723-01 – Учебный проект**, как показано на рисунке 22.

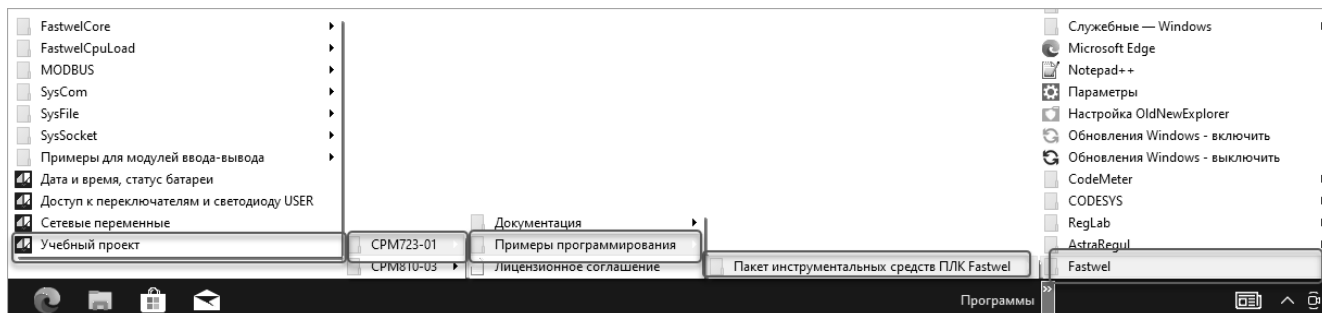


Рисунок 22 – Открытие примера программирования в IDE МЭК 61131-3 Astra.IDE

Если установка Fastwel PLC Application Toolkit выполнена успешно, на экране монитора будет отображено главное окно Astra.IDE с открытым проектом tutorial.project и окном **Среда проекта**, как показано на рисунке 23.

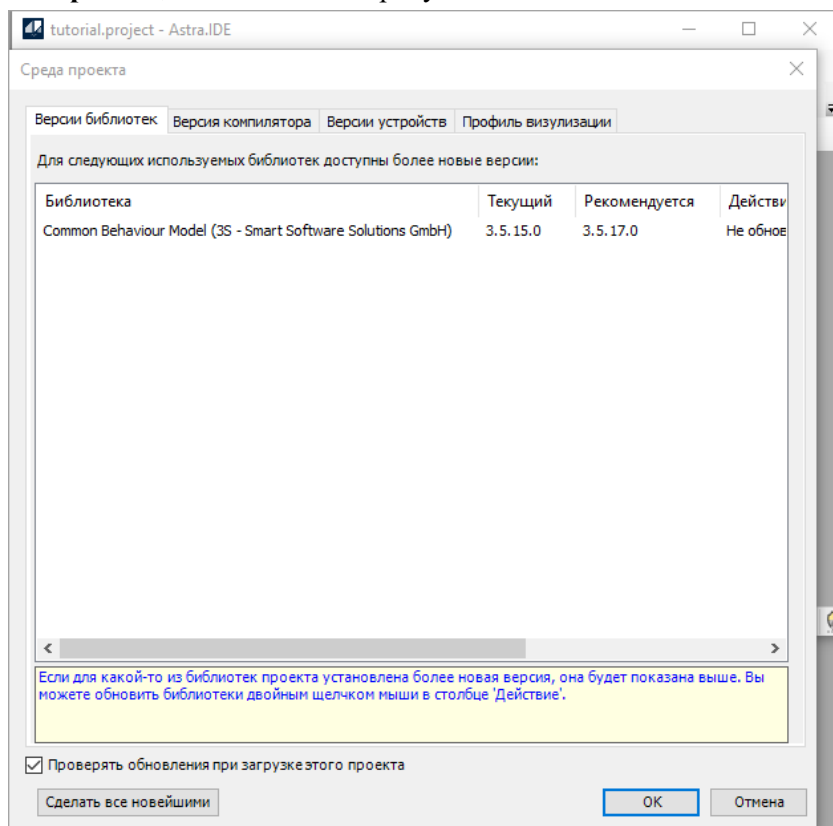



Рисунок 23 – Главное окно Astra.IDE после первого запуска с открытым учебным проектом

4. Для обновления версий компилятора, описаний устройств, профиля визуализации и, возможно, библиотек в открытом проекте следует нажать кнопку **Сделать все новейшими**, а затем дважды **ОК**: для закрытия окна сообщения об обновлении версий различных частей проектной информации и для закрытия окна **Среда проекта**.  
Для сохранения без изменений версий компилятора, описаний устройств и профиля визуализации необходимо нажать **Отмена** в окне **Среда проекта**.

	<p>Версия профиля визуализации должна соответствовать версии компилятора.</p> <p>При обновлении версии компилятора до новейшей в ранее созданном проекте требуется, как минимум, обновить до новейшей версию профиля визуализации.</p> <p>При обновлении версии профиля визуализации до новейшей в ранее созданном проекте требуется, как минимум, обновить до новейшей версию компилятора.</p>
---	---

5. Для отображения наиболее часто используемых рабочих областей представления проектной информации в главном окне Astra.IDE необходимо нажать следующие сочетания клавиш:

Alt+1 – для отображения вкладки **POU**;

Alt+0 – для отображения вкладки **Устройства** с древовидным списком устройств в проекте;

Alt+3 – для отображения области вывода сообщений среды разработки.

Внешний вид главного окна Astra.IDE после выполнения указанных действий показан на рисунке 24.

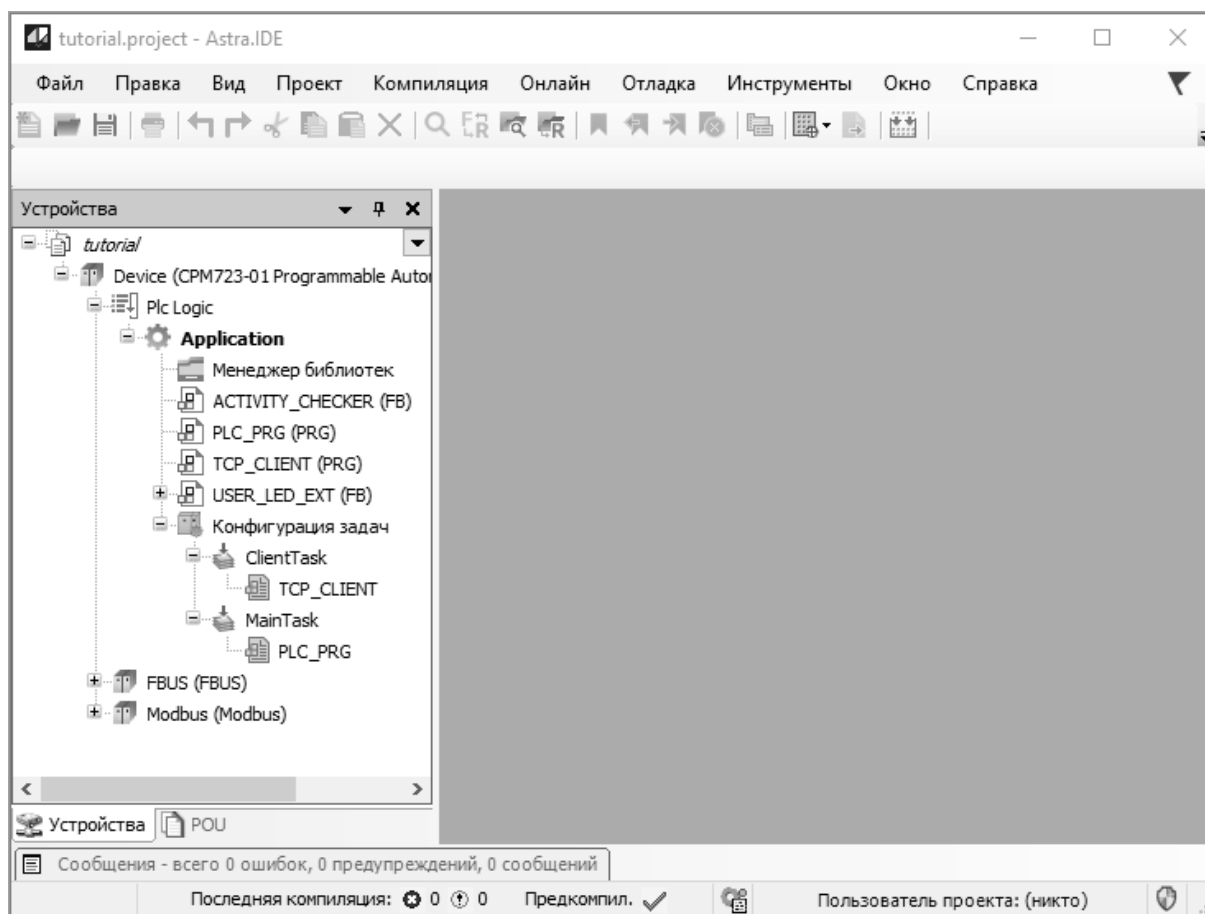


Рисунок 24 – Главное окно Astra.IDE с включенным отображением рабочих областей

6. В главном меню Astra.IDE выполнить команду **Компиляция – Генерировать код**. В панели **Сообщения**, расположенной в нижней части главного окна CODESYS V3, должно быть отображено порядка 7 сообщений, показано на рисунке 25, завершающихся строкой:

**Компиляция завершена – 0 ошибок, 0 предупреждений : готово к загрузке!**

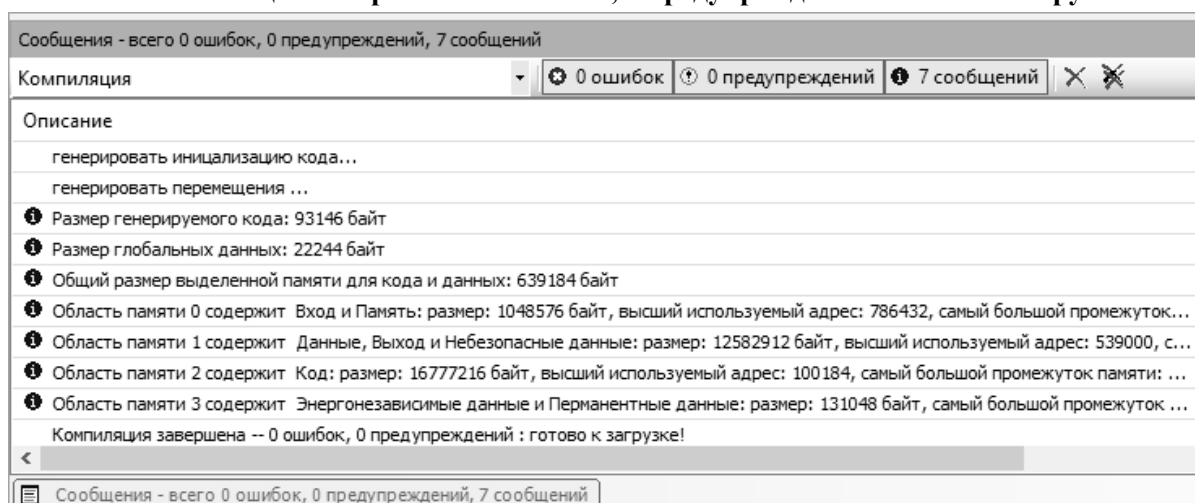



Рисунок 25 – Сообщения о выполнении и успешном завершении генерации кода Astra.IDE

7. Для завершения работы с Astra.IDE без сохранения изменений следует выполнить команду меню **Файл – Выход** и нажать **Нет** в диалоговой панели с предложением о сохранении изменений.

	<p>В IDE МЭК 61131-3 Astra.IDE или CODESYS версии от 3.5.17.0 и выше при наличии в проекте элементов визуализации, в том числе не целевой, при открытии проекта, созданного в более ранней версии IDE МЭК 61131-3, потребуется обновить версии компилятора и профиля визуализации до текущих из состава используемой IDE МЭК 61131-3.</p> <p>Рекомендуется делать обновление профиля визуализации и версии компилятора в окне <b>Установки проекта</b> по команде меню <b>Проект – Установки проекта</b>.</p> <p>Обновление версий в окне <b>Среда проекта</b> по кнопке <b>Сделать все новейшими</b> может привести к невозможности скомпилировать проект без ошибок.</p>
---	--

### 3.5. Настройка параметров связи между IDE МЭК 61131-3 и контроллером

#### 3.5.1. Общие сведения

Взаимодействие между IDE МЭК 61131-3 и контроллерами Fastwel для загрузки и отладки приложений, передачи файлов между ПК и контроллером, мониторинга переменных приложения и выполнения диагностических и сервисных операций осуществляется при помощи специального коммуникационного сервиса, устанавливаемого на ПК в процессе установки IDE МЭК 61131-3.

При установке Astra.IDE данный коммуникационный сервис называется Astra.IDE Gateway 64 (наименование в оснастке services.msc – *CODESYS Gateway V3 Version 3.5.x.x*).

При использовании CODESYS V3 коммуникационный сервис называется CODESYS Gateway (наименование в оснастке services.msc – *CODESYS Gateway V3 Version 3.5.x.x*).

Далее по тексту данный коммуникационный сервис будет называться IDE Gateway.


Значок сервиса  отображается в области уведомлений панели задач Windows, как показано на рисунке 26 (выделен контуром желтого цвета).



Рисунок 26 – Значок коммуникационного сервиса IDE МЭК 61131-3 в области уведомлений Windows

IDE Gateway обеспечивает взаимодействие IDE МЭК 61131-3 с одним или несколькими контроллерами с использованием специального протокола прикладного уровня поверх транспортных протоколов TCP, UDP, через последовательный порт интерфейса RS-232C и, при необходимости, через другие интерфейсы физического уровня.



Кроме того, IDE Gateway обеспечивает возможность удаленного доступа IDE МЭК 61131-3 на других ПК к контроллерам, находящимся в одной сети или непосредственно подключенным к ПК, на котором запущен IDE Gateway.

Для взаимодействия между IDE МЭК 61131-3 и контроллерами Fastwel могут использоваться коммуникационные порты RS-232C, сервисные порты USB и/или порты Ethernet контроллеров.

Для подключения порта интерфейса RS-232C контроллеров Fastwel к ПК для взаимодействия с IDE МЭК 61131-3 может использоваться кабель ACS00092-01 или аналогичный ноль-модемный кабель.

Для подключения сервисного порта интерфейса USB контроллеров Fastwel к порту USB ПК может использоваться кабель ACS00092-02 или аналогичный кабель USB A (m) – Mini-USB B (m).

Настоящий подраздел содержит указания по настройке параметров сервиса IDE Gateway для обеспечения возможности взаимодействия между IDE МЭК 61131-3 и контроллерами Fastwel на примере Astra.IDE.

	<p>После выбора сетевого пути к контроллеру при закрытии окна <b>Выбор устройства</b> нажатием <b>ОК</b>, показанного на рисунке 43 или 49, после сканирования сети или при начальной установке соединения IDE МЭК 61131-3 с контроллером на экран может быть выведено сообщение о завершении срока действия сертификата зашифрованного соединения среды разработки с контроллером, показанное на рисунке 27.</p> <p>При необходимости обновления данного сертификата следует нажать <b>Yes</b> и на открытой вкладке <b>Безопасность</b> создать в контроллере новый самоподписанный сертификат с требуемым сроком действия для категории <i>Encrypted Communication</i> нажатием .</p> <p>При отсутствии необходимости использования зашифрованного соединения среды разработки с контроллером необходимо нажать <b>No</b>.</p> <p>Для сохранения сделанного выбора следует отметить флажок <b>Remember decision</b>.</p>
---	--

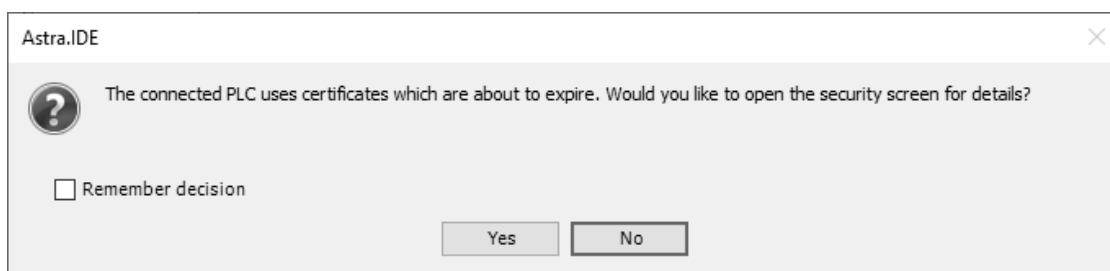



Рисунок 27 – Сообщение о завершении срока действия сертификата зашифрованного соединения среды разработки с контроллером

	<p>Для использования зашифрованного соединения между средой разработки и контроллером необходимо отметить флажок <b>Зашифрованное соединение</b> в меню <b>Устройство</b> на вкладке <b>Device – Установки соединения</b>, как показано на рисунке 28.</p>
---	--

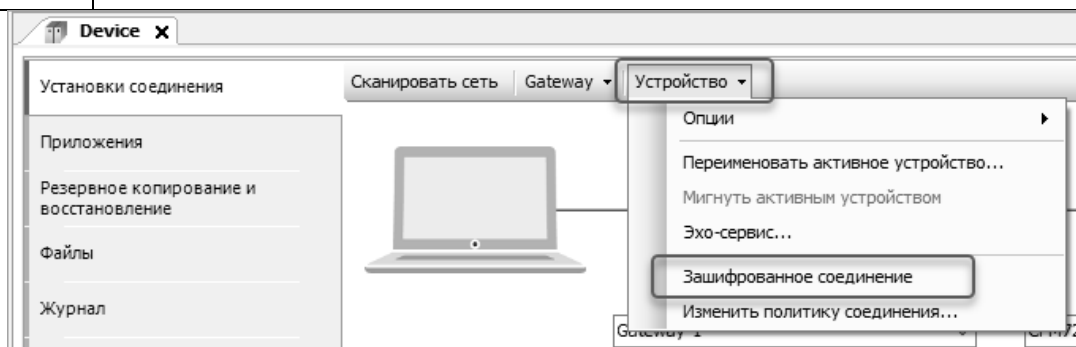



Рисунок 28 – Активация зашифрованного соединения среды разработки с контроллером

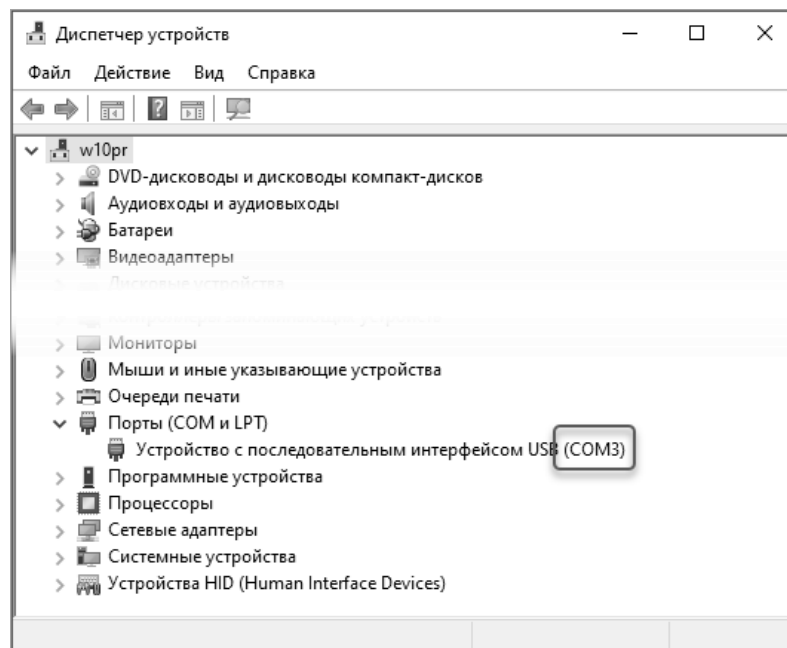
### 3.5.2. Настройка сервиса IDE Gateway для работы через сервисный USB-порт или порт интерфейса RS-232C контроллера

Перед настройкой IDE Gateway для взаимодействия с контроллером через сервисный порт USB требуется выполнить следующие действия:

1. Подключить сервисный USB-порт контроллера к USB-порту ПК кабелем ACS00092-02 или аналогичным USB A (m) – Mini-USB B (m).
2. Включить питание контроллера.
3. Через 10 – 15 с убедиться, что индикатор RUN или RUN/ERR контроллера светится зеленым цветом (непрерывно или прерывисто) или светится прерывисто, меняя цвет с красного на зеленый и погасая.
4. На ПК нажать сочетание клавиш +R и в диалоговой панели **Выполнить** ввести команду *devmgmt.msc*, нажать Enter и в окне **Диспетчер устройств** найти и запомнить

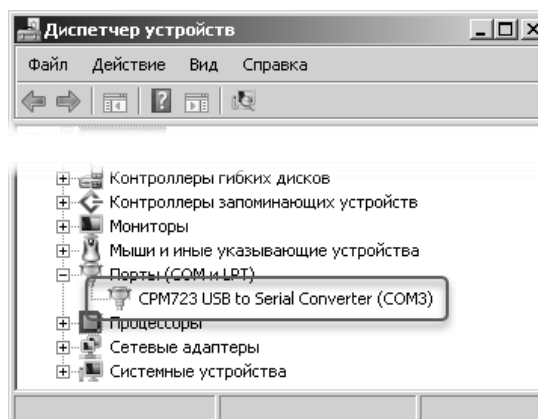


имя устройства с последовательным интерфейсом USB, созданного для сервисного порта USB контроллера, подключенного к ПК, как показано на рисунке 29.



**Рисунок 29 – Имя устройства с последовательным интерфейсом для сервисного порта USB контроллера в Windows 10**

На ПК с операционной системой Windows 7 Service Pack 1, Windows 8/8.1 имя устройства будет содержать наименование контроллера, сервисный порт USB которого подключен к порту интерфейса USB ПК, как показано на рисунке 30.



**Рисунок 30 – Имя устройства с последовательным интерфейсом для сервисного порта USB контроллера в Windows 7**

Настройка IDE Gateway для связи с контроллером через сервисный порт USB или последовательный порт интерфейса RS-232C выполняется следующим образом:

1. Отключить контроллер от сети Ethernet для гарантии связи между контроллером и IDE МЭК 61131-3 только через сервисный USB-порт или порт RS-232C контроллера.



Если контроллер одновременно подключен к сети Ethernet и к USB-порту или к порту интерфейса RS-232C ПК, то, с высокой вероятностью, сервис IDE Gateway при поиске совместимых устройств будет использовать сеть Ethernet.

2. Включить питание контроллера, если это не было сделано ранее.  
Если подключение контроллера к данному USB-порту ПК выполняется впервые, произойдет активация и установка драйвера соответствующего устройства. Для ускорения процесса установки в операционных системах Windows 7 и Windows 8/8.1 следует щелкнуть на ссылке **Пропустить загрузку драйвера из центра обновления Windows**, как показано на рисунке 31.



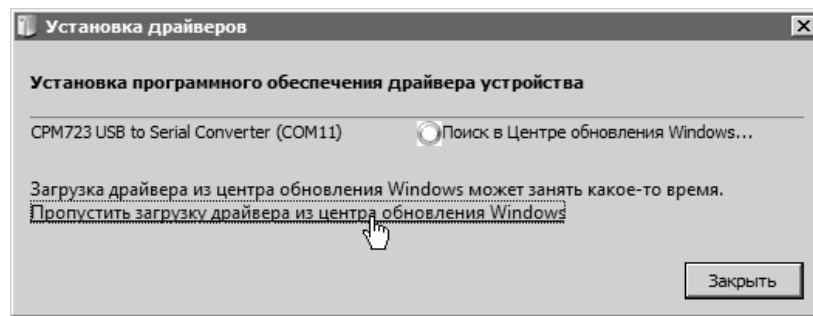


Рисунок 31 – Пропуск поиска драйвера в центре обновления Windows 7

3. Определить имя последовательного порта ПК, соответствующего сервисному порту контроллера в **Диспетчере устройств** Windows в соответствии с приведенными выше указаниями, и запомнить числовой суффикс после префикса *COM* имени устройства (например, для COM3 числовой суффикс 3), который будет использоваться при настройке параметров IDE Gateway.

При последующих подключениях сервисного порта контроллера к тому же USB-порту ПК в операционной системе будет появляться последовательный порт с тем же именем.



При подключении сервисного порта USB контроллера к другому USB-порту ПК последовательный порт будет получать имя с другим числовым суффиксом.

4. В IDE МЭК 61131-3 открыть один из проектов для контроллера используемого типа из поставляемых в составе Fastwel PLC Application Toolkit примеров программирования, в соответствии с указаниями п. 3.4, либо создать новый проект для платформы, соответствующей типу используемого контроллера.
5. В дереве проекта на вкладке **Устройства** дважды щелкнуть левой кнопкой мыши на элементе *Device (CPMXXX-ZZ Programmable Automation Controller)*. В правой области главного окна IDE МЭК 61131-3 будет отображен редактор устройства, открытый на вкладке **Device – Установка соединения**, как показано на рисунке 32.
6. На вкладке **Device–Установка соединения** выбрать команду меню **Gateway–Конфигурация локального Gateway...**, как показано на рисунке 33.  
На экран монитора будет выведено окно конфигурации IDE Gateway, показанное на рисунке 34.

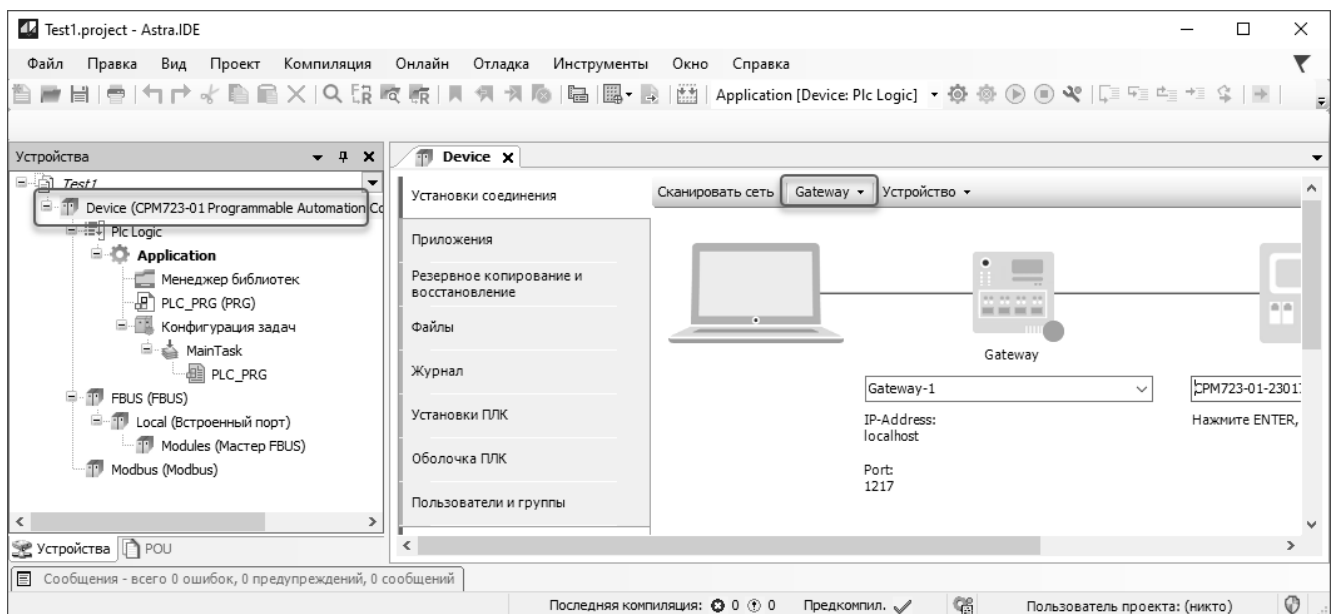


Рисунок 32 – Вкладка Установки соединения редактора устройства

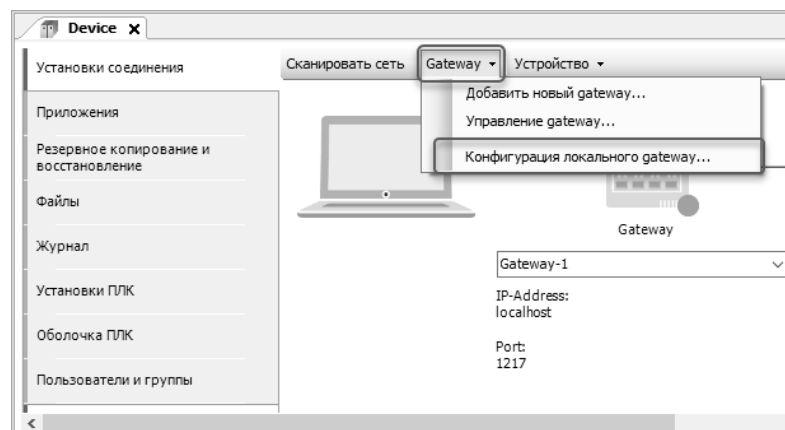


Рисунок 33 – Открытие окна конфигурации IDE Gateway

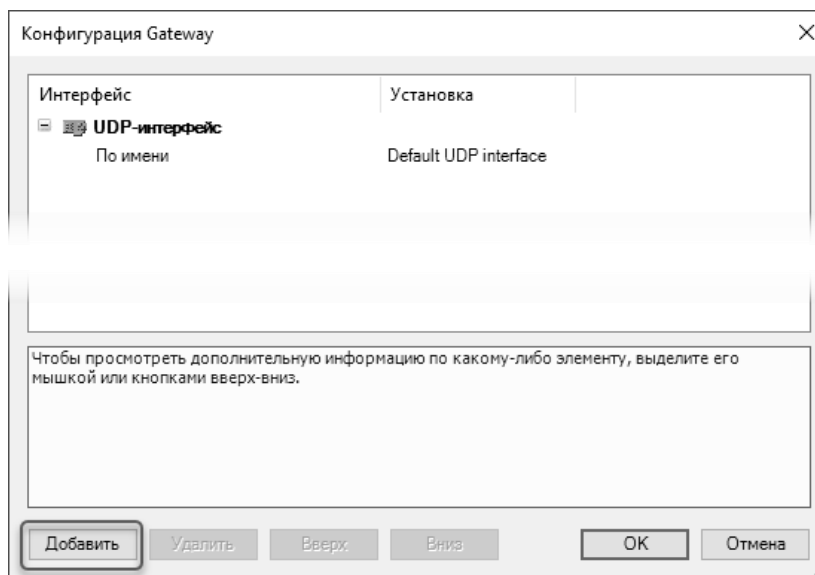


Рисунок 34 – Окно конфигурации IDE Gateway

7. В окне Конфигурация Gateway нажать кнопку **Добавить**, затем выбрать команду меню **Добавить интерфейс...** и нажать Enter. В списке интерфейсов появится новый элемент **СОМ-порт** с параметрами по умолчанию, показанными на рисунке 35.

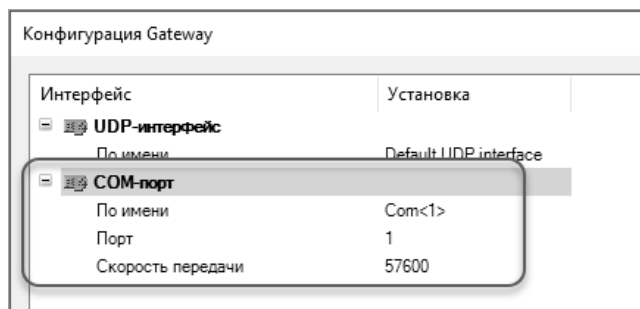


Рисунок 35 – Описание интерфейса СОМ-порт с параметрами по умолчанию

8. Дважды щелкнуть левой кнопкой мыши в ячейке **СОМ-порт–По имени:Установка** и ввести имя интерфейса, как показано на рисунке 36, и нажать Enter. Имя может состоять из символов латинского алфавита и/или цифр и символов подчеркивания и должно отличаться от имен других интерфейсов, введенных в качестве параметра **По имени**.
9. Дважды щелкнуть левой кнопкой мыши в ячейке **СОМ-порт–Порт:Установка** и ввести числовой суффикс имени последовательного порта, присвоенного сервисному порту USB подключенного контроллера при выполнении операции 4 вышеприведенных или операции 3 настоящих указаний или используемого для связи ПК с сервисным портом RS-232C контроллера, как показано на рисунке 37, после чего нажать Enter. Например, для порта *COM3* необходимо ввести *3*.

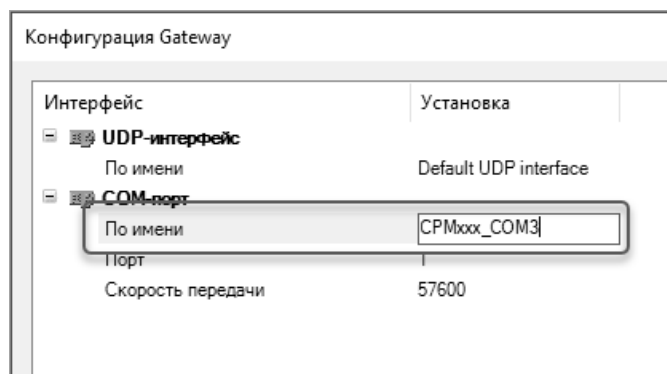


Рисунок 36 – Ввод имени интерфейса IDE Gateway

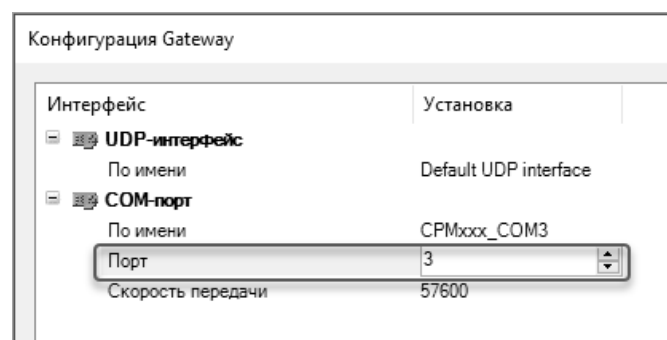


Рисунок 37 – Ввод номера последовательного порта

10. Дважды щелкнуть левой кнопкой мыши в ячейке **COM-порт–Скорость передачи**: *Установка*, ввести **115200** и нажать Enter.
11. Щелкнуть левой кнопкой мыши на элементе **COM-порт** и выбрать команду **Добавить параметр конфигурации** в контекстном меню, как показано на рисунке 38.  
В столбце *Интерфейс* под параметром **COM-порт–Порт** появится название нового параметра в выпадающем списке. Если добавляемый параметр имеет имя **Авто-адресация**, нажать Enter, в противном случае выбрать в выпадающем списке параметр **Авто-адресация** и нажать Enter. Значение данного параметра должно быть равно **TRUE**, как показано на рисунке 39.

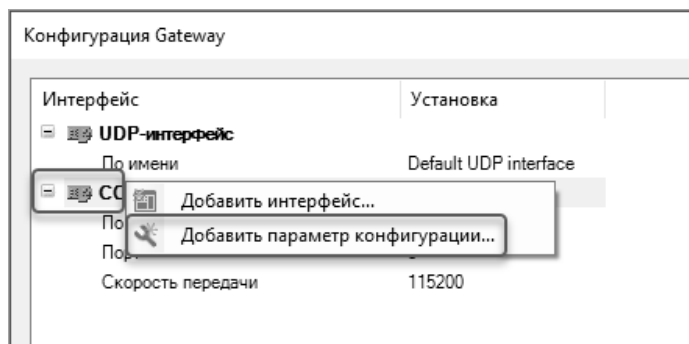


Рисунок 38 – Добавление параметра конфигурации для интерфейса COM-порт

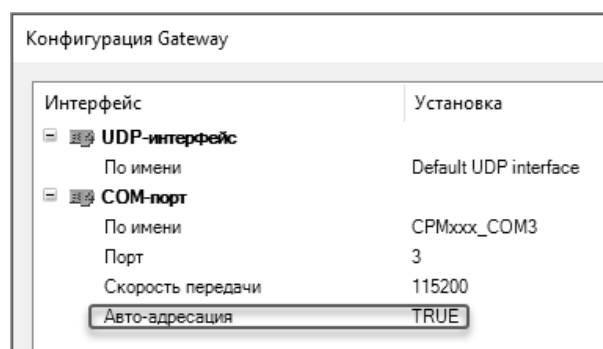


Рисунок 39 – Параметр Авто-адресация интерфейса COM-порт

12. Выбрать интерфейс **COM-порт**, после чего нажимать кнопку **Вверх**, расположенную в нижней части окна **Конфигурация Gateway**, до тех пор, пока интерфейс **COM-порт** со всеми относящимися к нему параметрами не окажется в самой верхней позиции списка, как показано на рисунке 40.

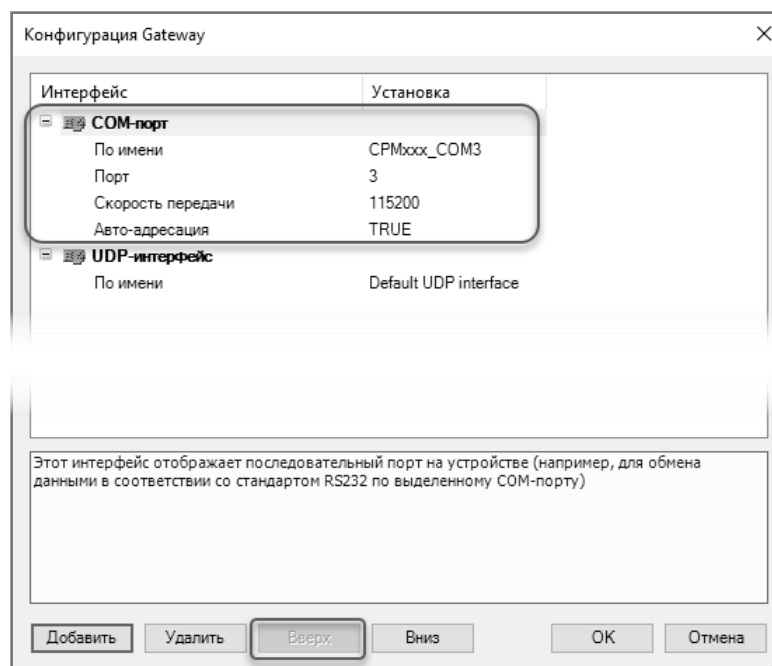



Рисунок 40 – Интерфейс COM-порт в верхней позиции списка интерфейсов IDE Gateway

13. Закрыть окно **Конфигурация Gateway** нажатием кнопки **OK**.
14. Остановить сервис IDE Gateway, для чего щелкнуть над его значком  в области уведомлений панели задач Windows и выбрать команду **Stop Gateway** в появившемся контекстном меню, как показано на рисунке 41. После успешного останова значок сервиса будет отображаться градациями серого.

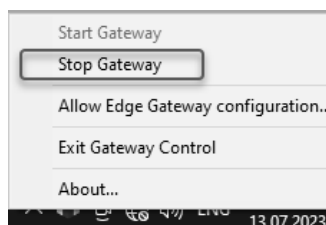



Рисунок 41 – Останов IDE Gateway

15. Повторно запустить сервис IDE Gateway, для чего перейти в окно IDE МЭК 61131-3, щелкнув внутри области вкладки **Device–Установки соединения**, и подождать 5-10 секунд, либо выполнить команду **Start Gateway** в контекстном меню над значком сервиса  в области уведомления панели задач Windows.

Значок вновь запущенного сервиса будет отображаться в области уведомлений с использованием красного и светло-серого цветов, а в правом нижнем углу изображения сервиса на вкладке **Device – Установки соединения** будет отображен символ круглой формы зеленого цвета, как показано на рисунке 42.

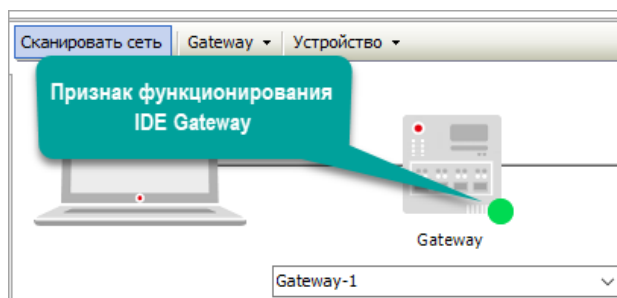



Рисунок 42 – Признак функционирования сервиса IDE Gateway

16. Выполнить команду **Сканировать сеть...** в меню вкладки **Device–Установки соединения**. На экран монитора будет выведено окно **Выбор устройства**. Если предыдущие указания были выполнены правильно, в списке устройств будет отображен элемент *CPMXXX-ZZ-YUММХХХNNNN [0001]*, соответствующий контроллеру, подключенному к ПК, как показано на рисунке 43, где *XXX* – числовой суффикс модели, а *ZZ* – числовой суффикс исполнения, например CPM723-01-23017238864.
- Часть имени элемента *xxxNNNN* совпадает с серийным номером контроллера, что позволяет убедиться, что настроено и проверено соединение именно с требуемым контроллером.
- Часть имени *YUММ* соответствует году и месяцу производства контроллера, если он выпущен до 2021 г. Начиная с 2021 г., часть имени соответствует году и кварталу производства контроллера.
- Например, на рисунке 43 показано обнаруженное устройство с серийным номером 723.8864, произведенное в первом квартале 2023 г.
- Значение *[0001]* свидетельствует об использовании сервисом IDE Gateway одноуровневой адресации удаленных устройств, что характерно для взаимодействия с устройством, непосредственно подключенным к ПК, на котором функционирует сервис IDE Gateway.
17. В списке **Выберите сетевой путь** щелкнуть на элементе, соответствующем контроллеру, с которым требуется установить связь.

	<p>Нажатие кнопки <b>Помогать</b> в окне <b>Выбор устройства</b> приводит к изменению состояния индикатора "USER" или "USR" контроллера, к которому выбран сетевой путь в окне <b>Выбор устройства</b>.</p>
---	---

В правой области окна **Выбор устройства** будут отображены полное имя устройства, адрес устройства в сети, идентификатор целевой платформы (**ID таргета**), версия системного программного обеспечения (**Версия таргета**), название целевой платформы (**Имя таргета**), название производителя (**Производитель таргета**) и др.

18. Для закрытия окна **Выбор устройства** следует нажать кнопку **ОК**. Информация о выбранном сетевом пути к контроллеру будет отображена на вкладке **Device–Установки соединения**, как показано на рисунке 44.

Более подробная информация о сервисе IDE Gateway приведена в справочной системе IDE МЭК 61131-3 (в разделе **Руководство по интерфейсу пользователя – Объект 'Устройство' и Общий редактор устройств – Вкладка 'Установки соединения'** русскоязычного варианта справочной системы).

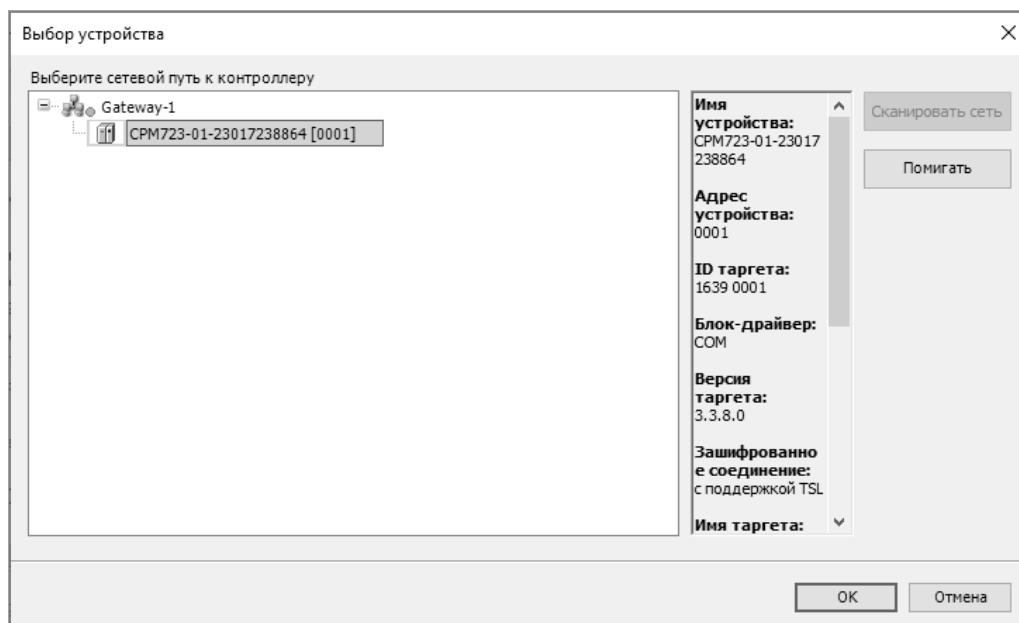


Рисунок 43 – Окно сканирования сети

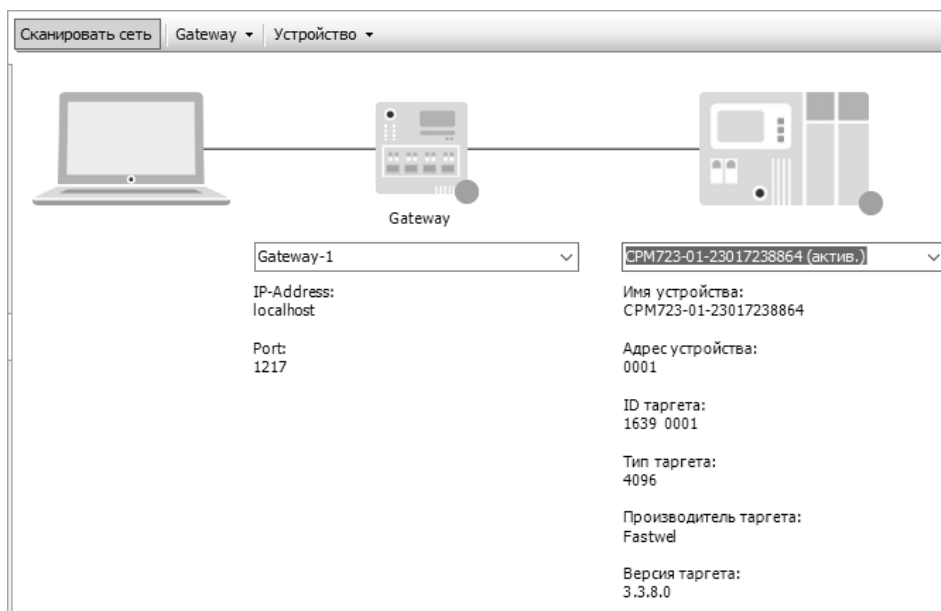


Рисунок 44 – Информация о контроллере, выбранном для последующего сетевого взаимодействия



Если Fastwel PLC Application Toolkit и IDE МЭК 61131-3 установлены в Windows, функционирующей под управлением виртуальной среды Oracle® VM VirtualBox, перенаправление интерфейсов USB в гостевую Windows должно производиться в режиме **Контроллер USB 2.0 (EHCI)**, установленном в настройках USB виртуальной машины Windows.

### 3.5.3. Настройка сервиса IDE Gateway для работы по сети Ethernet

#### 3.5.3.1. Общие сведения

IDE МЭК 61131-3 имеет в своем составе развитые средства сетевого взаимодействия как между средой разработки и контроллерами, так и непосредственно между контроллерами по сети Ethernet с использованием специализированного протокола прикладного уровня поверх стандартных протоколов транспортного уровня UDP и TCP.

Настоящий подраздел содержит основные указания по настройке коммуникационного сервиса IDE Gateway для взаимодействия с контроллерами по сети Ethernet.

Информация о принципе работы сетевых средств IDE МЭК 61131-3 приведена в разделе **Использование управляющих сетей** интерактивной справочной системы среды разработки.

После установки Fastwel PLC Application Toolkit и IDE МЭК 61131-3 на ПК коммуникационный сервис IDE Gateway готов к взаимодействию между средой разработки и контроллером по сети Ethernet.

На рисунке 45 показано содержимое окна **Конфигурация Gateway**, отображаемого на экране по команде меню **Gateway–Конфигурация локального gateway** на вкладке **Device–Установки соединения** после установки среды разработки.

Интерфейс **UDP-интерфейс** с именем *Default UDP interface* используется для поиска в сети контроллеров, поддерживающих протокол прикладного уровня IDE МЭК 61131-3 поверх UDP, а также для обмена данными и командами между средой разработки и контроллерами по протоколу UDP. При использовании протокола UDP, как минимум, требуется знать, в какой подсети находятся контроллеры, и доступна ли эта подсеть на данном ПК. При этом не нужно знать IP-адреса удаленных контроллеров, поскольку поиск узлов сети выполняется автоматически с использованием системы адресации сетевого протокола IDE МЭК 61131-3 и направленных широковещательных сообщений.

В конфигурацию IDE Gateway также входит скрытый **TCP-интерфейс**, который предназначен для обмена данными и командами с контроллерами, поддерживающими протокол прикладного уровня IDE МЭК 61131-3 поверх транспортного протокола TCP. При использовании интерфейса TCP требуется знать IP-адрес (или сетевое имя) каждого контроллера, с которым предполагается взаимодействовать по сети Ethernet.

Для выполнения последующих указаний потребуется хотя бы один контроллер Fastwel (CPM723-01, CPM810-03 и т.п.), подключенный к сети Ethernet, к которой также подключен ПК с установленной IDE МЭК 61131-3 и Fastwel PLC Application Toolkit.

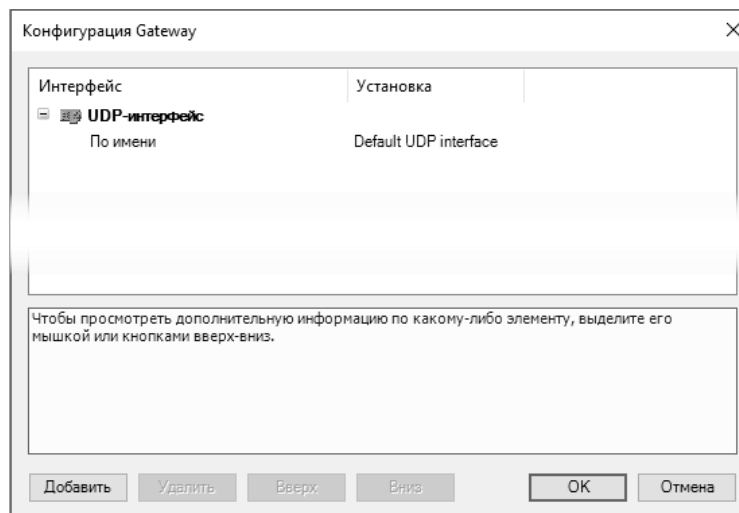


Рисунок 45 – Исходная конфигурация сервиса IDE Gateway после установки IDE МЭК 61131-3

	<p>Сетевое имя (имя хоста) контроллера определяется системным параметром <b>Имя хоста (Host name)</b> на странице <b>Параметры сети (Network Parameters)</b> встроенного веб-конфигуратора контроллера, доступ к которому осуществляется при помощи веб-браузера. Требования к веб-браузеру приведены в п. 0.</p> <p>Если для контроллера задано сетевое имя (параметр <b>Имя хоста</b>), то оно будет отображаться в окне <b>Выбор устройства</b>, выводимом на экран при сканировании сети.</p>
--	---

### 3.5.3.2. Связь с контроллерами через TCP-интерфейс

Для создания пути связи с контроллером через TCP-интерфейс в проекте для платформы, соответствующей типу используемого контроллера, открытому в IDE МЭК 61131-3, следует перейти на вкладку устройства **Установки соединения**, ввести IP-адрес или сетевое имя, определяемое системным параметром **Имя хоста**, контроллера в поле адреса удаленного устройства, как показано на рисунке 46, и нажать Enter.

Если введен правильный IP-адрес или сетевое имя контроллера, и контроллер доступен по сети с данного ПК, то через некоторое время под полем ввода IP-адреса будет отображена информация о контроллере, представленная на рисунке 47, а в правом нижнем углу изображения контроллера появится символ круглой формы зеленого цвета ●, свидетельствующий о наличии активного пути связи с контроллером.

Таким образом, при установке связи с контроллером через TCP-интерфейс должно быть решено два вопроса:

1. Какой IP-адрес (какие IP-адреса) имеет контроллер.
2. Доступен ли контроллер по сети с данного ПК, т.е. могут ли быть доставлены IP-пакеты с данного ПК до контроллера.

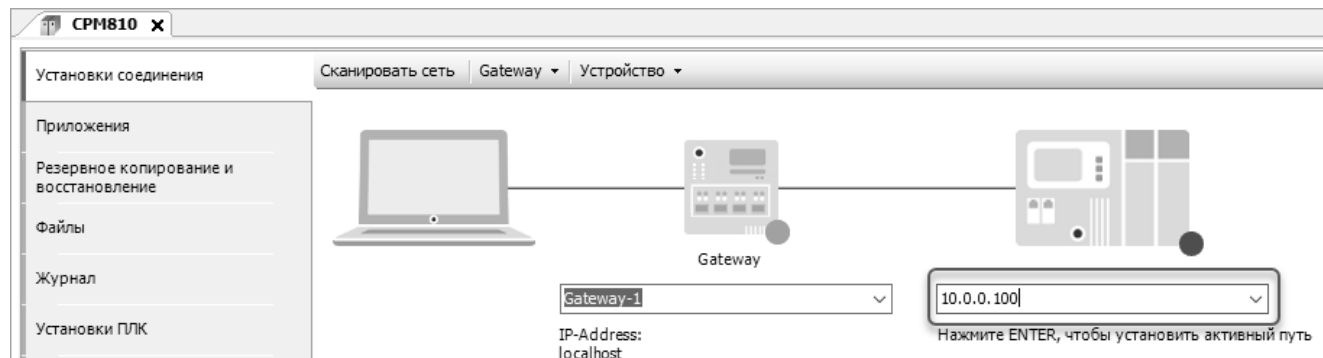
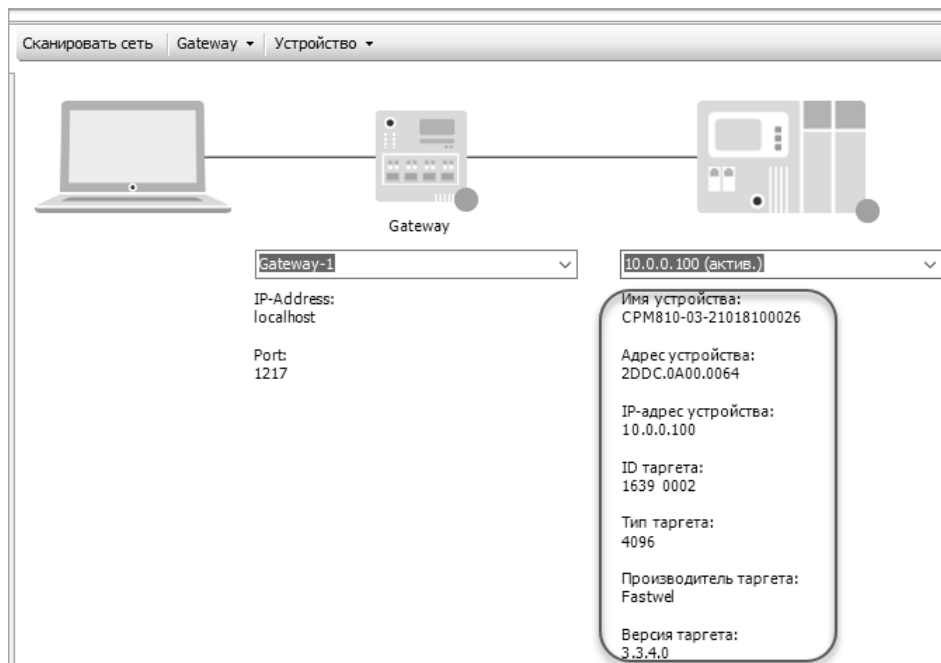


Рисунок 46. Ввод IP-адреса для создания пути через TCP-интерфейс



**Рисунок 47 – Информация о контроллере по заданному IP-адресу**


Описание способов определения IP-параметров контроллера приведено в документации на используемый контроллер.

Если контроллер только что извлечен из упаковки или запущен в режиме с заводскими параметрами (все десять переключателей включены), то его первые два сетевых интерфейса имеют следующие параметры:

IP-адрес интерфейса LAN1: 10.0.0.100/8

IP-адрес интерфейса LAN2: 10.0.0.101/8

Для проверки доступности контроллера по сети с ПК:

1. Нажать сочетание клавиш -R.
2. В поле **Открыть** появившегося окне **Запуск программы** ввести *cmd.exe* и нажать Enter.
3. В появившемся окне *cmd.exe* ввести команду *ping <IP-адрес контроллера>*. Например, если контроллер имеет IP-адрес 10.0.0.100, то следует ввести:  
**ping 10.0.0.100**
4. Нажать Enter и убедиться, что в окне *cmd.exe* отображаются строки, аналогичные следующим:  
**Ответ от 10.0.0.100: число байт=32 время<1мс TTL=128**  
**Ответ от 10.0.0.100: число байт=32 время<1мс TTL=128**  
**Ответ от 10.0.0.100: число байт=32 время<1мс TTL=128**
5. Если в окне *cmd.exe* отображаются строки, аналогичные следующим:  
**Ответ от 10.0.0.252: Заданный узел недоступен.**  
**Превышен интервал ожидания для запроса.**  
**Превышен интервал ожидания для запроса.**  
**Превышен интервал ожидания для запроса.**  
то либо контроллер имеет другой IP-адрес, либо ПК или контроллер (или оба) не подключены к сети, в которой определен IP-адрес искомого контроллера.
6. В окне *cmd.exe* ввести команду *ipconfig* и убедиться, что среди перечисленных сетевых адаптеров ПК имеется адаптер с IPv4-адресом, принадлежащим той же подсети, в которой находится IP-адрес искомого контроллера. Например, если производится поиск в сети контроллера с исходными заводскими настройками LAN1:10.0.0.100/8, LAN2:10.0.0.101/8, то на ПК должен присутствовать адаптер, физически подключенный к той же подсети, что и контроллер, и имеющий IP-адрес в диапазоне от 10.0.0.1 до 10.0.0.254, исключая 10.0.0.100 и 10.0.0.101, и маску подсети 255.0.0.0:



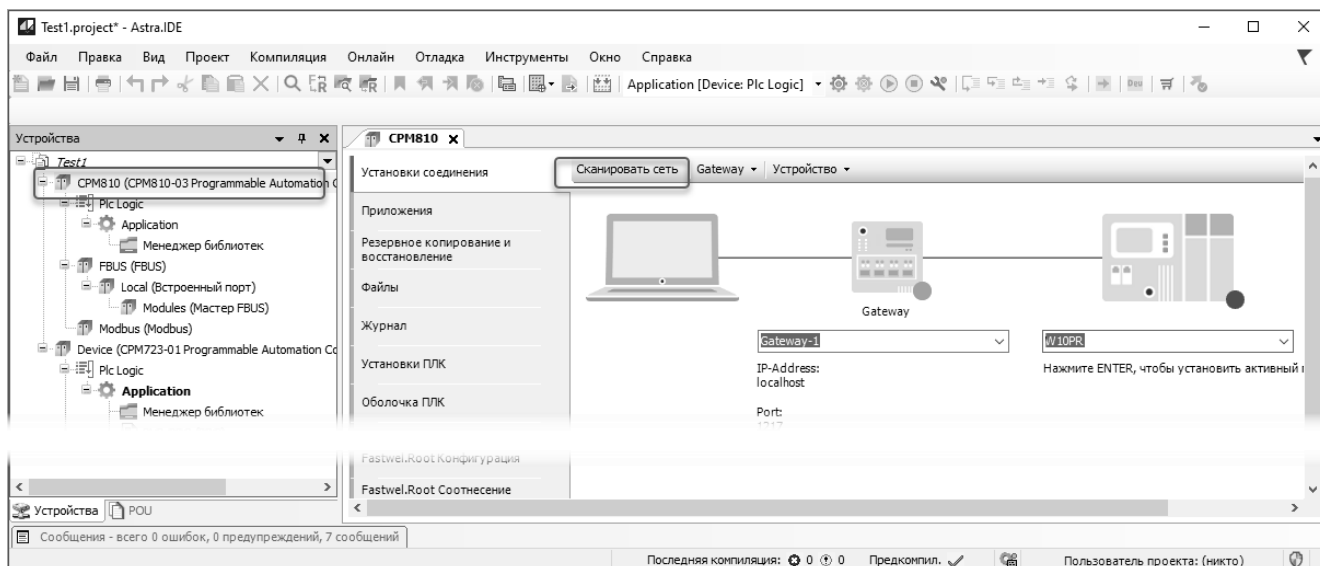
**Ethernet adapter Подключение по локальной сети:**

DNS-суффикс подключения . . . . . :  
 Локальный IPv6-адрес канала . . . : fe80::51df:fc27:7e58:7c84%13  
 IPv4-адрес . . . . . : 10.0.0.252  
 Маска подсети . . . . . : 255.0.0.0  
 Основной шлюз . . . . . :

**3.5.3.3. Связь с контроллерами через UDP-интерфейс**

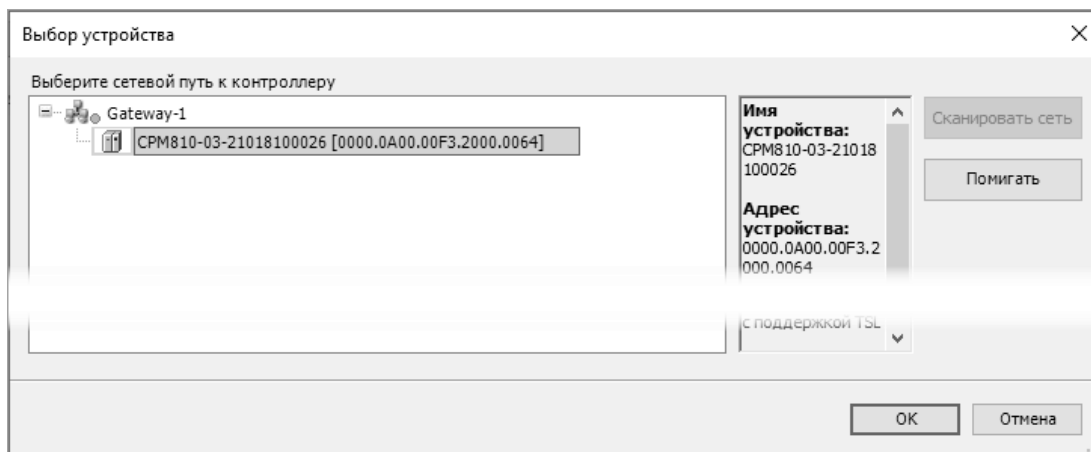
Для создания сетевого пути связи с контроллером через UDP-интерфейс:

1. В открытом проекте для платформы, соответствующей типу контроллера, с которым предполагается установить связь, перейти на вкладку устройства **Установки соединения** и выполнить команду **Сканировать сеть**, как показано на рисунке 48.



**Рисунок 48 – Запуск поиска устройств, поддерживающих протокол IDE МЭК 61131-3**

На экран монитора будет выведено окно **Выбор устройства** со списком найденных контроллеров, как показано на рисунке 49.



**Рисунок 49 – Успешный результат поиска устройств**

Контроллерам CPMXXX-ZZ соответствуют элементы списка с именами CPMXXX-ZZ-YUММХХХNNNN [AAAA.BBBB...]. Часть имени элемента XXXNNNN совпадает с серийным номером контроллера.

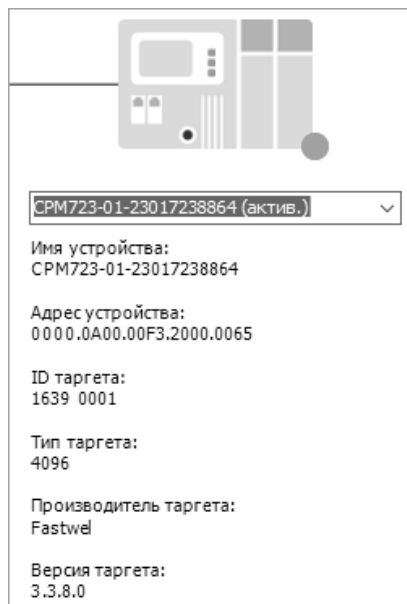
Часть имени YUММ соответствует году и месяцу производства контроллера, если он выпущен до 2021 г. Начиная с 2021 г., часть имени соответствует году и кварталу производства контроллера.

Например, на рисунке 49 показано устройство с серийным номером 810.0026, произведенное в первом квартале 2021 г.

Значение адреса [0000.0A00.00F3.2000.0064] свидетельствует об использовании сервисом IDE Gateway иерархической адресации удаленных устройств, что характерно для взаимодействия по сети Ethernet с использованием протокола UDP.

Более подробная информация об адресации, используемой в протоколе IDE МЭК 61131-3, приведена в подразделе **Использование управляющих сетей – Сеть и адресация: Адресация и маршрутизация и Структура адресов** справочной системы IDE МЭК 61131-3.

2. Выбрать требуемое устройство, щелкнув на соответствующем элементе в списке найденных устройств, и нажать **ОК**. Окно **Выбор устройства** будет закрыто, а на вкладке **Установки соединения** появится информация о созданном сетевом пути, по содержанию аналогичная показанной на рисунке 50: полное имя устройства, адрес в сети IDE МЭК 61131-3, идентификатор целевой платформы (**ID таргета**), версия системного программного обеспечения (**Версия таргета**), название целевой платформы (**Имя таргета**), название производителя (**Производитель таргета**) и др.



**Рисунок 50 – Информация об активном пути связи с контроллером**

Если по завершении поиска устройств окно **Выбор устройства** не содержит ни одного найденного устройства, то следует убедиться, что искомый контроллер доступен в сети, к которой подключен ПК.

Если предполагается, что контроллер и ПК подключены к одному и тому же физическому сегменту сети Ethernet и находятся в одной и той же IP-подсети, то следует проверить, что контроллер доступен на данном ПК и в данной подсети разрешена передача направленных широковещательных сообщений.

Если известен IP-адрес контроллера, то следует выполнить проверку доступности контроллера по сети на ПК в соответствии с указаниями п. 3.5.3.2.

Если контроллер только что извлечен из упаковки или запущен в режиме с заводскими параметрами (все десять переключателей включены), то его первые два сетевых интерфейса имеют следующие параметры:

IP-адрес интерфейса LAN1: 10.0.0.100/8

IP-адрес интерфейса LAN2: 10.0.0.101/8

Если контроллер отвечает на ping, но его не удастся найти по команде **Сканировать сеть...** на вкладке **Установки соединения**, то для проверки возможности передачи направленных широковещательных сообщений обратитесь к администратору сети.

При изложении указаний настоящего подраздела предполагается, что контроллеры и ПК находятся в одной подсети. Пример организации такой сети, в которую входят ПК с IP-адресом 10.0.0.252/8 и два контроллера CPM810-03 с установленными при помощи переключателей IP-адресами 10.0.0.3/8, 10.0.0.4/8 и 10.0.0.5/8, 10.0.0.6/8, представлен на рисунке 51. В данном случае оба контроллера доступны по сети на данном ПК, и результат сетевого поиска в окне **Выбор устройства** будет выглядеть, как показано на рисунке 49.

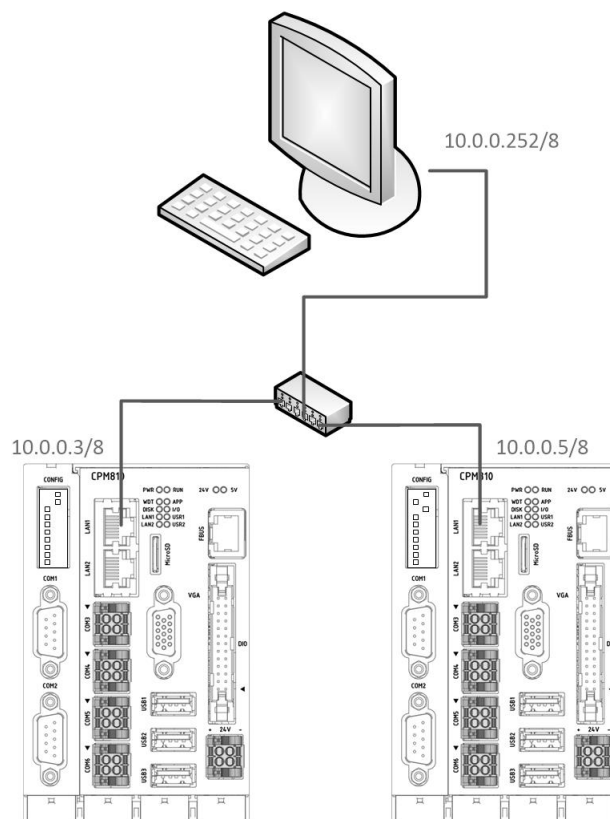


Рисунок 51 – Компьютер и два контроллера в подсети 10.0.0.0/8

На практике иногда требуется использовать иерархические сетевые топологии, в которых не все контроллеры доступны по сети на ПК, где функционирует сервис IDE Gateway. Пример такой топологии представлен на рисунке 52.

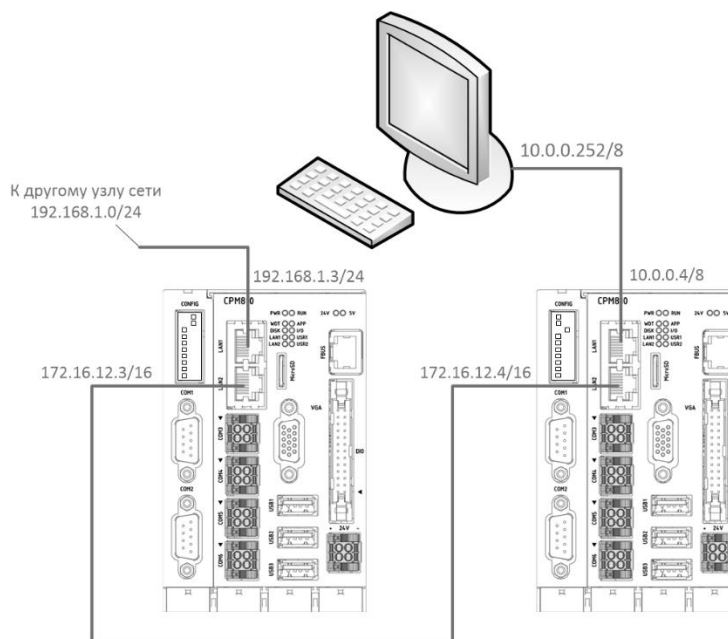


Рисунок 52 – Компьютер и два контроллера в разных подсетях с включенной параллельной маршрутизацией IDE МЭК 61131-3

В данном случае сетевые интерфейсы контроллеров настроены на работу в разных подсетях, и на ПК доступен по сети только контроллер с IP-адресом 10.0.0.4/8, непосредственно входящий в подсеть ПК.

Для того, чтобы сервис IDE Gateway имел возможность обнаружения и взаимодействия со всеми контроллерами, находящимися в разных подсетях, необходимо на каждом контроллере, связывающем две смежные подсети, включить опцию **Разрешить параллельную маршрутизацию** на странице **Маршрутизация CODESYS** веб-интерфейса конфигурирования и применить конфигурацию.

Например, если при использовании топологии, показанной на рисунке 52, требуется доступ к обоим контроллерам из среды разработки, необходимо включить опцию **Разрешить параллельную маршрутизацию** на контроллере с IP-адресом 10.0.0.4/8. В таком случае окно **Выбор устройства**, отображаемое на экране ПК при сканировании сети, будет содержать информацию о двух доступных устройствах, как показано на рисунке .

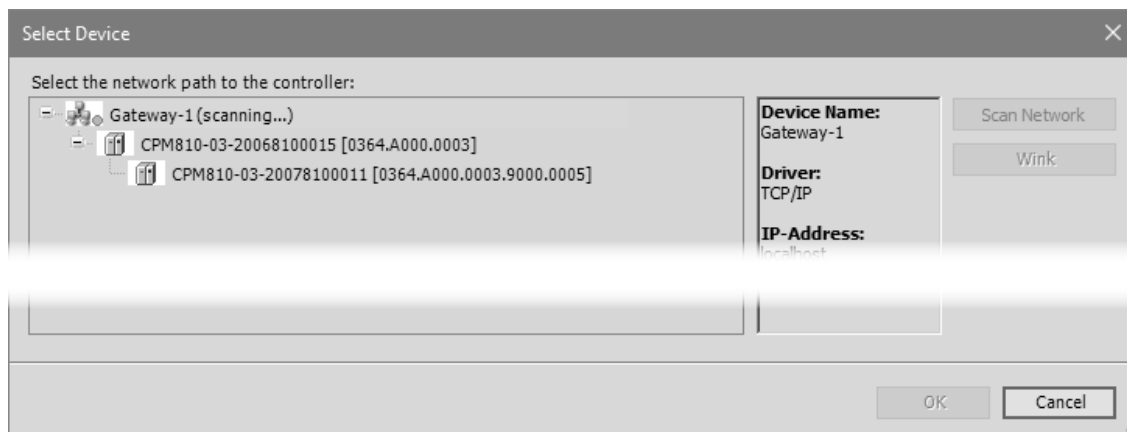


Рисунок 53 – Результат поиска в двух подсетях IDE Gateway

### 3.6. Настройка параметров среды разработки

#### 3.6.1. Общие сведения

Настоящий подраздел содержит краткие сведения о настройке некоторых параметров IDE МЭК 61131-3, влияющих на качество и продуктивность повседневной работы пользователя. Более подробная информация приведена в подразделе **Руководство по интерфейсу пользователя – Опции** справочной системы IDE МЭК 61131-3.

Настройка параметров IDE МЭК 61131-3 выполняется в окне **Опции**, которое отображается на экране ПК по команде **Инструменты – Опции** главного меню IDE МЭК 61131-3. Данные параметры распространяются на все сеансы работы с IDE МЭК 61131-3 на данном ПК со всеми открываемыми проектами.

#### 3.6.2. Изменение языка пользовательского интерфейса

В ряде случаев может потребоваться изменить язык пользовательского интерфейса IDE МЭК 61131-3 на английский и обратно на русский.

При инсталляции IDE МЭК 61131-3 на ПК для элементов пользовательского интерфейса и справочной системы автоматически устанавливается язык, выбранный в качестве основного для операционной системы ПК.

Для выбора другого языка интерфейса пользователя и/или интерактивной справочной системы IDE МЭК 61131-3 следует:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Международные установки (International Settings)**.
3. Установить переключатель **Язык интерфейса (User Interface Language)** в положение **Другой язык (Specific language)** и в выпадающем списке справа выбрать требуемый язык для элементов пользовательского интерфейса среды разработки.
4. Если требуется оставить без изменений или изменить язык справочной системы, следует установить переключатель **Язык онлайн-справки (Help Language)** в положение **Другой язык (Specific language)** и в выпадающем списке справа выбрать требуемый язык для отображения текста интерактивной справочной системы, либо оставить переключатель в положении **Тот же, что и для интерфейса (Same as user interface language)**.
5. Закрыть окно **Опции (Options)** нажатием **ОК**.

6. Завершить работу IDE МЭК 61131-3, после чего запустить ее повторно. Язык пользовательского интерфейса IDE МЭК 61131-3 будет изменен.

### 3.6.3. Интеллектуальный ввод

IDE МЭК 61131-3 содержит мощные средства повышения продуктивности разработки приложений, включая интеллектуальный ввод исходного текста, при котором среда разработки в процессе ввода предлагает различные варианты продолжения вводимой строки во всплывающих контекстных подсказках и при нажатии некоторых сочетаний клавиш (Ctrl+<пробел>, Tab и др.) самостоятельно дополняет вводимую строку недостающими символами имени переменной, члена структуры, программной единицы и других элементов синтаксиса используемого текстового языка программирования.

При разработке приложений с очень большим количеством переменных и программных единиц использование данной функции может замедлить реакцию среды разработки на нажатие клавиш, если операционная система Windows установлена на не очень производительный ПК или функционирует в виртуальной среде.

Если при вводе текста программы с большим количеством переменных наблюдается замедление появления символов в окне текущего открытого редактора IDE МЭК 61131-3, то для временного или постоянного отключения функций интеллектуального ввода следует:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Интеллектуальный ввод (SmartCoding)**.
3. Снять флажки следующих опций:  
**Показывать все экземпляры переменных в ассистенте ввода (Show all instance variables in input assistant),**  
**Показывать компоненты после ввода точки (List components after typing a dot (.)),**  
или, как минимум,  
**Показывать компоненты во время ввода (List components immediately when typing).**
4. Снять флажок опции **Подчеркивать ошибки в редакторе (Underline Errors in the Editor)**, чтобы в процессе ввода текста не выполнялась автоматическая проверка синтаксиса и семантики вводимого текста программы.
5. Закрыть окно **Опции (Options)** нажатием **ОК**.

### 3.6.4. Мониторинг переменных

IDE МЭК 61131-3 при подключении к контроллеру командой **Онлайн (Online) – Логин (Login)** обеспечивает возможность просмотра (мониторинга) текущих значений переменных непосредственно в области ввода и представления текста программного кода, как показано на рисунке 54.

При большом количестве переменных в приложении, выполняющемся на контроллере, скорость обновления значений в области ввода и представления текста программы может снизиться, особенно при подключении к контроллеру через сервисный USB-порт либо при использовании параллельной маршрутизации CODESYS V3.

Имеется возможность мониторинга значений переменных только в области декларации переменных программных единиц приложения, что позволяет улучшить скорость обновления значений большого количества переменных.

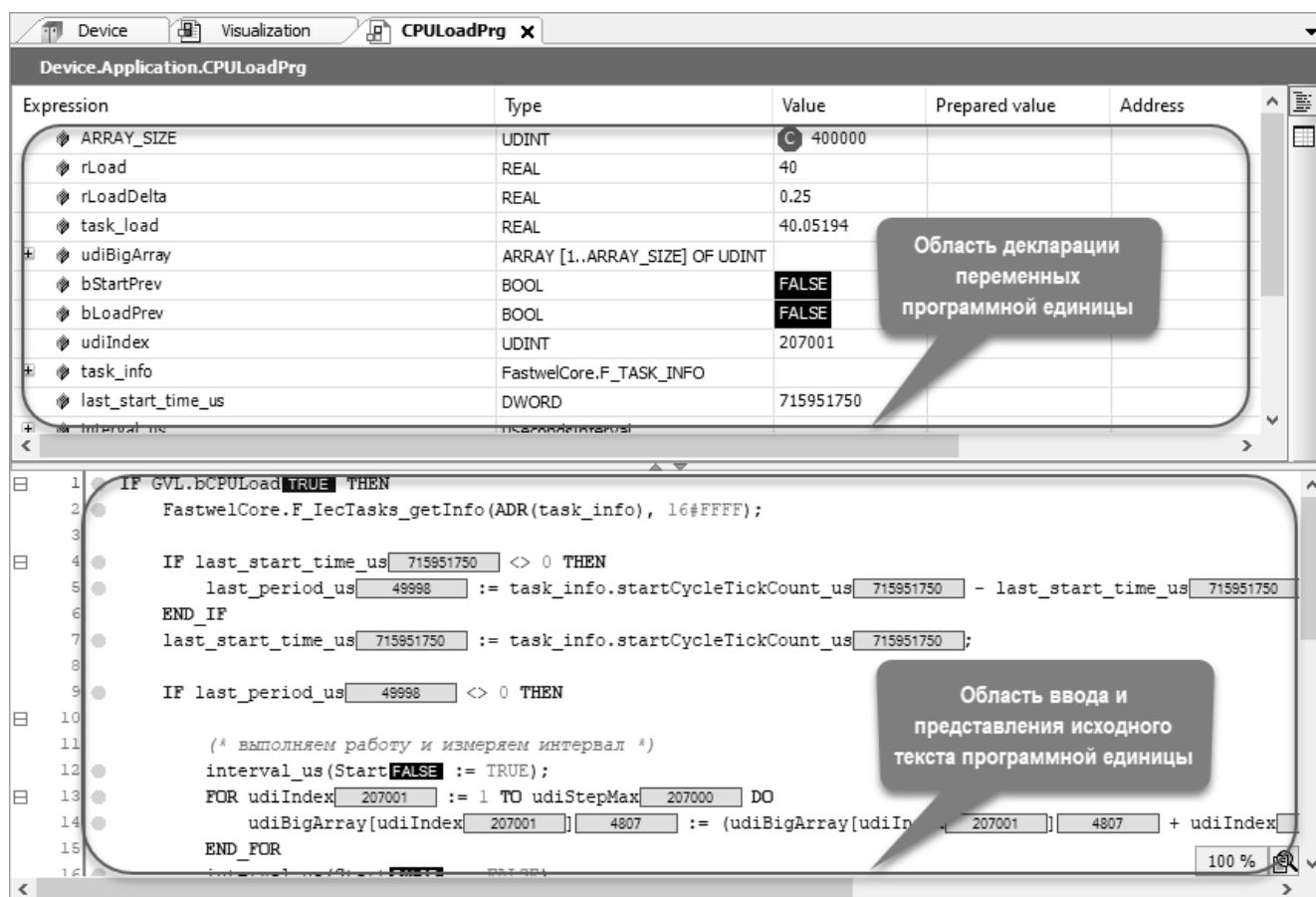


Рисунок 54 – Мониторинг переменных в тексте программы

Для временного или постоянного отключения мониторинга значений переменных в области ввода и представления текста программной единицы:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Текстовый редактор (Text editor)**.
3. Выбрать вкладку **Мониторинг (Monitoring)** и снять флажок **Использовать встроенный мониторинг (Enable inline monitoring)**, как показано на рисунке 55.
4. Закрыть окно **Опции (Options)** нажатием **ОК**. Мониторинг значений переменных будет выполняться только в области декларации переменных.

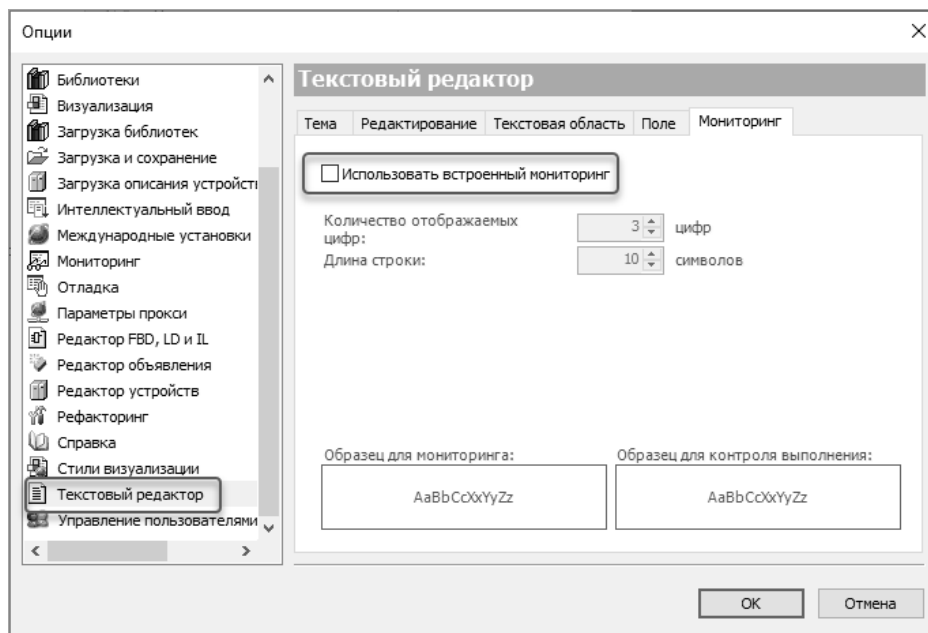


Рисунок 55 – Отключение мониторинга значений переменных в области представления текста программы

### 3.6.5. Управление содержимым главного окна среды разработки при запуске

По умолчанию при запуске IDE МЭК 61131-3 в правой области ее главного окна отображается начальная страница с лентой новостей, списком основных операций и списком ранее открытых проектов.

Для изменения содержимого главного окна среды разработки при запуске:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Загрузка и сохранение (Load and Save)**.
3. В выпадающем списке **При запуске: (At startup)** выбрать одну из более приемлемых для дальнейшего использования опций:  
**Загружать последний проект (Load last loaded project)** – при последующем запуске будет предпринята попытка открыть проект, который был открыт в предыдущем сеансе работы со средой разработки;  
**Ничего не делать (Show empty environment)** – при последующем запуске главное окно среды разработки будет пустым.
4. Закрыть окно **Опции (Options)** нажатием **ОК**.

### 3.6.6. Автоматическое и резервное сохранение проекта во время работы

По умолчанию в процессе работы на проекте IDE МЭК 61131-3 не сохраняет внесенные в проект изменения до тех пор, пока не выполнена команда **Файл (File) – Сохранить проект (Save Project)** или не нажато сочетание клавиш **Ctrl+S**.

О наличии внесенных изменений свидетельствует символ \* справа от имени файла проекта в панели заголовка главного окна среды разработки, как показано на рисунке 56.

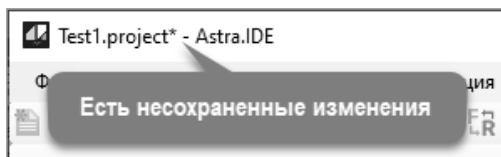


Рисунок 56 – Признак наличия изменений с момента последнего сохранения

Для обеспечения сохранности результатов работы над проектом:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Загрузка и сохранение (Load and Save)**.
3. Для включения периодического автоматического сохранения проекта отметить флажок **Автосохранение с интервалом (Automatically save every)** и установить требуемый период автоматического сохранения в минутах в соответствующем поле справа от флажка.
4. Для включения автоматического сохранения проекта перед каждой компиляцией отметить флажок **Сохранить перед компиляцией (Save before build)**.
5. Для включения функции автоматического создания резервной копии файла проекта отметить флажок **Резервное копирование (Create backup files)**. Впоследствии в каталоге размещения файла проекта будет создаваться резервная копия файла проекта с тем же именем и расширением *backup*.
6. Закрыть окно **Опции (Options)** нажатием **ОК**.

### 3.6.7. Размещение вложенных вкладок редактора устройств

Окно редактора устройств IDE МЭК 61131-3 по умолчанию имеет вертикальное размещение вложенных вкладок, как показано на рисунке 57.

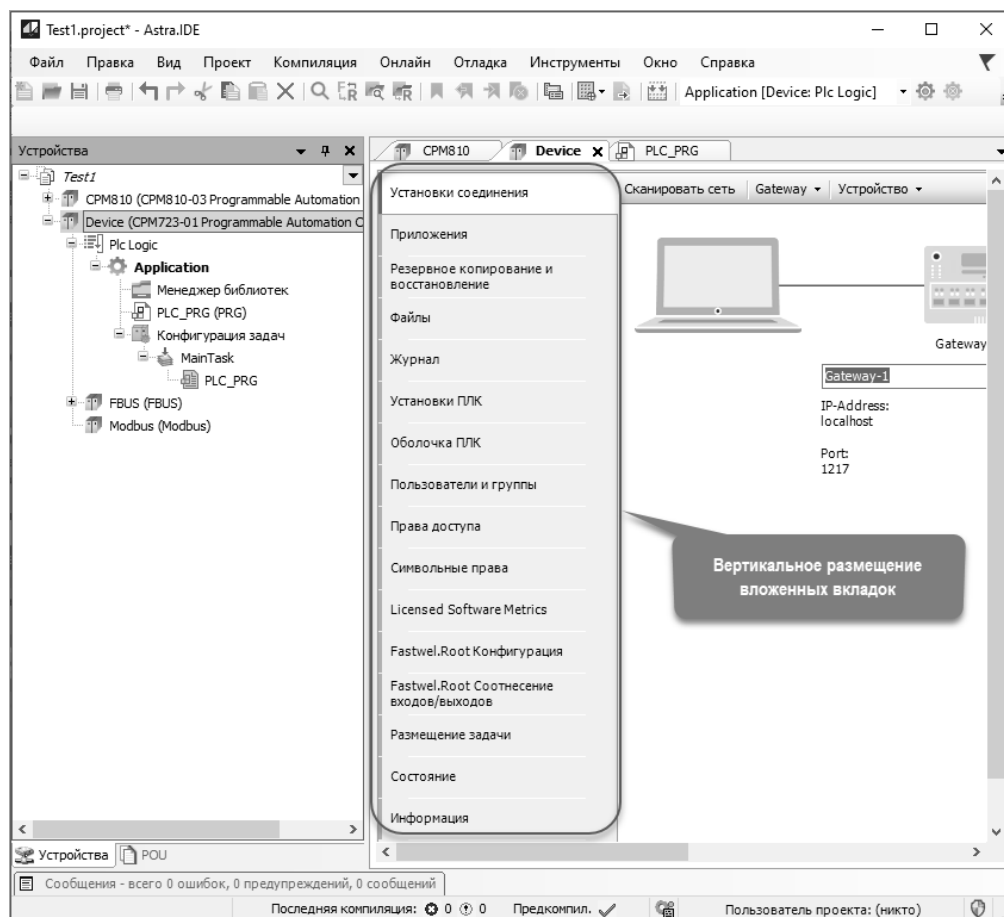


Рисунок 57 – Вертикальное размещение вложенных вкладок редактора устройств

Если данный стиль размещения вложенных вкладок доставляет неудобства в процессе работы, возможно выбрать горизонтальное размещение вкладок, как это было сделано в IDE МЭК 61131-3 более ранних выпусков:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню CODESYS V3.
2. В окне **Опции (Options)** выбрать категорию параметров **Редактор устройств (Device Editor)**.
3. Отметить флажок **Исп. горизонтальные вкладки (Use horizontal tab pages)**.
4. Закрыть окно **Опции (Options)** нажатием **ОК**. Вложенные вкладки редактора устройств будут расположены горизонтально, как показано на рисунке 58.



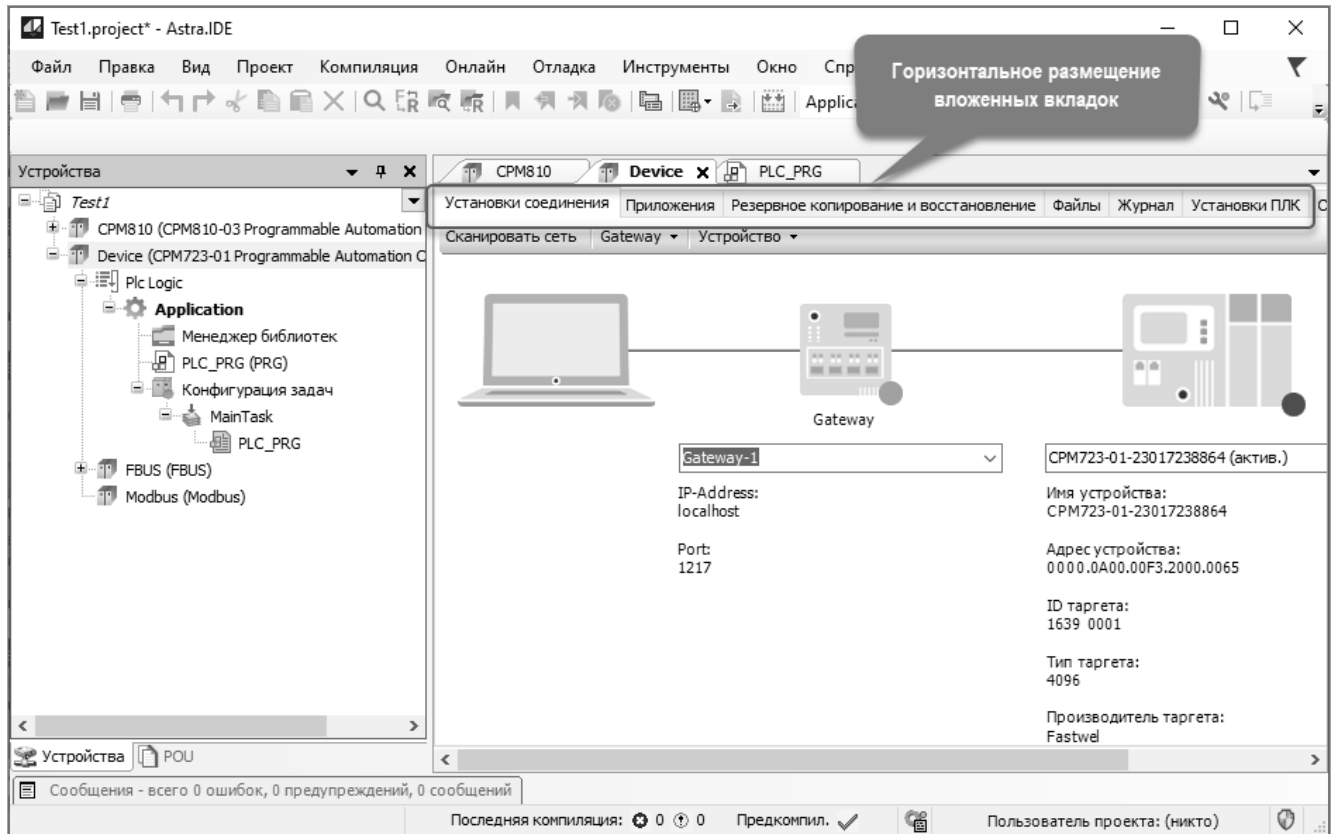


Рисунок 58 – Горизонтальное размещение вложенных вкладок редактора устройства

### 3.6.8. Управление отображением координатной сетки в формах визуализации

Даже если контроллер не содержит подсистемы целевой визуализации, проект IDE МЭК 61131-3 может содержать формы визуализации для параметризации алгоритмов во время пуско-наладочных работ и решения других задач, требующих представления в удобной для восприятия форме значений переменных приложения в среде разработки, чтение и запись которых выполняет сервис мониторинга IDE МЭК 61131-3.

По умолчанию в формах визуализации IDE МЭК 61131-3 выключено отображение координатной сетки, которая в ряде случаев облегчает редактирование и позиционирование графических элементов. Для включения отображения координатной сетки в формах визуализации:

1. Выполнить команду **Инструменты (Tools) – Опции (Options)** в главном меню.
2. В окне **Опции (Options)** выбрать категорию параметров **Визуализация (Visualization)**.
3. Выбрать вкладку **Сетка (Grid)** и установить флажок **Сетка:Видимая (Grid:Visible)**.  
При необходимости имеется возможность изменить шаг координатной сетки путем установки значения в поле **Размер (Size)**.
4. При необходимости привязки графических элементов экранных форм к координатной сетке следует установить флажок **Сетка:Активная (Grid:Active)**.
5. Закрыть окно **Опции (Options)** нажатием **ОК**.

### 3.7. Настройка параметров проекта

#### 3.7.1. Общие сведения

Настоящий подраздел содержит сведения о некоторых параметрах, определяемых пользователем для текущего проекта, открытого в IDE МЭК 61131-3.

Настройка параметров проекта выполняется в окне **Установки проекта (Project Settings)**, которое отображается на экране ПК по команде **Проект (Project) – Установки проекта (Project Settings)** главного меню IDE МЭК 61131-3.

Более подробная информация обо всех параметрах проекта приведена в подразделе **Руководство по интерфейсу пользователя – Диалог 'Установки проекта'** справочной системы IDE МЭК 61131-3.

#### 3.7.2. Параметры компиляции

Если проект открывается в IDE МЭК 61131-3 более новой версии, чем версия, в которой был изначально создан проект, то при открытии проекта на экране ПК может быть выведено окно **Среда проекта (Project Environment)** с предложением обновить до наиболее актуальных версии компилятора, библиотек и других аспектов проектной информации, как показано на рисунке 59.

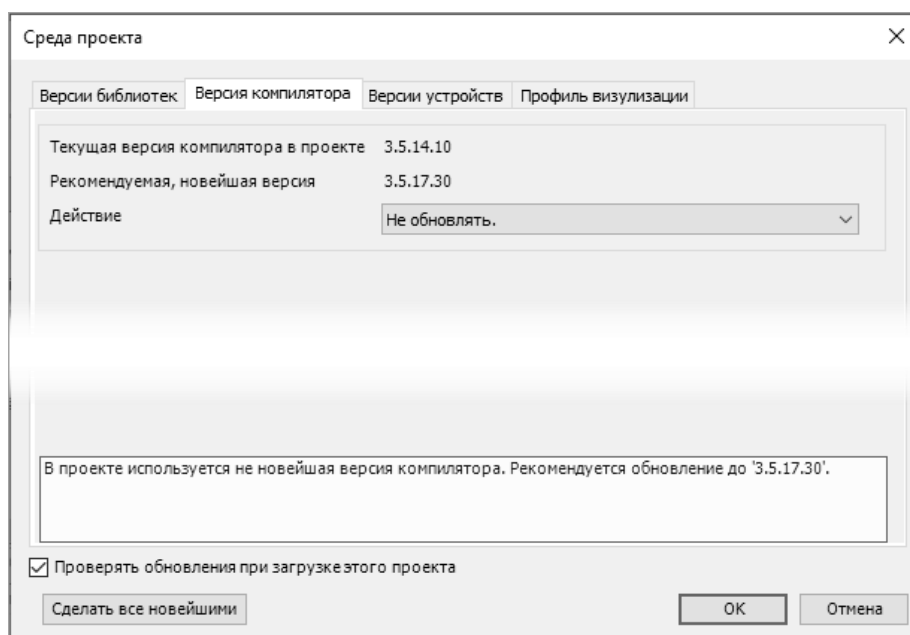




Рисунок 59 – Предложение об обновлении версий проектной информации

Обновление версии компилятора до новейшей приводит к тому, что построение (компиляция) проекта будет в дальнейшем выполняться более новой версией компилятора, установленной вместе с текущей версией среды разработки.



	<p>Если проект был создан и соответствующее приложение загружено в контроллер в IDE МЭК 61131-3 более ранней версии, чем используется в настоящий момент для открытия проекта, то обновление любых аспектов проектной информации приведет к изменению проекта и невозможности соединения с контроллером без требования загрузки измененного приложения, как минимум, методом онлайн-изменения (<b>Логин с онлайн-заменой</b>).</p> <p>В Astra.IDE или CODESYS версии от 3.5.17.0 и выше при открытии проекта, созданного в IDE МЭК 61131-3 более ранней версии, и отказе от обновлений проектной информации потребуется загрузка обновленного приложения, как минимум, методом онлайн-изменения даже в случае успешной компиляции проекта и при наличии файлов с информацией о компиляции (compileinfo) и загрузке (bootinfo), автоматически созданных IDE МЭК 61131-3 более ранней версии после загрузки приложения в контроллер.</p>
---	--

	<p>В Astra.IDE или CODESYS версии от 3.5.17.0 и выше при наличии в проекте элементов визуализации, в том числе не целевой, для успешной компиляции проекта потребуется обновить версии компилятора и профиля визуализации до текущих из состава используемой IDE МЭК 61131-3.</p> <p>Рекомендуется делать обновление профиля визуализации и версии компилятора в окне <b>Установки проекта</b> по команде меню <b>Проект – Установки проекта</b>.</p> <p>Обновление версий в окне <b>Среда проекта</b> по кнопке <b>Сделать все новейшими</b> может привести к невозможности безошибочной генерации кода проекта.</p>
---	---

Если нет необходимости переходить к новым версиям проектной информации, следует снять флажок **Проверять обновления при загрузке проекта (Check for updates when loading this project)** и нажать **ОК**. Впоследствии при открытии проекта окно с предложением обновления отображаться не будет.

При необходимости изменить версию компилятора для построения проекта:

1. Выполнить команду **Проект (Project) – Установки проекта (Project Settings)**.
2. В появившемся окне **Установки проекта (Project Settings)** выбрать категорию параметров **Опции компиляции (Compile options)**.
3. Выбрать желаемую версию компилятора в выпадающем списке **Версия компилятора (Compiler Version) – Конкретная версия (Fix Version)**.
4. Если в проекте имеются формы визуализации, выбрать категорию **Профиль визуализации** и установить профиль визуализации, версия которого совпадает с выбранной версией компилятора или соответствует выбранной версии компилятора.
5. Закрыть окно нажатием кнопки **ОК**.

	<p>Если в IDE МЭК 61131-3 отсутствует установленный профиль визуализации, версия которого соответствует требуемой версии компилятора, рекомендуется установить наиболее новые версии компилятора и профиля визуализации в окне <b>Установки проекта</b>.</p>
	<p>В Astra.IDE или CODESYS версии от 3.5.17.0 обновление версий компилятора и профиля визуализации в окне <b>Среда проекта</b> по кнопке <b>Сделать все новейшими</b> может привести к невозможности безошибочной генерации кода проекта.</p>

### 3.7.3. Параметры загрузки исходного кода проекта в контроллер

Приложение загружается в контроллер из IDE МЭК 61131-3 в бинарном формате, так что впоследствии, в случае утраты исходного текста (файла проекта), не будет возможности сопровождения и развития системы управления, в состав которой входит контроллер.

Для автоматической или полуавтоматической загрузки исходного текста приложения в контроллер вместе с загрузкой бинарной информации:

1. Выполнить команду **Проект (Project) – Установки проекта (Project Settings)**.
2. В появившемся окне **Установки проекта (Project Settings)** выбрать категорию параметров **Загрузка исходного кода (Source Download)**.
3. В выпадающем списке **Целевое устройство** выбрать, требуется ли загружать исходный текст в каждый контроллер, описанный в открытом проекте, или в контроллер конкретного типа, используемого в качестве целевой платформы для текущего приложения в открытом проекте.
4. Выбрать режим записи в группе параметров **Режим записи**, показанной на рисунке 60: **При загрузке программы и онлайн-замене (Implicitly at program download and online change)** – исходный текст будет загружаться в контроллер при любой загрузке приложения;

**При создании загрузочного проекта (Implicitly at creating bootproject)** – исходный текст будет загружаться только при создании загрузочного проекта по команде **Онлайн (Online)** – **Создать загрузочное приложение (Create boot application)**;

**При создании загрузочного проекта, загрузке и онлайн-замене (Implicitly at creating bootproject, download and online change)** – исходный текст будет загружаться при любом способе загрузки приложения в контроллер, описанном выше;

**Уточнять при загрузке программы и онлайн-изменении (Prompt at program download and online change)** – при загрузке измененного приложения на экране будет отображаться диалоговая панель с предложением загрузить исходный код в контроллер;

**Только по требованию (Only on demand)** – режим выбран по умолчанию и означает, что загрузка исходного кода в контроллер должна быть выполнена вручную командой меню **Онлайн (Online)** – **Загрузка исходного кода в подсоединенное устройство (Source download to connected device)** или командой **Файл (File)** – **Загрузка исходного кода (Source download)**.

5. Закрывать окно **Установки проекта** нажатием кнопки **ОК**.

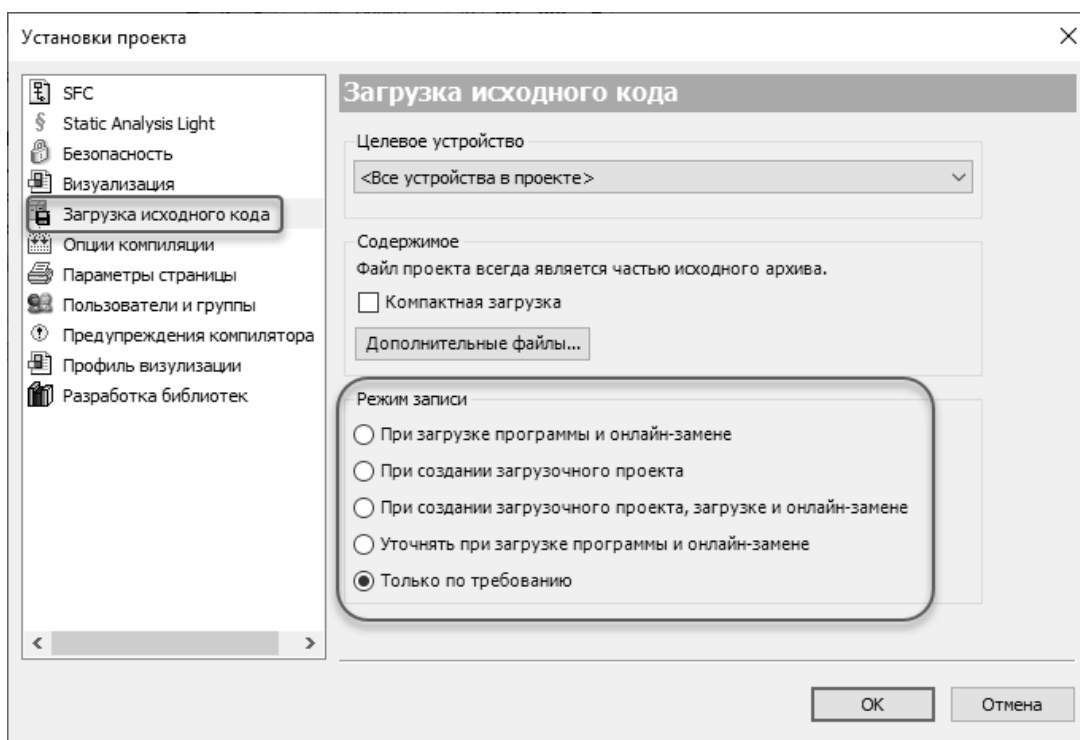



Рисунок 60 – Параметры загрузки исходного кода проекта в контроллер

Впоследствии проект может быть выгружен из контроллера командой меню **Файл (File)** – **Выгрузка исходного кода (Source upload)**.

#### 3.7.4. Защита файла проекта от несанкционированного доступа

При необходимости предотвращения несанкционированного доступа к исходному коду проекта имеется возможность защиты файла проекта паролем.

	<p>В случае утраты установленного пароля нет никакой возможности открыть проект, в том числе специалистами компании-изготовителя IDE МЭК 61131-3.</p>
---	---

Для защиты файла проекта паролем:

1. Выполнить команду **Проект (Project)** – **Установки проекта (Project Settings)**.
2. В появившемся окне **Установки проекта (Project Settings)** выберите категорию параметров **Безопасность (Security)**.
3. Установить опцию **Шифрование**, а затем **Пароль**, как показано на рисунке 61.

4. Ввести требуемый пароль в поля **Новый пароль (New password)** и **Подтверждение пароля (Confirm new password)** и закрыть окно нажатием кнопки **ОК**.
5. Сохранить проект командой **Файл (File) – Сохранить (Save)**. При последующих попытках открыть проект на экран ПК будет выводиться диалоговая панель **Пароль (Encryption Password)** с требованием ввести пароль.

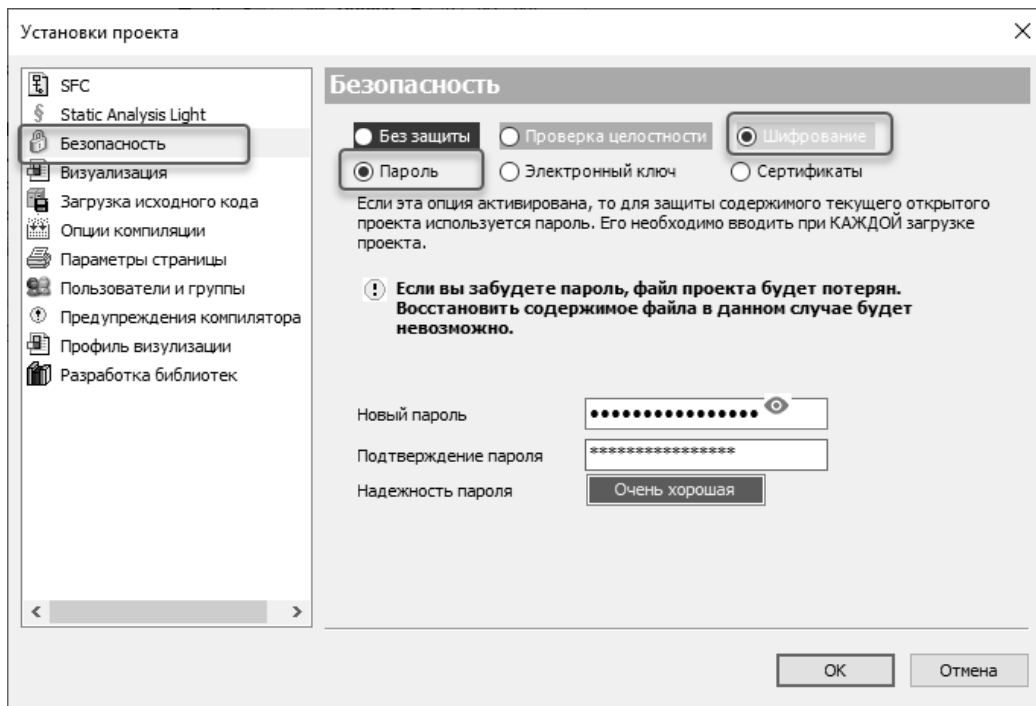


Рисунок 61 – Установка пароля на файл проекта

Для снятия защиты файла проекта ранее установленным паролем:

1. Выполнить команду **Проект (Project) – Установки проекта (Project Settings)**.
2. В появившемся окне **Установки проекта (Project Settings)** выбрать категорию параметров **Безопасность (Security)**.
3. Выбрать опцию **Проверка целостности** и закрыть окно нажатием кнопки **ОК**.
4. Сохранить проект командой **Файл (File) – Сохранить (Save)**. При последующем открытии проекта не будет запроса ввести пароль.

## 4. Разработка приложений для контроллеров Fastwel в IDE МЭК 61131-3

### 4.1. Общие сведения

Настоящий раздел содержит информацию об использовании IDE МЭК 61131-3 для разработки прикладного программного обеспечения (*приложения*) для контроллеров Fastwel на языках стандарта ГОСТ Р МЭК 61131-3 и о принципах функционирования системы исполнения приложений контроллеров Fastwel.

Специфическая информация о программировании и настройке системных параметров контроллеров Fastwel, периферийных модулей и сетевых протоколов, поддерживаемых контроллерами, приведена в документах, перечисленных в таблице 1 п 1.3.

### 4.2. Программная модель приложения

#### 4.2.1. Общие сведения

Программная модель большинства языков программирования состоит из понятий, посредством которых представляются элементы данных, операции над данными, прикладная программа, как средство реализации пользовательского алгоритма, а также окружение прикладной программы, в котором она запускается и с которым взаимодействует.

Основные элементы программной модели IDE МЭК 61131-3, предназначенные для реализации исполняемого кода прикладных программ для контроллеров, определены стандартом ГОСТ Р МЭК 61131-3 (IEC 61131-3).

Настоящий подраздел содержит информацию о программной модели приложения IDE МЭК 61131-3 с учетом особенностей реализации системы исполнения приложений в контроллерах Fastwel.

#### 4.2.2. Проект IDE МЭК 61131-3

*Проектом* далее называется сохраняемая в одном файле с расширением .project (файле проекта) или .projectarchive (архивном файле проекта) именованная совокупность описаний элементов программной модели IDE МЭК 61131-3, используемых для представления прикладного алгоритма, который должен выполняться на одном или нескольких контроллерах, конфигурации аппаратных средств и средств взаимодействия одного или нескольких контроллеров по сети друг с другом и другими узлами сети.

Минимальный работоспособный проект IDE МЭК 61131-3 ППО для контроллеров Fastwel состоит из следующих элементов:

1. Целевое устройство – описание устройства типа *CPMXXX-ZZ Programmable Automation Controller* (где *XXX* – числовой суффикс модели, а *ZZ* – числовой суффикс исполнения), определяющее тип целевого процессора для компилятора IDE МЭК 61131-3, типы, количество и размеры сегментов памяти приложения и набор типов данных, библиотек и других функциональных возможностей IDE МЭК 61131-3, поддерживаемых системой исполнения контроллера.  
Целевое устройство в проекте IDE МЭК 61131-3 представляет контроллер или модуль центрального процессора контроллера как вычислительное устройство для загрузки конфигурации аппаратных средств, сетевых сервисов и пользовательского приложения из IDE МЭК 61131-3.
2. Приложение – именованный элемент проектной информации, принадлежащий элементу *<Имя\_устройства> (CPMXXX-ZZ Programmable Automation Controller) – Plc Logic*, показанному на рисунке 62. Приложение представляет собой логически связанную конфигурацию исполняемых программных единиц и других элементов программной модели ГОСТ Р МЭК 61131-3. Одно целевое устройство Fastwel может содержать не более одного приложения.
3. Одну программную единицу (Program Organization Unit или *POU*) типа *PROGRAM* по ГОСТ Р МЭК 61131-3.

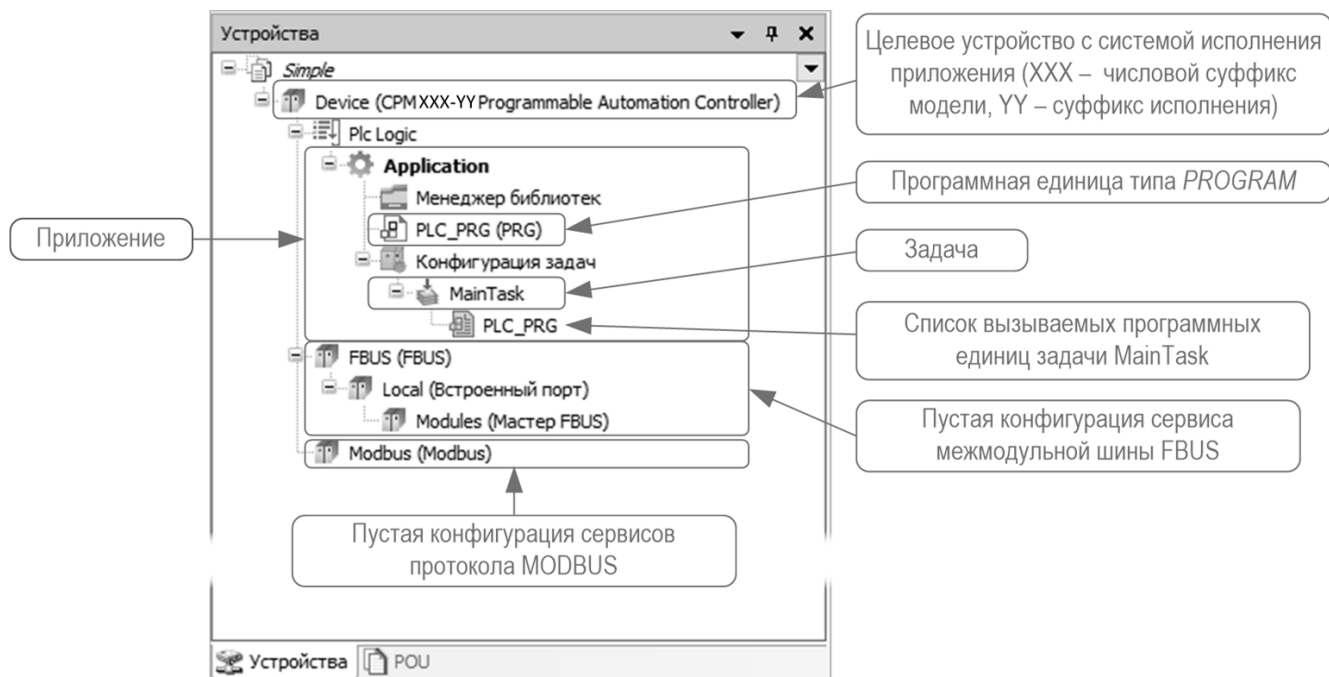



Рисунок 62 – Элементы минимального проекта для контроллера Fastwel

4. Одну циклическую задачу, в список вызываемых программных единиц (список вызовов) которой включена хотя бы одна POU типа *PROGRAM*.

Конфигурацией приложения (контроллера) далее будет называться совокупность целевого устройства и всех принадлежащих ему элементов проектной информации в дереве проекта.

Конфигурация приложения в минимальном работоспособном проекте, создаваемом по команде **Файл – Новый проект** в IDE МЭК 61131-3 с Fastwel PLC Application Toolkit по шаблону *Стандартный проект* для одного контроллера CPM723-01, показана на рисунке 62.

	<p>Проект IDE МЭК 61131-3 может содержать несколько конфигураций приложения для нескольких целевых устройств, в том числе разных типов.</p>
---	---

#### 4.2.3. Программы, функциональные блоки и функции

Программа *PROGRAM* в IDE МЭК 61131-3 является основной программной единицей (POU) для реализации пользовательского алгоритма и представлена следующими основными атрибутами и свойствами:

- уникальным именем, по которому ее можно вызывать из других POU или включить в список вызовов одной из задач;
- интерфейсом доступа по данным в виде наборов входных и выходных переменных, а также свойств;
- интерфейсом доступа по исполнению в виде действий, методов и переходов;
- внутренними переменными, определяющими состояние программной единицы между вызовами;
- временными переменными (VAR\_TEMP), время жизни которых ограничено текущим вызовом POU, действия или метода POU.

В приложении может существовать только один экземпляр программы с некоторым именем.


Программные единицы типа *PROGRAM* могут вызывать другие программные единицы типа *PROGRAM*, а также экземпляры программных единиц типа *FUNCTION BLOCK* (функциональный блок) и *FUNCTION* (функция).

Отличие между программой и функциональным блоком состоит в том, что в приложении может быть только один экземпляр программы с некоторым именем, а экземпляры функционального блока



можно объявлять в виде переменных среди внутренних переменных программ и других функциональных блоков и вызывать их из кода других программных единиц.

Функция (*FUNCTION*) очень похожа на программу (*PROGRAM*), но не имеет внутренних переменных, значения которых сохраняются между вызовами. Т.е. время "жизни" функции ограничивается временем между началом ее выполнения в точке вызова и возвратом на следующую инструкцию после точки вызова, тогда как время "жизни" программ и функциональных блоков ограничено временем функционирования приложения.

	<p>Программные единицы (POU) и пользовательские типы данных (DUT) могут создаваться как на уровне одного целевого устройства, входящего в проект, так и глобально для всех целевых устройств, имеющихся в проекте.</p> <p>В первом случае программные единицы и типы данных создаются на вкладке <b>Устройства</b>, расположенной в левой области главного окна IDE МЭК 61131-3, в качестве дочерних элементов для элемента дерева проекта, представляющего приложение, как показано на рисунке 63. При таком способе создания программных единиц и типов данных они будут доступны только в приложении для выбранного целевого устройства.</p> <p>Во втором случае программные единицы (POU) и пользовательские типы данных создаются на вкладке <b>POU</b>, расположенной в левой области главного окна IDE МЭК 61131-3, как показано на рисунке 64.</p>
---	--

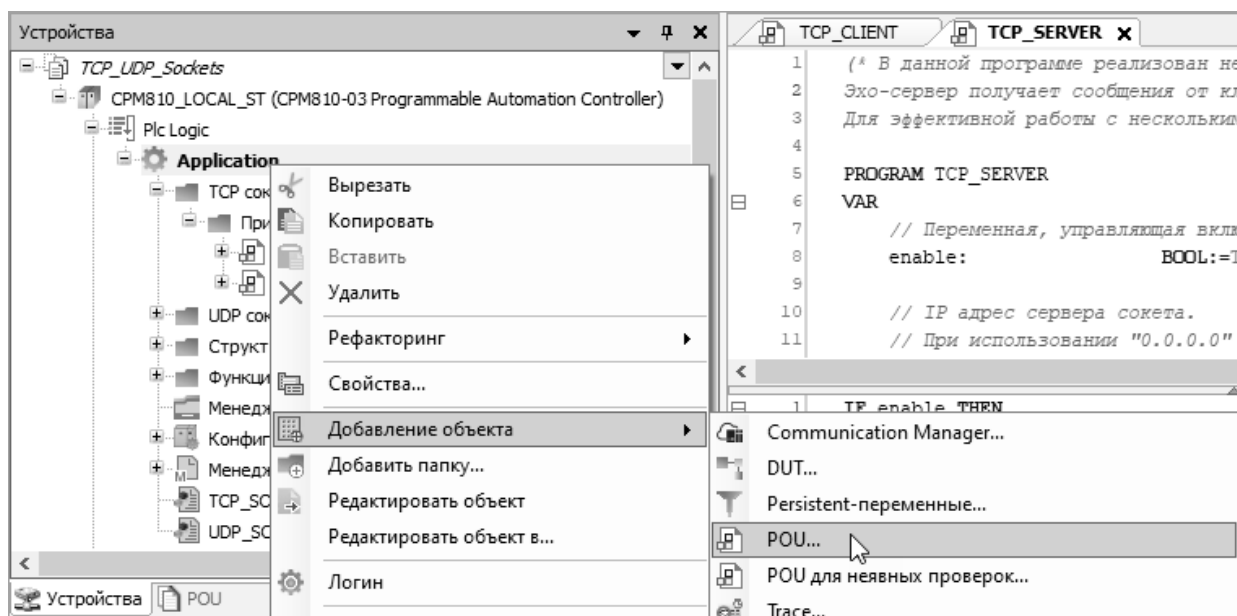


Рисунок 63 – Создание программной единицы на уровне приложения для целевого устройства

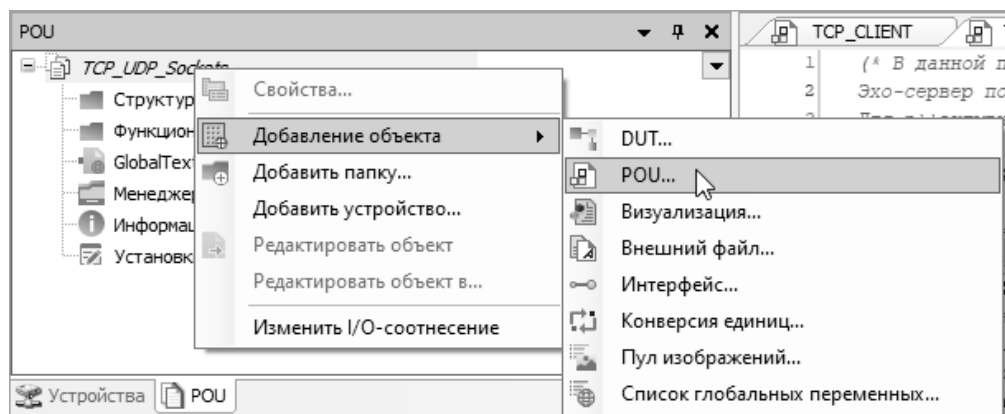


Рисунок 64 – Создание программной единицы на уровне проекта



#### 4.2.4. Задачи

Понятие *задача* (Task) олицетворяет процессор, способный выполнять запущенную на нем последовательность программных единиц типа *PROGRAM* при наступлении заданного условия запуска. Таким образом, задача является элементом управления программами, определяя для них контекст выполнения (как правило, указатель текущей выполняемой инструкции, стек, набор значений регистров процессора, состояние математического сопроцессора и другую системную информацию).

Очередной вызов всех программ в списке вызовов некоторой задачи называется *циклом* данной задачи, при этом непосредственно перед циклом задача вводит данные окружения, на которые ссылаются программные единицы данной задачи, а по завершении цикла – выводит данные в окружение. Более подробная информация о механизмах обмена данными с окружением приведена в п. 4.3.

Система исполнения приложений контроллеров Fastwel поддерживает циклические, ациклические и свободно-исполняемые задачи.

Последовательность программ, выполняемых в контексте *циклической* задачи, вызывается с периодом, заданным для данной задачи, а для последовательности программ, выполняемых в контексте ациклической задачи, условием запуска является передний фронт булевой переменной-события (для задач типа *Событие*) или удержание в состоянии TRUE (для задач типа *Статус*) некоторой булевой переменной, представляющей контролируемый статус.

*Свободно-исполняемые* (Free Running) задачи не имеют периода выполнения, т.е. их новый цикл запускается почти сразу после завершения выполнения предыдущего цикла.

Более подробное описание типов задач приведено в п. 4.4.

В проекте, показанном на рисунке 62, задача *MainTask* в списке вызовов содержит единственную программу *PLC\_PRG*, которая будет циклически вызываться после загрузки приложения в контроллер.

Основным общим свойством задач всех типов является *приоритет*, отражающий срочность или степень важности программных единиц, выполняемых в контексте задачи, по сравнению с программными единицами, выполняемыми в контексте других задач. Приоритеты задач определяют формальный критерий, согласно которому система исполнения контроллера предоставляет физический процессор той или иной задаче: если в некоторый момент времени должны выполняться или уже запущены несколько задач, то процессор получает задача с наивысшим приоритетом. Процесс определения задачи, которой в некоторый момент времени должен быть отдан процессор, называется планированием задач (tasks scheduling), а компонент системного программного обеспечения, занимающийся планированием задач, называется планировщиком (scheduler).

В системе исполнения приложений контроллеров Fastwel задачи отображаются непосредственно на потоки исполнения (threads) операционной системы, и каждой задаче может быть задан приоритет от 0 до 31, при этом наивысшему приоритету для планировщика среди всех задач приложения соответствует наименьшее абсолютное значение приоритета, равное 0.

В системе исполнения приложений МЭК 61131-3 контроллеров Fastwel имеется несколько классов приоритета задач:

1. Задачи с приоритетом от 0 до 7 являются задачами жесткого реального времени, которые вытесняют все задачи приложения с большим абсолютным значением приоритета, а также все системные потоки операционной системы, за исключением потока сервиса ввода-вывода, обслуживающего обмен данными по локальной межмодульной шине FBUS.  
Задача с приоритетом 0 имеет одинаковый приоритет с потоком сервиса ввода-вывода локальной шины FBUS.  
Если в некоторый момент времени должно начаться выполнение более одной задачи с одинаковым приоритетом в пределах 0 до 7, то выполняться будет задача, ожидавшая освобождения процессора дольше остальных задач одинакового с ней приоритета.  
Задача с приоритетом от 0 до 7 не освобождает процессор до завершения очередного цикла или до готовности более приоритетной задачи.
2. Задачи с приоритетом от 8 до 15 являются задачами реального времени, которые вытесняют все задачи приложения с большим абсолютным значением приоритета, а также все системные потоки операционной системы, за исключением потоков сервиса



ввода-вывода, обслуживающих обмен данными по локальной и удаленным шинам FBUS, а также сервисов протокола MODBUS RTU/ASCII.

Если в некоторый момент времени должно начаться выполнение более одной задачи с одинаковым приоритетом в пределах 8 до 15, то выполняться будет задача, ожидавшая освобождения процессора дольше остальных задач одинакового с ней приоритета.


Задача с приоритетом от 8 до 15 не освобождает процессор до завершения очередного цикла или до готовности более приоритетной задачи.

3. Задачи с приоритетом от 16 до 31 фактически имеют одинаковый уровень приоритета для планировщика и всегда вытесняются задачами реального времени и системными потоками сервиса ввода-вывода, обслуживающими обмен по шинам FBUS. Данные задачи планируются с разделением времени между собой, при этом чем дольше некоторая, готовая к исполнению, задача не получает процессор, тем больше вероятность предоставления ей процессора планировщиком.


Таким образом, можно считать, что значения приоритета от 16 до 31 соответствуют значению 16.

	<p>Задачи реального времени разного приоритета (от 0 до 15) и задачи, выполняющиеся с разделением времени (с приоритетом от 16 и выше), при обращении из вызываемых в их контексте программных единиц к одним и тем же глобальным переменным по чтению и записи могут нарушить логику работы соответствующих алгоритмов (см. п. 6.4.3.8).</p>
	<p>Задачи реального времени одинакового приоритета (например, 15), обращающиеся к одним и тем же глобальным переменным по чтению и записи, не нарушают целостность данных друг другу, поскольку всегда выполняются до завершения очередного цикла.</p>

В системе исполнения приложений контроллеров Fastwel имеется специальная сервисная задача, функционирующая с периодом, который определяется значением параметра (*CPMXXX-ZZ Programmable Automation Controller*): **Fastwel.Root.Конфигурация : Период выполнения сервисной задачи**. Данная задача, помимо прочего, проверяет условия выполнения ациклических задач типа *Событие* и *Статус* и устанавливает им признак готовности к выполнению. В частности, если для некоторой ациклической задачи типа *Событие* произошел переход переменной-события из состояния FALSE в TRUE, то данная ациклическая задача начнет выполняться не позднее, чем по истечении интервала времени, равного периоду сервисной задачи. Таким образом, ациклические задачи типа *Событие* могут выполняться с периодом, равным двукратному периоду сервисной задачи. Ациклические задачи типа *Статус* способны выполняться с периодом, равным периоду сервисной задачи.

	<p>Сервисная задача вытесняет все циклические и свободно-исполняемые задачи с приоритетами от 8 до 31, системные потоки сервисов сетевых протоколов и сервиса ввода-вывода, обслуживающего обмен по удаленным шинам FBUS.</p>
---	---

В системе исполнения приложений и в среде разработки IDE МЭК 61131-3 для задачи любого типа может быть настроен специальный сторожевой таймер, обеспечивающий контроль "зависания" исполняемого кода задачи. Зависанием циклической задачи считается выполнение ее цикла в течение времени, существенно превышающего заданный для нее период. Ациклические задачи типа *Событие* и *Статус*, а также свободно-исполняемые задачи, не имеют периода выполнения, поэтому для них, как и для циклических задач, разработчик приложения может включить сторожевой таймер (флажок **Сторожевой таймер – Включить** конфигурации задачи) и задать интервал времени, в течение которого должны завершаться вызовы программных единиц задачи (параметр **Сторожевой таймер – Время** конфигурации задачи). При этом параметр **Сторожевой таймер – Восприимчивость** предназначен для установки допустимого количества превышений заданного интервала сторожевого таймера, следующих друг за другом: если для параметра **Восприимчивость** установлено значение 0, 1 или 2, то для цикла задачи допускается однократное превышение интервала сторожевого таймера, а при втором подряд превышении контроллер будет переведен в безопасный режим. Для значений более 2 допустимое количество циклов с превышением на 1 меньше заданного значения (для 3 допускается 2 цикла с превышением, для n – n-1 цикл).

	В системе исполнения приложений контроллеров Fastwel, если сторожевой таймер задачи не включен, то система исполнения включает его автоматически и устанавливает интервал сторожевого таймера 30 секунд и восприимчивость 1.
---	--


#### 4.2.5. Пользовательские типы данных (DUT)

В приложениях IDE МЭК 61131-3 имеется возможность создания следующих пользовательских типов данных, как на уровне приложения для одного целевого устройства, так и на уровне проекта в целом на вкладке **POU**:

1. **STRUCT** – не примитивный тип данных, описывающий набор "полей" (fields), называемых "членами структуры" (struct members), других типов: примитивных, не примитивных типов и/или их массивов, расположенных в памяти друг за другом в порядке следования их имен в декларации типа.

Переменные типа **BOOL** занимают в памяти один байт.

При необходимости иметь члены структуры, представляющие логические переменные типа **BOOL**, следует пользоваться специальным типом **BIT**. Члены структуры типа **BIT**, следующие друг за другом, автоматически упаковываются компилятором в байты.

	<p>Если перед декларацией структуры не объявлен атрибут выравнивания <i>pack_mode</i>, то для целевых устройств на базе микропроцессоров с архитектурой ARM (CPM723, CPM823, CPM833 и т.п.) используется выравнивание на 8 байт, а для целевых устройств на базе микропроцессоров с архитектурой x86 (CPM810-03) – 4 байта.</p> <p>Более подробная информация об атрибуте <i>pack_mode</i> приведена в подразделе <b>Руководство по программированию – Прагмы – Прагмы атрибутов – Атрибут 'pack_mode'</b> интерактивной справочной системы IDE МЭК 61131-3.</p>
---	--

2. Перечисление – тип данных, описывающий именованные наборы значений.
3. Псевдоним – тип данных, эквивалентный другому декларируемому типу данных, но имеющий другое имя.
4. Объединение (**UNION**) – тип данных, предназначенный для обращения к некоторой области памяти с использованием разных типов данных. Например, переменные объединения:

```

TYPE VariantValue :
UNION
    byteValue   : BYTE;
    wordValue   : WORD;
    dwordValue  : DWORD;
    realValue   : REAL;
    lrealValue  : LREAL;
END_UNION
END_TYPE

```

будут занимать в памяти 8 байт (соответствует длине **LREAL**) и могут служить для размещения в них значений всех перечисленных в объединении типов.

Более подробная информация о типах данных приведена в подразделе **Руководство по программированию – Типы данных** интерактивной справочной системы IDE МЭК 61131-3.


#### 4.2.6. Символьная конфигурация

При установке Fastwel PLC Application Toolkit инсталлятор по умолчанию устанавливает на ПК и регистрирует приложение CODESYS OPC Server V3. Данное приложение является OPC-сервером и позволяет организовать обмен данными между приложениями класса SCADA/HMI на ПК и контроллерами с системой исполнения приложений МЭК 61131-3.

Для обеспечения возможности чтения и записи переменных приложения IDE МЭК 61131-3, функционирующего на контроллере, через CODESYS OPC Server V3, необходимо добавить в приложение для контроллера элемент проектной информации *Символьная конфигурация*, после чего выбрать переменные приложения, к которым требуется иметь доступ через OPC-сервер, установить для них тип доступа (чтение, запись, чтение-запись), скомпилировать и загрузить приложение в контроллер.

Затем при помощи приложения OPC Configurator, установленного с CODESYS OPC Server V3, следует создать конфигурацию OPC-сервера, содержащую описания связей OPC-сервера с контроллерами по логическим или IP-адресам через локальный или удаленный сервис IDE МЭК 61131-3.

Элемент проектной информации **Символьная конфигурация** позволяет выбрать переменные приложения, к которым требуется иметь доступ через сервер OPC UA, входящий в состав СПО контроллера. Более подробная информация о конфигурировании сервера OPC UA приведена в п. 4.7.

	<p>Сервер OPC UA перед началом использования должен быть активирован в веб-конфигураторе СПО контроллера.</p>
---	---

Например, для обеспечения приложениям SCADA/HMI возможности обмена данными через OPC UA с контроллером, в котором функционирует приложение, загруженное из учебного проекта tutorial.project из состава Fastwel PLC Application Toolkit, необходимо выполнить следующие действия:

1. В дереве проекта щелкнуть правой кнопкой мыши на элементе (*CPMXXX-ZZ Programmable Controller*) – *Plc Logic – Application* и в контекстном меню выполнить команду **Добавление объекта – Символьная конфигурация**.
2. Не изменяя исходных опций, установленных в диалоговой панели **Добавить Символьная конфигурация**, нажать кнопку **Добавить**, как показано на рисунке 65.

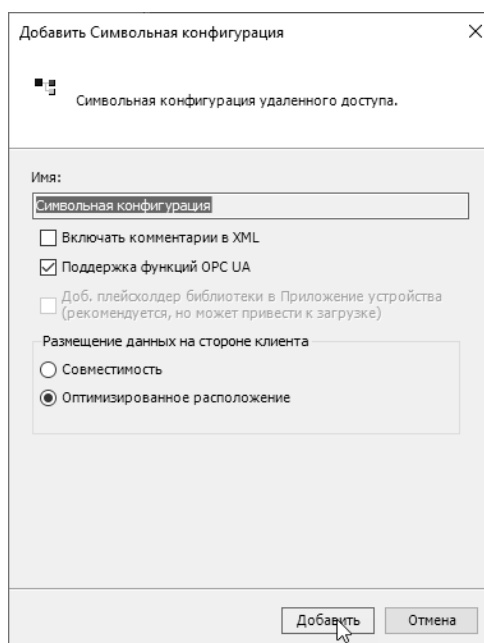


Рисунок 65 – Создание символьной конфигурации в приложении

В дереве проекта появится элемент *Символьная конфигурация*, а в правой части главного окна среды разработки будет открыт редактор символьной конфигурации.

3. Выполнить команду **Компиляция** нажатием кнопки в меню редактора символьной конфигурации, показанной на рисунке 66.

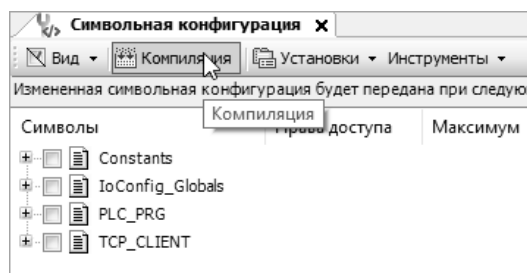




Рисунок 66 – Компиляция для получения информации о переменных приложения

4. На вкладке редактора символьной конфигурации раскрыть переменные программных единиц *PLC\_PRG* и *TCP\_CLIENT* и отметить флажок выбора для переменных *PLC\_PRG.dwCycleCounter*, *TCP\_CLIENT.app\_control\_cmd*,

*TCP\_CLIENT.app\_relay1\_state* и *TCP\_CLIENT.app\_relay2\_state*, как показано на рисунке 67.

5. В столбце **Права доступа** изменить тип доступа к переменным *PLC\_PRG.dwCycleCounter*, *TCP\_CLIENT.app\_relay1\_state* и *TCP\_CLIENT.app\_relay2\_state* на "только чтение", для чего по два раза щелкнуть в соответствующих ячейках столбца **Права доступа** до изменения значков  на значки .
6. Выполнить команду **Компиляция – Генерировать код**, а затем загрузить приложение в контроллер методом полной загрузки.
7. Запустить приложение в контроллере командой **Отладка – Старт** или клавишей F5.

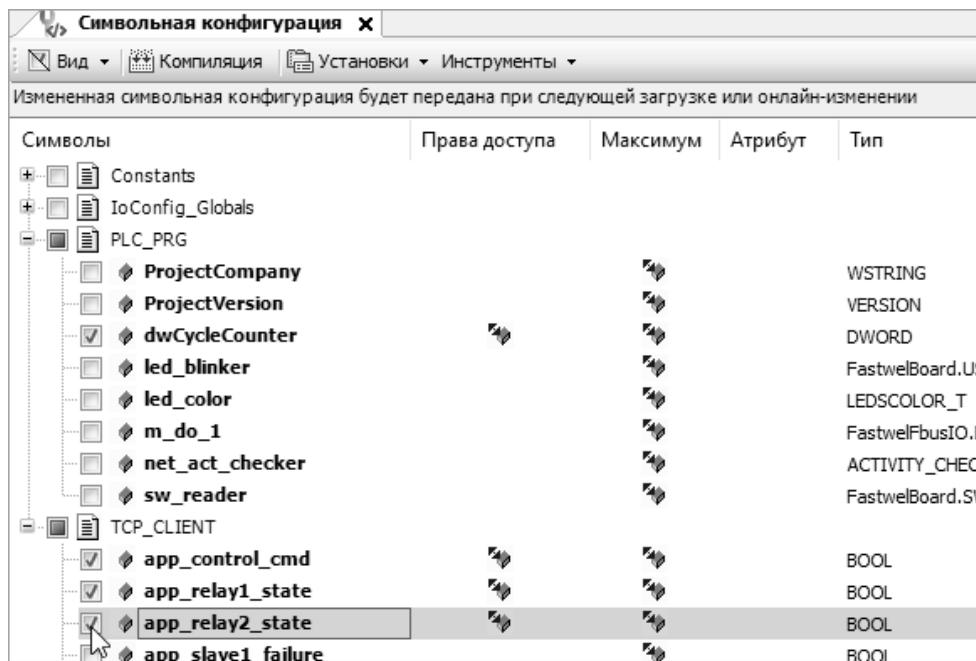


Рисунок 67 – Выбор переменных для формирования символической информации

8. Запустить веб-браузер, установить соединение с контроллером, перейти на страницу **Параметры OPC UA** и выполнить настройку, как показано на рисунке 68, после чего нажать кнопку **Применить конфигурацию**.

Разрешить OPC UA: ☒

Локальный IP адрес: По умолчанию

Порт:

Мин. период опроса элемента, мс:

Политика безопасности: Default (None)

Аутентификация: Disabled

Разрешить простой текстовый пароль: Default (No)

Имя хоста в качестве имени сервера: ☐

Имя сервера:

Рисунок 68 – Активация сервера OPC UA в контроллере (без защиты соединения)

9. После перезапуска контроллера на ПК запустить клиентское приложение OPC UA, например, Unified Automation UaExpert, командой меню **Server – Add** добавить описание сервера с параметрами, показанными на рисунке 69, скорректировав IP-адрес контроллера в поле **Endpoint Url** в соответствии с тем, какой порт сети Ethernet контроллера находится в одной подсети с ПК и какой для него установлен IP-адрес, после чего закрыть диалоговую панель **Add Server** нажатием **OK**.

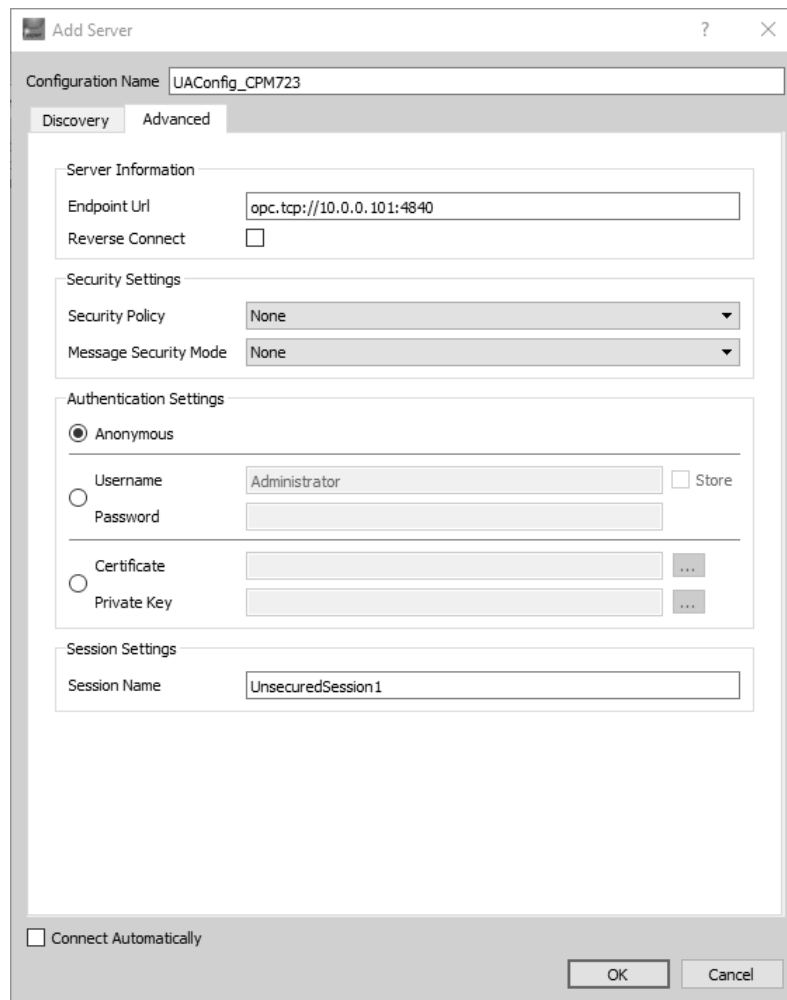



Рисунок 69 – Анонимное соединение с сервером без шифрования

10. В контекстном меню созданной конфигурации соединения в UaExpert выбрать команду **Connect**, как показано на рисунке 70. При успешном соединении пиктограмма сервера изменится на , а в области **Address Space** будут отображены элементы адресного пространства сервера OPC UA контроллера: *DeviceSet* и *Server*.

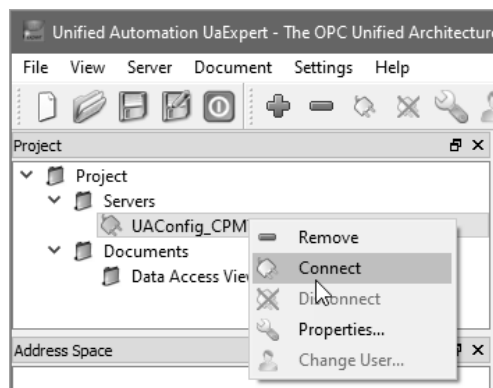


Рисунок 70 – Установка соединения с контроллером по OPC UA

11. В области **Address Space** раскрыть элемент *DeviceSet – CPMXXX-ZZ Programmable ... – Resources – Application – Programs* и перетащить последний элемент (*Programs*) в область **Data Access View**, нажав **Yes** в диалоговой панели **Recursively Add Nodes**. В области **Data Access View** появятся четыре переменные, добавленные в символьную конфигурацию приложения при выполнении действий по пунктам 4 – 5, как показано на рисунке 71, значение переменной *dwCycleCounter* будет изменяться.



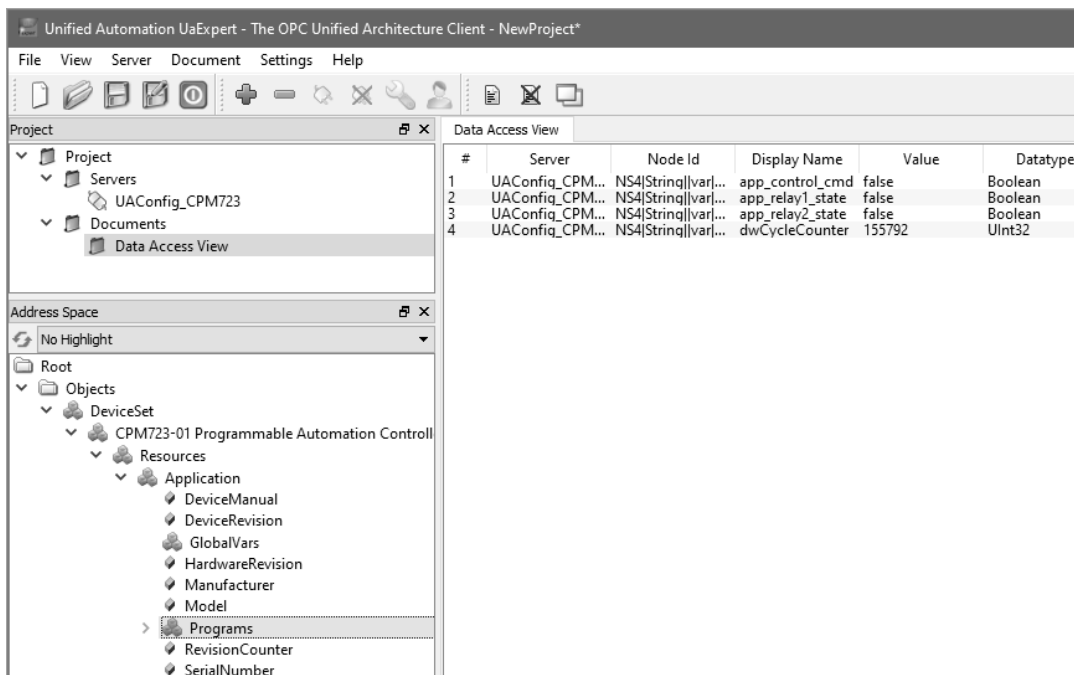


Рисунок 71 – Доступ к переменным символьной конфигурации через OPC UA

Более подробная информация о символьной конфигурации приведена в подразделе **Руководство по интерфейсу пользователя – Объект 'Символьная конфигурация'** интерактивной справочной системы IDE МЭК 61131-3.

При количестве переменных в символьной конфигурации около 1000 установка связи между клиентом OPC UA и контроллером или сервером OPC и контроллером может занимать длительное время.

При количестве переменных более 10000, включенных в символьную конфигурацию контроллеров Fastwel, устойчивый обмен данными не гарантируется.

#### 4.2.7. Глобальные переменные

В приложениях IDE МЭК 61131-3 могут использоваться глобальные переменные (VAR\_GLOBAL), которые создаются в именованных списках на уровне приложения целевого устройства или на уровне проекта.

Именованный список глобальных переменных на уровне приложения целевого устройства создается на вкладке **Устройства** командой контекстного меню **Добавление объекта – Список глобальных переменных** над приложением целевого устройства, как показано на рисунке 72.

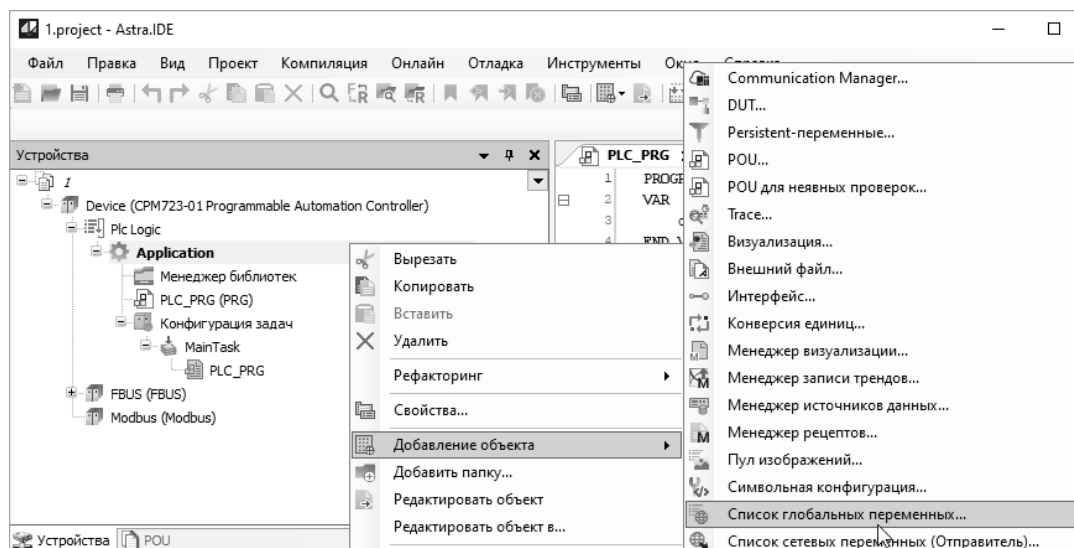


Рисунок 72 – Добавление списка глобальных переменных на уровне целевого устройства

Именованный список глобальных переменных на уровне проекта создается на вкладке **POU**, как показано на рисунке 73.

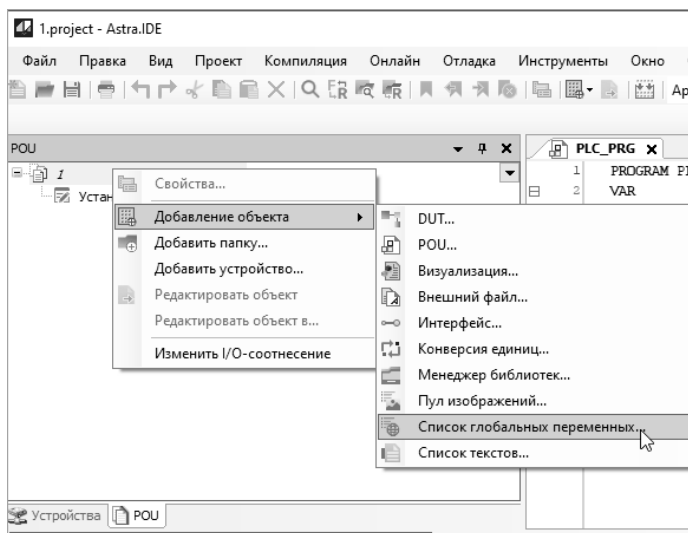


Рисунок 73 – Добавление списка глобальных переменных на уровне проекта

Для доступа к глобальным переменным в разных списках используются полные имена переменных, включающие имя списка, точку, а затем имя переменной в списке. Например, имя *GVL.glSensorBound* предназначено для доступа к переменной *glSensorBound*, декларированной в списке глобальных переменных с именем *GVL*.



При чтении и записи значений глобальных переменных в POU, вызываемых из разных задач с разными приоритетами, алгоритмы приложения могут функционировать неправильно с непредсказуемым результатом (см. п. 6.4.3.8).

Более подробная информация об использовании глобальных переменных приведена в подразделе **Руководство по программированию – Переменные – Глобальные переменные - VAR\_GLOBAL** интерактивной справочной системы IDE МЭК 61131-3.

#### 4.2.8. Энергонезависимые переменные

В системе исполнения контроллеров Fastwel поддерживаются *энергонезависимые* переменные (VAR RETAIN) и *энергонезависимые сохраняемые* переменные (VAR PERSISTENT, VAR RETAIN PERSISTENT).

Оба вида энергонезависимых переменных размещаются в сегменте энергонезависимых переменных общим размером 131048 байт, реализованном на базе микросхемы статической памяти с линейным доступом и резервным питанием от батареи (CPM723) или на основе микросхемы энергонезависимой памяти с линейным доступом MRAM, не требующей резервного питания от батареи (CPM810, CPM823, CPM833).



Питание от батареи подается на микросхему статической памяти только при отсутствии основного питания контроллера. Срок службы батареи, по данным производителя, составляет от 4,5 до 10 лет в зависимости от температуры окружающего воздуха и времени нахождения контроллера в выключенном состоянии. При меньших значениях температуры обеспечивается меньший срок службы батареи.


Статус исправности батареи может быть получен в коде приложения путем использования функции SysTimeRtcControl из библиотеки SysTimeRtc, следующим образом:

```
VAR
  bBatteryStatus   : BOOL;
  iControlTag      : DINT;
  iControlResult   : DINT;
  iec_res          : RTS_IEC_RESULT;
END_VAR
```



```
(* Получаем статус исправности литиевой батареи. *)
(* ВНИМАНИЕ! Это очень дорогая операция! *)
iControlTag := 0;
iec_res := SysTimeRtc.SysTimeRtcControl(iControlTag, iControlResult);
bBatteryStatus := CmpErrors.Errors.ERR_OK = iec_res AND 0 <> iControlResult;
```

В данном фрагменте кода исправный статус батареи индицируется значением TRUE переменной *bBatteryStatus*.

	<p>Время выполнения функции SysTimeRtc.SysTimeRtcControl при получении статуса батареи составляет до 2 мс.</p>
---	--

Примеры получения статуса батареи имеются в проектах DateTimeUtilities.project, из состава Fastwel PLC Application Toolkit (см. таблицу 4).

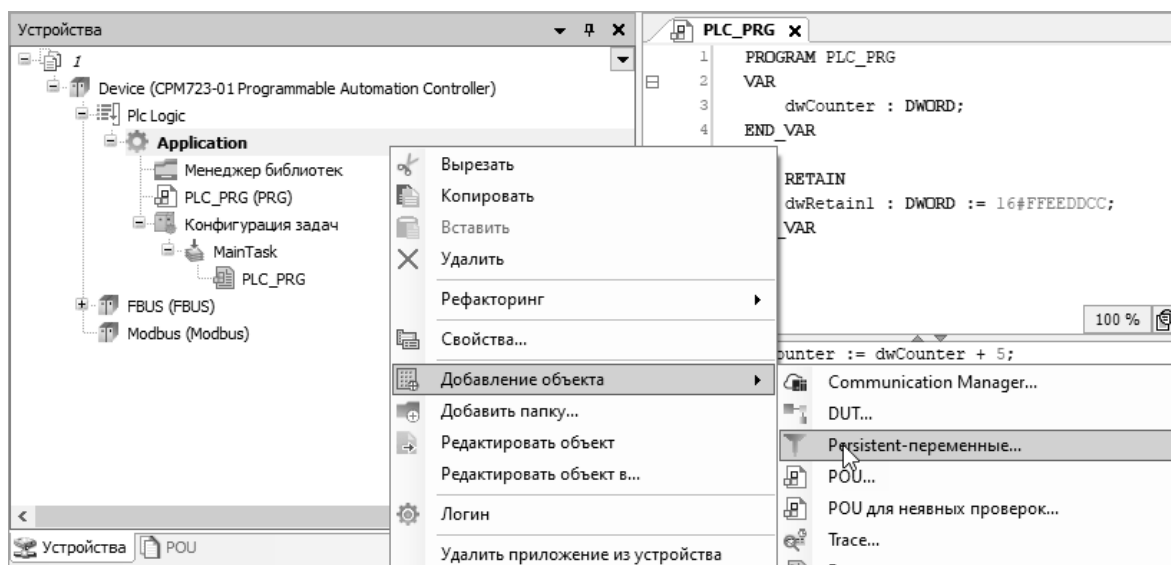
Энергонезависимые переменные могут объявляться в качестве внутренних переменных программ и функциональных блоков в декларации VAR RETAIN и занимать весь сегмент энергонезависимых переменных. Значения энергонезависимых переменных сохраняются неизменными после сброса приложения, перезапуска контроллера, в том числе выключением и повторным включением питания, а также после онлайн-изменения приложения.

Энергонезависимые переменные VAR RETAIN получают начальные значения, указанные при их декларации, после "холодного" сброса приложения по команде **Онлайн – Сброс холодный**, после вызова функции FwPlatformReset с параметром F\_RESET\_COLD (см. п. 6.4.3.2), а также в случае обновления приложения методом полной загрузки.

Энергонезависимые сохраняемые переменные (VAR\_GLOBAL PERSISTENT RETAIN) сохраняют свои значения после выключения питания, сброса и "холодного" сброса контроллера, после загрузки обновленного приложения методом онлайн-замены и полной загрузки.

Энергонезависимые сохраняемые переменные VAR PERSISTENT RETAIN должны декларироваться в специальном списке *Persistent-переменные*, добавляемом в приложение соответствующей командой контекстного меню, как показано на рисунке 74.

Если в проекте отсутствуют другие энергонезависимые переменные, то для энергонезависимых сохраняемых переменных доступно не более 131004 байта в сегменте энергонезависимых переменных. Это связано с тем, что для управления хранением значений энергонезависимых сохраняемых переменных требуется дополнительное пространство для хранения служебной информации.



**Рисунок 74 – Добавление в проект списка энергонезависимых сохраняемых переменных**

Если требуется иметь энергонезависимые сохраняемые переменные в качестве внутренних переменных программы или функционального блока, то необходимо выполнить следующие действия:

1. Объявить одну или несколько переменных в области деклараций требуемой программной единицы в отдельном блоке VAR RETAIN PERSISTENT. Например:  

```
PROGRAM PLC_PRG

VAR RETAIN PERSISTENT
    dwMyPersistentVar : DWORD;
END_VAR
```
2. Если это не было сделано ранее, добавить в приложение список энергонезависимых сохраняемых переменных командой **Добавление объекта – Persistent-переменные**, как показано на рисунке 74.
3. Открыть редактор списка энергонезависимых сохраняемых переменных, дважды щелкнув на элементе, соответствующем созданному списку *PersistentVars* (имя списка может быть другим) в дереве проекта, и выполнить команду меню **Компиляция – Генерировать код**.
4. Установить текстовый курсор во внутреннюю область окна редактора списка энергонезависимых сохраняемых переменных и выполнить команду меню **Объявления – Добавить все пути экземпляров**, как показано на рисунке 75.

Автоматически сформированные квалифицированные имена всех сохраняемых переменных, принадлежащих программным единицам, будут помещены в список энергонезависимых сохраняемых переменных. Например, для упомянутой выше переменной *dwMyPersistentVar*, объявленной в программе PLC\_PRG, будет добавлена следующая декларация:

```
VAR_GLOBAL PERSISTENT RETAIN
    // Сгенерированный путь persistent-переменной
    PLC_PRG.dwMyPersistentVar: DWORD;
END_VAR
```

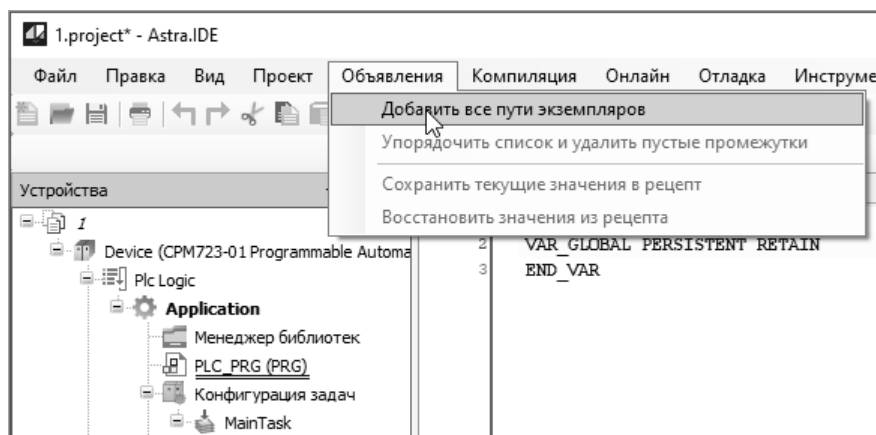


Рисунок 75 – Определение экземпляров внутренних энергонезависимых сохраняемых переменных




5. При необходимости установить начальные значения энергонезависимым сохраняемым переменным в списке, например:

```
VAR_GLOBAL PERSISTENT RETAIN
    // Сгенерированный путь persistent-переменной
    PLC_PRG.dwMyPersistentVar: DWORD := 16#FFFF0001;
END_VAR
```

Значения энергонезависимых сохраняемых переменных остаются неизменными после всех видов сброса приложения (кроме заводского), после полного перезапуска контроллера, в том числе выключением и повторным включением питания, а также после онлайн-изменения приложения и обновления приложения методом полной загрузки.

При добавлении в проект новых энергонезависимых сохраняемых переменных, а также при изменении структурных типов данных, использованных при объявлении энергонезависимых сохраняемых переменных, компилятор стремится сохранить неизменным местоположение ранее созданных энергонезависимых сохраняемых переменных в памяти, чтобы при любом виде обновления приложения в контроллере их значения оставались неизменными. Это может привести к фрагментации сегмента энергонезависимых переменных, в результате чего размера свободного пространства может не хватить для размещения новых энергонезависимых переменных, либо для имеющихся в приложении энергонезависимых сохраняемых переменных будет невозможно сохранить последние значения, что приведет к присвоению им начальных значений.

Команда **Компиляция – Очистить все** и последующая повторная генерация кода приложения иногда позволяют решить проблему нехватки энергонезависимой памяти из-за фрагментации, однако не всегда гарантирует сохранение значений ранее созданных энергонезависимых сохраняемых переменных.

	<p>Ситуация, когда при загрузке измененного приложения в контроллер невозможно сохранить местоположение и значения энергонезависимых сохраняемых переменных наиболее вероятна и характерна при использовании массивов и, особенно, массивов структур и/или функциональных блоков после изменения состава полей структурных типов и/или функциональных блоков и/или количества элементов массивов.</p>
	<p>При чтении и записи значений глобальных энергонезависимых и энергонезависимых сохраняемых переменных в ROU, вызываемых из разных задач с разными приоритетами, алгоритмы приложения могут функционировать неправильно с непредсказуемым результатом (см. п. 6.4.3.8).</p>
	<p>Присвоение начальных значений (инициализация) глобальным энергонезависимым и энергонезависимым сохраняемым переменным при запуске приложения выполняется системой исполнения после начальной установки всех прочих переменных, включая экземпляры функциональных блоков и переменные, сопоставленные с адресами во входном и выходном образе процесса (см. п. 4.3).</p> <p>Принудительная инициализация переменных, соотнесенных с входными (%I*) или выходными (%Q*) адресами в образе процесса, значениями глобальных энергонезависимых и энергонезависимых сохраняемых переменных, должна выполняться либо в обработчике системного события <i>LegacyOnInit</i>, либо в функции с атрибутами <i>linkalways</i> и <i>call_after_global_init_slot</i> со значением <i>slot</i> более 1000.</p> <p>Например, для инициализации регистров хранения сервера MODBUS, с которыми соотнесена переменная <i>dwMbInputs</i>, значениями энергонезависимой сохраняемой переменной <i>dwModbusSetpoints</i> может использоваться следующая функция:</p> <pre>{attribute 'linkalways'} {attribute 'call_after_global_init_slot' := '1001'} FUNCTION InitHoldings : BOOL     SysMem.SysMemCpy ( ADR(dwMbInputs) ,                       ADR(dwModbusSetpoints) ,                       SIZEOF(dwMbInputs) ); END_FUNCTION</pre> <p>Информация об обработке системных событий приведена в п. 4.5.</p>

Информация об энергонезависимых переменных приведена в подразделе **Руководство по программированию – Переменные – Перманентная переменная – PERSISTENT** и в подразделе **Руководство по программированию – Переменные – Энергонезависимая переменная – PERSISTENT** интерактивной справочной системы IDE МЭК 61131-3.

Информация об атрибутах приведена в подразделе **Руководство по программированию – Прагмы – Прагмы атрибутов** интерактивной справочной системы IDE МЭК 61131-3.

#### 4.2.9. Сетевые переменные

Система исполнения контроллеров Fastwel поддерживает механизм сетевого обмена данными между контроллерами, принадлежащими одной подсети, средствами специального протокола прикладного уровня IDE МЭК 61131-3, основанного на передаче широкополосных сообщений.

Данные, подлежащие передаче другим узлам сети с использованием механизма сетевых переменных, должны быть объявлены в виде переменных в *списке передаваемых глобальных сетевых переменных*.

Список передаваемых глобальных сетевых переменных добавляется в конфигурацию приложения командой контекстного меню **Добавление объекта – Список сетевых переменных (Отправитель)**, как показано на рисунке 77. При выполнении данной команды меню на экран монитора ПК выводится

диалоговая панель **Добавить Список сетевых переменных (Отправитель)**, показанная на рисунке 77, в которой следует настроить параметры передачи переменных списка, в том числе:

в поле **Имя** ввести имя списка переменных, например, *SenderVarList*;

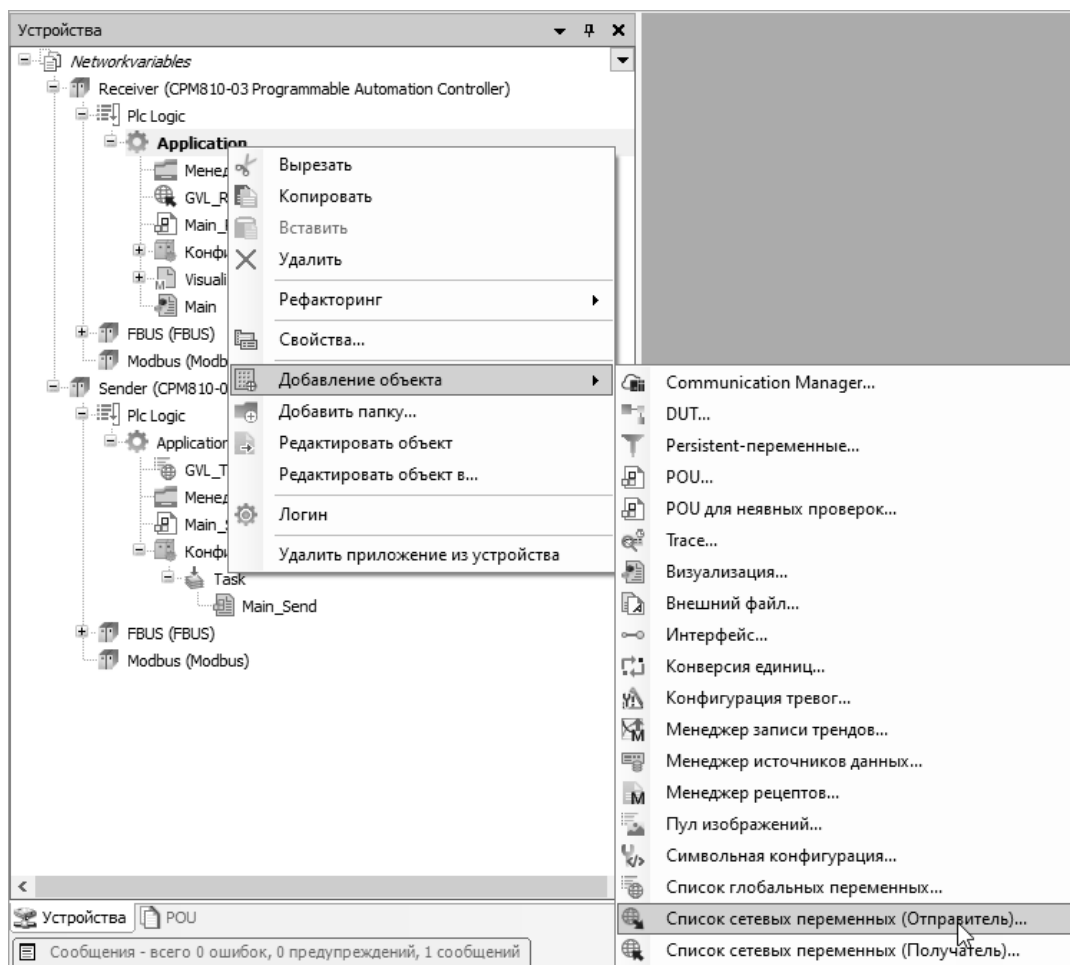
в поле **Тип сети** выбрать *UDP*;

в поле **Задача** указать циклическую задачу приложения, которая будет осуществлять передачу данных;

в поле **Идентификатор списка** ввести числовой идентификатор, который отличается от ранее использованных в проекте идентификаторов списков передаваемых переменных;

установить флажок **Упаковка переменных**;

выбрать условие передачи путем установки опций **Циклическая передача**, **Передача по изменению** и/или **Передача по событию**.



**Рисунок 76 – Добавление списка передаваемых глобальных сетевых переменных**

По завершении настройки параметров списка передаваемых переменных следует нажать кнопку **Установки** справа от поля **Тип сети** и дополнительно задать широковещательный адрес и номер порта для сетевого запроса передачи переменных в диалоговой панели **Установки сети**.

По умолчанию предлагается использовать глобальный широковещательный запрос с адресом 255.255.255.255. Рекомендуется ограничить распространение широковещательного запроса только подсетью, в которую входят контроллеры, для чего в качестве IP-адреса следует использовать адрес направленного широковещательного запроса используемой подсети. Значение адреса направленного широковещательного запроса подсети образуется путем применения побитового OR инвертированной маски подсети к адресу подсети, т.е. все биты адреса подсети, соответствующие нулям в маске, устанавливаются в 1.

**Рисунок 77 – Создание и настройка параметров списка передаваемых глобальных переменных**

Например, для адреса подсети 10.0.0.0/8 адрес направленного широковещательного запроса определяется следующим образом:

$(10.0.0.0 \text{ OR } (\text{NOT } 255.0.0.0)) \rightarrow 10.255.255.255$

Для адреса подсети 172.16.0.0/12 адрес направленного широковещательного запроса определяется так:


$(172.16.0.0 \text{ OR } (\text{NOT } 255.240.0.0)) \rightarrow 172.31.255.255$

Наконец, для адреса подсети 192.168.0.0/24 адрес направленного широковещательного запроса:

$(192.168.0.0 \text{ OR } (\text{NOT } 255.255.255.0)) \rightarrow 192.168.0.255$



Результат выполнения команды *ipinfo* **Оболочки ПЛК** в блоке IP-параметров каждого интерфейса содержит значение адреса направленного широковещательного запроса в строке *Broadcast*.

По завершении настройки следует нажать кнопку **Добавить** в диалоговой панели создания списка сетевых переменных, и вновь созданный список появится в дереве проекта со значком , а в правой области главного окна среды разработки будет открыта вкладка редактора списка глобальных переменных, в который нужно добавить объявления переменных, чьи значения должны передаваться в сеть.

После ввода деклараций передаваемых переменных можно связать список с файлом экспорта, который впоследствии будет подключен к спискам *принимаемых глобальных сетевых переменных* в конфигурации приложения других узлов сети. Данный файл экспорта будет автоматически повторно формироваться всякий раз при выполнении команды меню **Компиляция – Перекомпиляция**.

Для настройки связи с файлом экспорта необходимо щелкнуть правой кнопкой мыши над списком передаваемых глобальных сетевых переменных в дереве проекта, в контекстном меню выбрать **Свойства**, а затем вкладку **Связь с файлом** в появившейся диалоговой панели свойств списка сетевых переменных, как показано на рисунке 78.

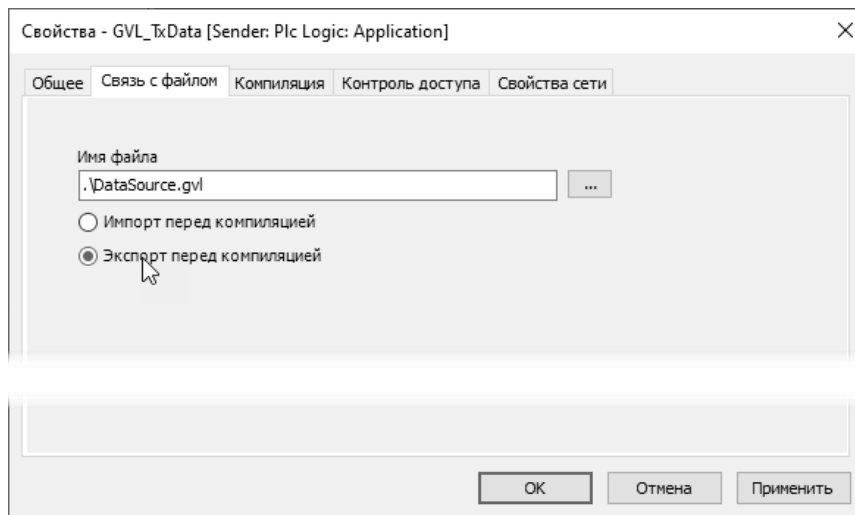


Рисунок 78 – Настройки связи списка передаваемых сетевых переменных с файлом экспорта

Для того, чтобы файл экспорта формировался в каталоге размещения файла проекта, в поле **Имя файла** следует вручную ввести путь и имя файла экспорта с использованием относительного пути, как показано на рисунке 78. Нажатие кнопки **Применить** приведет к формированию файла экспорта.

Для завершения настройки списка передаваемых переменных следует нажать кнопку **ОК**.

Для приема значений переменных на других узлах сети необходимо добавить в конфигурацию их приложений *списки принимаемых глобальных сетевых переменных*, которые либо непосредственно связаны со списком передаваемых переменных, либо связаны с файлом экспорта, ранее определенным для списка передаваемых переменных.



Рисунок 79 – Создание и настройка параметров списка принимаемых глобальных переменных

Список принимаемых глобальных сетевых переменных добавляется в конфигурацию приложения командой контекстного меню **Добавление объекта – Список сетевых переменных (Получатель)**. При выполнении данной команды на экран выводится диалоговая панель **Добавить Список сетевых переменных (Получатель)**, показанная на рисунке 79, в которой нужно определить имя списка переменных, задачу приложения, в контексте которой будет осуществляться прием значений переменных, и выбрать список передаваемых переменных, чьи значения требуется получать по сети.

Если приложения для всех контроллеров, участвующих в обмене сетевыми переменными, принадлежат одному проекту IDE МЭК 61131-3, то можно выбрать список передаваемых переменных по имени в поле **Отправитель**, как показано на рисунке 79. В ином случае в поле **Отправитель** следует

выбрать опцию *Импорт из файла*, и в текстовом поле **Импорт из файла** ввести относительный путь к имени файла экспорта, связанного со списком передаваемых переменных.

Настройка параметров списка принимаемых переменных завершается нажатием кнопки **Добавить**. При правильной настройке связи со списком передаваемых переменных в списке принимаемых переменных будут автоматически сформированы объявления переменных, значения которых будут приниматься по сети, как показано на рисунке 80.

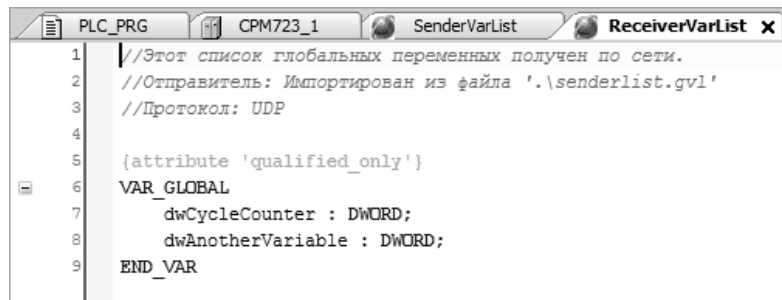


Рисунок 80 – Автодекларации сетевых переменных, принимаемых по сети

Если приложения для всех узлов сети принадлежат одному проекту IDE МЭК 61131-3, по окончании настройки списков передаваемых и принимаемых сетевых переменных имеется возможность загрузить приложения во все контроллеры командой **Онлайн – Множественная загрузка**.

	Перед множественной загрузкой в конфигурации приложения для каждого контроллера следует правильно установить путь связи среды разработки с каждым контроллером, иначе может произойти поочередная загрузка разных приложений проекта в один и тот же контроллер.
--	--

Пример простого проекта Networkvariables.project, содержащего конфигурацию приложений для двух контроллеров, обменивающихся данными с использованием механизма сетевых переменных, поставляется в Fastwel PLC Application Toolkit (см. таблицу 4).

	В проектах Networkvariables.project из Fastwel PLC Application Toolkit для списка передаваемых глобальных сетевых переменных <i>GVL_TxData</i> используется направленное широковещательное сообщение с адресом 10.255.255.255, т.е. контроллеры должны иметь IP-адреса в подсети 10.0.0.0/8.
	Если сетевые интерфейсы контроллера имеют IP-адреса в одной подсети, то передача сетевых переменных возможна только через интерфейс с IP-адресом LAN1.
	В проектах Networkvariables.project из Fastwel PLC Application Toolkit используется единственная переменная типа <i>StructuredNetworkVariable</i> , передаваемая по сети, при этом тип переменной объявлен на уровне проекта на вкладке <b>POU</b> . Таким образом, при необходимости изменения состава или типов полей структуры приложения для всех контроллеров проекте будут автоматически обновлены, и потребуются минимальные изменения в коде приложений для учета внесенных изменений.

Более подробная информация об использовании сетевых переменных приведена в подразделе **Использование управляющих сетей – Сетевые переменные** интерактивной справочной системы IDE МЭК 61131-3.

#### 4.2.10. Неподдерживаемые элементы программной модели приложения

Среда разработки IDE МЭК 61131-3, помимо описанных в настоящем подразделе элементов программной модели, позволяет добавлять в конфигурацию приложения ряд других элементов.

В системе исполнения контроллеров Fastwel не поддерживаются следующие типы элементов проектной информации:



*Redundancy Configuration**Конфигурация тревог (Alarm configuration)**Менеджер записи трендов (Trend recording manager)**Communication Manager**Менеджер источников данных (Data Server, Data Source Manager)**Менеджер рецептов (Recipe Manager)*

Использование форм визуализации в проектах IDE МЭК 61131-3 для контроллера CPM723-01, не имеющего подсистемы целевой визуализации, иллюстрируется в примерах программирования, поставляемых в Fastwel PLC Application Toolkit и отмеченных символом <sup>V</sup> в таблице 4.

### 4.3. Обмен данными с внешними устройствами

#### 4.3.1. Связь приложения с окружением

Внешние устройства, включая модули ввода-вывода, подчиненные узлы и коммуникационные объекты обмена данными сетевых протоколов и т.п. для приложения IDE МЭК 61131-3 являются так называемым *окружением*, поскольку явно не представлены в программной модели ГОСТ Р МЭК 61131-3. Взаимодействие приложения с окружением осуществляется через сегменты входных и выходных данных приложения, называемые входным и выходным образом процесса.

Программная модель ГОСТ Р МЭК 61131-3 неявно предполагает, что входные и выходные каналы внешних устройств, таких как модули ввода-вывода, коммуникационные объекты сетевых сервисов полевых шин, диагностические каналы различных подсистем системы исполнения и т.д., должны быть соотнесены или отображены (mapped) системой исполнения на сегменты входных и выходных данных.

*Соотнесены с сегментом входных данных* означает, что данные входных каналов внешних устройств должны некоторым способом передаваться системой исполнения во входной образ процесса в начале цикла каждой задачи приложения, вызывающей программные единицы, которым для работы нужны входные данные внешних устройств, такие как значения входных сигналов, состояния дискретных входов и т.п.

*Соотнесены с сегментом выходных данных* означает, что данные из выходного образа процесса должны некоторым способом передаваться системой исполнения в выходные каналы внешних устройств в конце цикла задач приложения, чьи программные единицы формируют аналоговые, дискретные, частотные и другие управляющие воздействия.

Для доступа к значениям входных и выходных каналов устройств в коде приложении должны использоваться так называемые *переменные прямого представления* (или "прямо представленные" переменные), снабженные спецификаторами размещения: %I – для сегмента входных данных; %Q – для сегмента выходных данных. После спецификатора сегмента %I или %Q следует представление адреса (смещения) и размера элемента данных в образе процесса, например, %IB1023.

Система исполнения приложений контроллеров Fastwel имеет сегменты входных и выходных данных, максимальный размер каждого из которых составляет 512 кбайт. Это означает, что приложение, загруженное в контроллер, потенциально может в цикле некоторой задачи прочитать из окружения до 512 кбайт входных данных и записать в окружение до 512 кбайт выходных данных.

Среда разработки IDE МЭК 61131-3 с установленным Fastwel PLC Application Toolkit поддерживает несколько способов соотнесения программных единиц приложения с окружением:

1. Декларация глобальных переменных или переменных программных единиц типа *PROGRAM*, непосредственно соотнесенных с адресами входного или выходного образа процесса.

```
VAR
    byteInput AT %IB42 : BYTE;
    wordOutput AT %QB1 : WORD;
END_VAR
```

Данный способ соотнесения иллюстрируется рисунком 81 и наиболее предпочтителен в случае, если требуется групповая программная обработка большого количества входных и выходных данных, декларированных в виде массивов структур. Групповая



обработка входных данных используется практически во всех проектах, поставляемых в качестве примеров программирования для библиотеки FastwelFbusIO в Fastwel PLC Application Toolkit.




	<p>Для указания адреса элемента данных в образе процесса в приложениях IDE МЭК 61131-3 для контроллеров Fastwel используются байтовые смещения относительно начала образа процесса.</p> <p>Например, %IB42 адресует байт (B) по байтовому смещению 42, начиная с 0, от начала входного образа процесса, %IW60 адресует слово (W) по байтовому смещению 60, начиная с 0, от начала входного образа процесса, а %QD28 адресует двойное слово (D) по байтовому смещению 28, начиная с 0, от начала выходного образа процесса.</p>
	<p>При использовании данного способа соотнесения требуется следить за возможным сдвигом адресов входных и выходных каналов при вставке и удалении в конфигурации приложения устройств с входными и выходными каналами.</p>



Рисунок 81 – Непосредственное соотнесение переменных с адресами в образе процесса

- Соотнесение имеющихся переменных программ и экземпляров функциональных блоков или имеющихся глобальных переменных с каналами во входном или выходном образе процесса, как показано на рисунке 82, в редакторе образа процесса на вкладке **Изменить I/O-соотнесение (Edit IO mapping)** или на вкладке **Соотнесение входов/выходов** редакторов устройств, добавленных в конфигурацию приложения. Двойной щелчок в ячейке **Переменная** в редакторе образа процесса приводит к появлению кнопки , позволяющей выбрать имеющуюся переменную, с которой требуется соотнести входной или выходной канал.

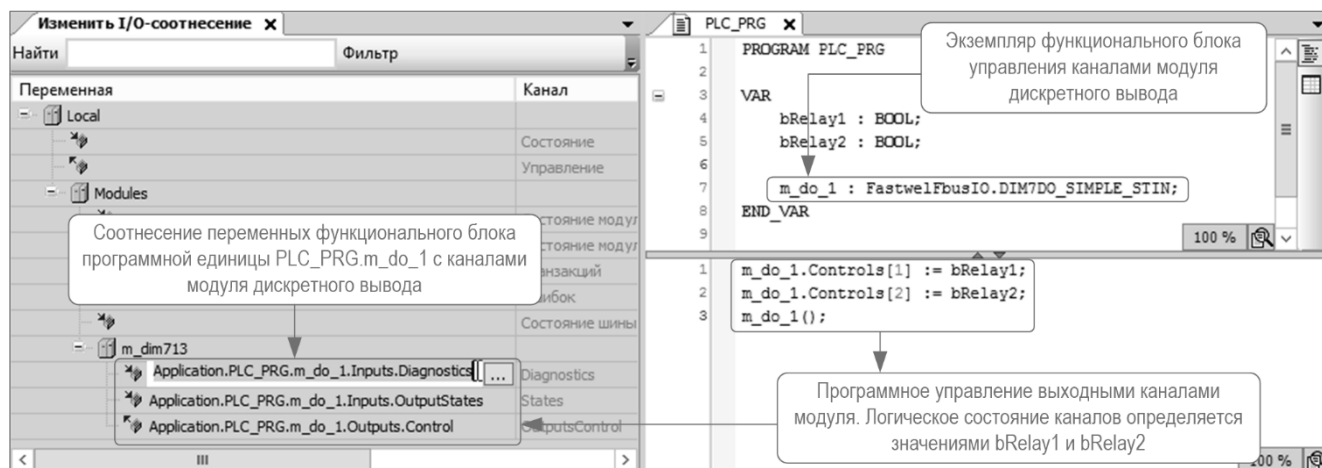


Рисунок 82 – Соотнесение имеющихся переменных экземпляра функционального блока с входными и выходными каналами

При соотнесении каналов с имеющимися переменными вставка и удаление элементов конфигурации приложения не требует от пользователя контроля сохранности ранее созданных соотнесений.

Недостатком данного способа является несколько более длительное, чем при использовании остальных способов соотнесения, время, требуемое на ввод и вывод данных при работе приложения из-за необходимости выполнения дополнительной операции копирования при вводе или выводе, а также более высокая по сравнению со способом 1 трудоемкость связывания большого количества переменных с каналами ввода-вывода при разработке проекта.



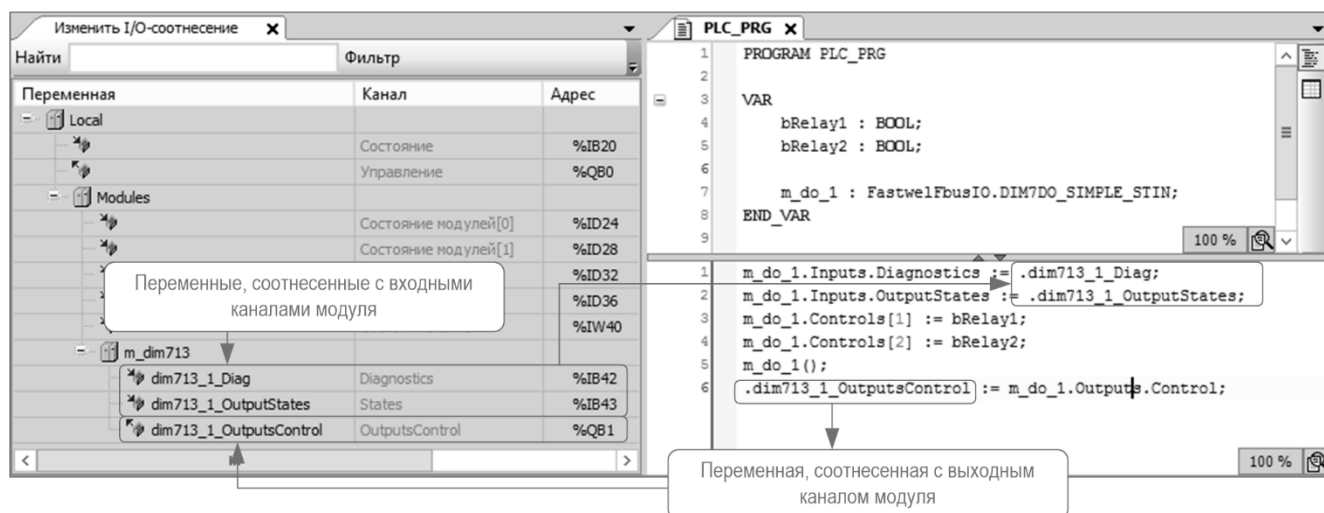
При использовании в нескольких задачах разного приоритета одной и той же глобальной переменной, соотнесенной данным способом с некоторым входным каналом, алгоритм менее приоритетной задачи с высокой вероятностью будет работать неправильно (см. п. 6.4.3.8).

3. Создание ссылочных переменных, соотнесенных с каналами во входном и выходном образе процесса, непосредственно в редакторе образа процесса или редакторе соотнесений, как показано на рисунке 83.

При использовании данного способа также не требуется следить за сдвигами адресов каналов модулей ввода-вывода и коммуникационных объектов при изменении структуры образа процесса, однако для соотнесения данным способом вручную пользователю доступны только переменные примитивных типов.



Для элементов конфигурации версии 1.0.0.3 и выше, описывающих модули ввода-вывода (устройство *Мастер FBUS*), в приложении автоматически создаются ссылочные переменные `<имя_модуля>_inr` и `<имя_модуля>_out` соответствующих структурных типов из библиотеки `FastwellFbusIO`, где `имя_модуля` – имя элемента, вставляемого в дерево проекта при добавлении или вставке описания модуля ввода-вывода.



**Рисунок 83 – Создание входных и выходных переменных в редакторе образа процесса**

4. Соотнесение с адресами в образе процесса недоопределенных конфигурируемых переменных прямого представления, принадлежащих программным единицам, в секции `VAR_CONFIG`, объявленной в отдельном списке глобальных переменных. Данный способ иллюстрируется рисунком 84. В стандарте ГОСТ Р МЭК 61131-3 недоопределенные конфигурируемые переменные прямого представления называются "частично определенными" или "не локализованными входными и выходными переменными", которые имеют спецификатор размещения в образе процесса `%I` или `%Q` с адресом, замененным на символ `'*'`.

Данный способ обычно используется для соотнесения недоопределенных конфигурируемых переменных прямого представления функциональных блоков и практически не отличается от способа 1.

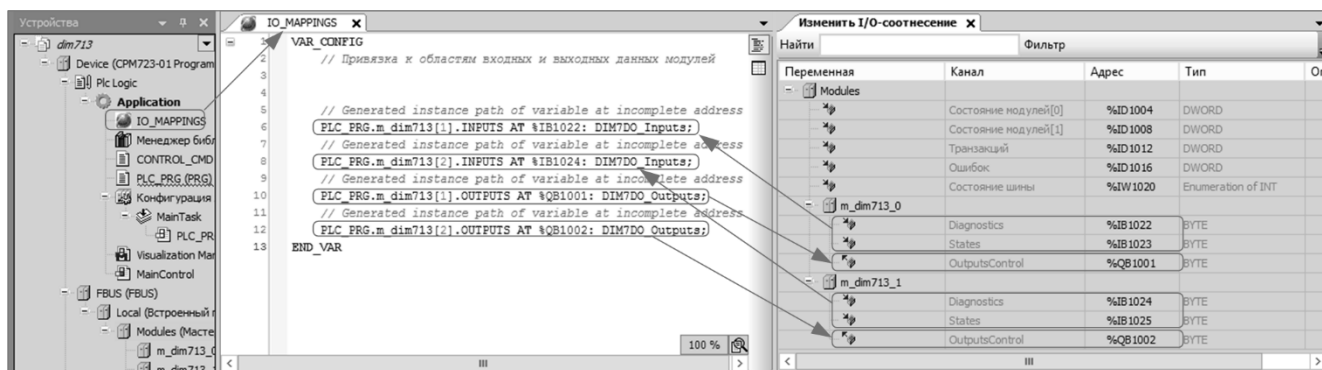


Рисунок 84 – Привязка недоопределенных конфигурируемых переменных в секции VAR\_CONFIG

Для автоматической вставки полных путей ко всем недоопределенным переменным прямого представления в секцию *VAR\_CONFIG* конфигурации приложения следует открыть список глобальных переменных, содержащий секцию *VAR\_CONFIG*, скомпилировать проект командой **Компиляция – Генерировать код** и выполнить команду меню **Объявления – Добавить все пути экземпляров (Declarations – Add all instance paths)**, после чего для каждого пути к переменной ввести фактический адрес в образе процесса для соотносимых каналов ввода или вывода.

При использовании способов 1 и 4 имеется возможность изменения начальных адресов областей, занимаемых каналами сервисов ввода-вывода во входном и выходном образе процесса, что позволяет избежать сдвигов адресов входных и выходных каналов одного сервиса при изменении структуры части образа процесса, занимаемой каналами другого сервиса. Для изменения адресов сервиса ввода-вывода в образе процесса, занимаемых локальным и удаленным портом шины FBUS:

- Щелкнуть правой кнопкой мыши над элементом дерева проекта, соответствующим порту FBUS, или над элементом (*CPMXXX-ZZ Programmable Automation Controller*) и в контекстном меню выбрать команду **Изменить I/O-соотнесение**.
- На вкладке редактора образа процесса **Изменить I/O-соотнесение** найти каналы *Состояние* и *Управление* порта FBUS, показанные на рисунке 85.

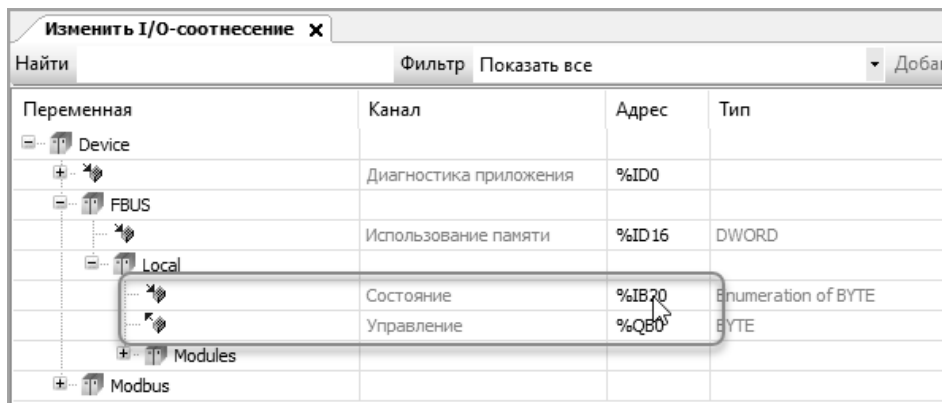



Рисунок 85 – Начальные адреса образа процесса порта FBUS

- Дважды щелкнуть левой кнопкой мыши в ячейке *Состояние* : **Адрес** и ввести новое значение начального адреса во входном образе процесса (например, 1000), с которого будут следовать адреса входных каналов модулей ввода-вывода, обслуживаемых данным портом FBUS, после чего нажать Enter. Значение адреса будет изменено на заданное, а слева от нового значения будет отображен символ **M**.
- Дважды щелкнуть левой кнопкой мыши в ячейке *Управление* : **Адрес** и ввести новое значение начального адреса в выходном образе процесса (например, 1000), с которого будут следовать адреса выходных каналов модулей ввода-вывода, обслуживаемых данным портом FBUS, и нажать Enter. Значение адреса будет изменено на заданное, а слева от нового значения будет отображен символ **M**.

В дальнейшем при добавлении в конфигурацию приложения описаний удаленных портов шины FBUS и связанных с ними модулей ввода-вывода адреса каналов модулей ввода-вывода, обслуживаемых локальным портом FBUS, не изменятся, если каналы вновь добавленных элементов конфигурации в

образе процесса не перекрывают адреса расположенных ниже их областей, значения которых заданы вручную.


Таким же образом могут быть изменены начальные адреса в образе процесса элементов конфигурации, соответствующих сервисам протоколов MODBUS, а также для описаний каждого подчиненного узла в конфигурации мастера протокола MODBUS TCP или MODBUS.

	<p>Не допускается использовать динамическое соотнесение переменных приложения с адресами каналов во входном (%I) образе процесса путем использования ключевого слова ADR. Например, вот этот код работать не будет:</p> <pre> VAR   pInputsStart : POINTER TO BYTE;   pTmpPtr : POINTER TO BYTE;   input_byte_1 : BYTE;   input_byte_2 : BYTE; END_VAR // Получаем абсолютный адрес во входном образе процесса. pInputsStart := ADR(%IB42); // Пробуем ввести данные из входного образа процесса... pTmpPtr := pInputsStart; // Здесь получится, т.к. ссылка на %IB42 сгенерирована компилятором. input_byte_1 := pTmpPtr^; // Предположительно перемещаемся на следующий адрес во входном образе // процесса. pTmpPtr := pTmpPtr + SIZEOF(pTmpPtr^); // Здесь не сработает, т.к. компилятор не сгенерировал ссылку на этот адрес. input_byte_2 := pTmpPtr^;</pre>
---	--

#### 4.3.2. Ввод и вывод данных

В настоящем подразделе приведено описание механизма обмена данными приложения с окружением, реализованного в системе исполнения приложений контроллеров Fastwel.

Основная особенность механизма обмена данными приложения с окружением состоит в том, что при наличии в приложении более одной задачи каждая задача, программные единицы которой любым способом соотнесены с входными каналами внешних устройств, имеет собственный, независимый от других, сегмент входных данных. Собственный сегмент входных данных каждой задачи будет далее называться *персональным входным сегментом* или *персональным сегментом*. Глобальный сегмент входных данных, на который могут ссылаться функции обработки системных событий приложения, и из которого происходит чтение входных данных приложения при мониторинге из среды разработки, будет продолжаться называться *входным образом процесса*.


	<p>Для контроллеров Fastwel с СПО версии 3.1.x.x и ниже для каждой задачи приложения создается персональный входной сегмент.</p> <p>Начиная с версии 3.2.x.x СПО контроллеров Fastwel, персональные входные сегменты задач создаются только при наличии в приложении более одной задачи.</p>
---	--

Таким образом, при работе нескольких циклических, ациклических и свободно-исполняемых задач разного приоритета, ссылающихся на одни и те же %I-адреса в окружении, задачи более высокого уровня приоритета (с меньшим абсолютным значением приоритета) не смогут испортить входные данные окружения, вводимые или введенные ранее задачами меньшего уровня приоритета (с большим абсолютным значением приоритета).

Этим обстоятельством объясняется предупреждение в п. 4.3.1 о недопустимости использования динамического соотнесения переменных приложения с адресами каналов во входном (%I) образе процесса, поскольку операция ADR(%IB42), выполненная в разных задачах, даст разные значения адресов во входном образе процесса. Кроме того, динамическое соотнесение не позволяет компилятору среды разработки правильно сгенерировать ссылки в коде приложения на образ процесса.

Вторая особенность механизма обмена данными приложения с окружением состоит в том, что в системе исполнения приложений контроллеров Fastwel системный сервис, обслуживающий обмен данными с модулями ввода-вывода, сервисы мастера и подчиненного узла протоколов MODBUS и MODBUS TCP и другие (в том числе не реализованные в настоящий момент) сервисы ввода-вывода, функционируют в собственных потоках исполнения операционной системы асинхронно и независимо

по отношению к циклическим, ациклическим и свободно-исполняемым задачам приложения IDE МЭК 61131-3, загруженного в контроллер. Таким образом, понятие "задача цикла шины" (bus cycle task), определенное в IDE МЭК 61131-3, не имеет смысла применительно к приложениям для контроллеров Fastwel.

	<p>Понятие "Задача цикла шины" в принципе не имеет смысла, если у контроллера имеется нескольких независимых шин с возможностью разночастотного или ациклического событийно-ориентированного обмена данными.</p>
---	--

Для обмена данными между образом процесса и внешними устройствами, промышленными сетями и другими элементами конфигурации приложения, имеющими входные и выходные каналы в образе процесса, в системе исполнения используется буферизация входных и выходных данных с механизмом защиты буферной памяти от совместного доступа со стороны задач приложения и потоков операционной системы, обслуживающих обмен с внешними устройствами. Таким образом обеспечивается асинхронность и независимость функционирования задач приложения и системных сервисов ввода-вывода.

Например, сервис, обслуживающий обмен данными с модулями ввода-вывода, содержит по одному потоку исполнения операционной системы на каждый порт FBUS, имеющийся в конфигурации приложения. Каждый поток исполнения производит циклический обмен данными с модулями с периодом, заданным для соответствующего данному порту элемента конфигурации типа *Мастер FBUS* значением параметра **Период опроса, мс**. В каждом цикла обмена с модулями поток сервиса FBUS выполняет следующие действия:



1. Захватывает блокировку буфера входного образа процесса. Блокировка предоставляется, если она не занята другим потоком/задачей. Если блокировка занята, поток ожидает ее освобождения.
2. Копирует данные входных каналов модулей, полученные в текущем цикле обмена, в буфер входного образа процесса.
3. Отпускает блокировку буфера входного образа процесса.
4. Захватывает блокировку буфера выходного образа процесса. Блокировка предоставляется, если она не занята другим потоком/задачей. Если блокировка занята, поток ожидает ее освобождения.
5. Забирает из буфера выходного образа процесса данные, ранее записанные в него одной или несколькими задачами приложения.
6. Отпускает блокировку буфера выходного образа процесса.
7. Передает групповой запрос модулям ввода-вывода.
8. Приостанавливается ("засыпает") до получения ответа от модулей. При получении ответа переходит к шагу 1.

Любая из нескольких задач приложения, вызывающая программные единицы, чьи переменные соотнесены с входными и выходными каналами модулей ввода-вывода, в каждом своем цикле выполняет следующие действия:

1. Захватывает блокировку буфера входного образа процесса. Блокировка предоставляется, если она не занята другим потоком/задачей. Если блокировка занята, задача ожидает ее освобождения.
2. Копирует данные из буфера входного образа процесса в собственный персональный сегмент входных (%I) данных, если в приложении более одной задачи, или глобальный сегмент входных данных – в противном случае.
3. Отпускает блокировку буфера входного образа процесса.
4. Выполняет программы из списка вызовов, определенного в конфигурации задачи.
5. Захватывает блокировку буфера выходного образа процесса. Блокировка предоставляется, если она не занята другим потоком/задачей. Если блокировка занята, задача ожидает ее освобождения.
6. Копирует данные, формируемые программными единицами данной задачи, из глобального сегмента выходных (%Q) данных в буфер выходного образа процесса.
7. Отпускает блокировку буфера выходного образа процесса.



8. Приостанавливается ("засыпает"), если нет условий для начала следующего цикла, заданных для задачи в конфигурации приложения. При наступлении условия начала цикла переходит к шагу 1.

	<p>Во всех перечисленных выше случаях, если освобождения блокировки начинает ожидать более приоритетный поток исполнения (задача), чем поток, ранее захвативший блокировку, приоритет потока, владеющего блокировкой, временно автоматически увеличивается до максимального приоритета среди всех потоков, ожидающих освобождения блокировки, пока владеющий поток не отпустит блокировку. Данный механизм реализован в операционной системе контроллера и называется <i>наследованием приоритетов</i> (priority inheritance).</p>
	<p>При наличии в приложении единственной задачи, начиная с версии 3.2.x.x СПО контроллеров Fastwel, персональный сегмент входных данных задачи не создается, и данные из буфера входного образа процесса помещаются в глобальный сегмент входных данных.</p>

### 4.3.3. Режимы ввода-вывода

В IDE МЭК 61131-3 изначально предусмотрено три режима ввода-вывода данных, один из которых устанавливается на вкладке **Установки ПЛК (PLC Settings)** редактора устройства типа (*CPMXXX-ZZ Programmable Automation Controller*) в выпадающем списке **Всегда обновлять переменные (Always update variables)**, как показано на рисунке 86:

1. Режим *Отключено (обновление только в задаче) (Disabled (update only if used in a task))* – данный режим устанавливается по умолчанию и обеспечивает ввод данных из окружения в персональные входные сегменты задач для нескольких задач или в глобальный сегмент – для одной задачи, и вывод данных из выходного образа процесса только для тех каналов внешних устройств (каналов модулей, коммуникационных объектов, каналов диагностики), описанных в конфигурации приложения, на которые есть *явные ссылки* в коде POU, вызываемых из задач приложения.

Ссылка на канал в коде POU считается явной, если соответствующая переменная, соотнесенная с каналом, используется в коде программной единицы в качестве операнда или в качестве результата выражения.

Например, пусть задача *MainTask* вызывает программу *PLC\_PRG* (содержит *PLC\_PRG* в списке вызовов), и в *PLC\_PRG* объявлены следующие переменные прямого представления:

```
VAR
    byteInput AT %IB42 : BYTE;
    wordOutput AT %QB1 : WORD;
END_VAR
```


Тогда для обновления значения элемента данных размером 1 байт по смещению 42 во входном сегменте *MainTask* во время исполнения приложения, в коде *PLC\_PRG* или в коде любой программной единицы, вызываемой из *PLC\_PRG*, должно присутствовать выражение, использующее переменную *byteInput*, например:

```
IF byteInput = 0 THEN
;
END_IF
```

Для обеспечения возможности вывода в окружение значения элемента данных размером 2 байта (длина переменной типа WORD) по смещению 1 из выходного образа процесса, в коде *PLC\_PRG* или в коде любой программной единицы, вызываемой из *PLC\_PRG*, должно присутствовать выражение, использующее переменную *wordOutput*, например:

```
wordOutput := wordOutput + 1;
```

Приведенные рассуждения о явных ссылках касаются всех способов соотнесения переменных с входными и выходными данными, описание которых приведено в п. 4.3.1.

	<p>Для формирования временных явных ссылок во время отладки могут использоваться следующие минимальные выражения в коде приложения:</p> <pre>byteInput; wordOutput;</pre> <p>При обнаружении таких выражений в исходном тексте компилятор выведет предупреждение о том, что данные выражения ничего не делают:</p> <p><b>C0139: Код 'byteInput;' не имеет действия. Это сделано намеренно?</b></p> <p>Данное предупреждение можно игнорировать.</p>
---	---

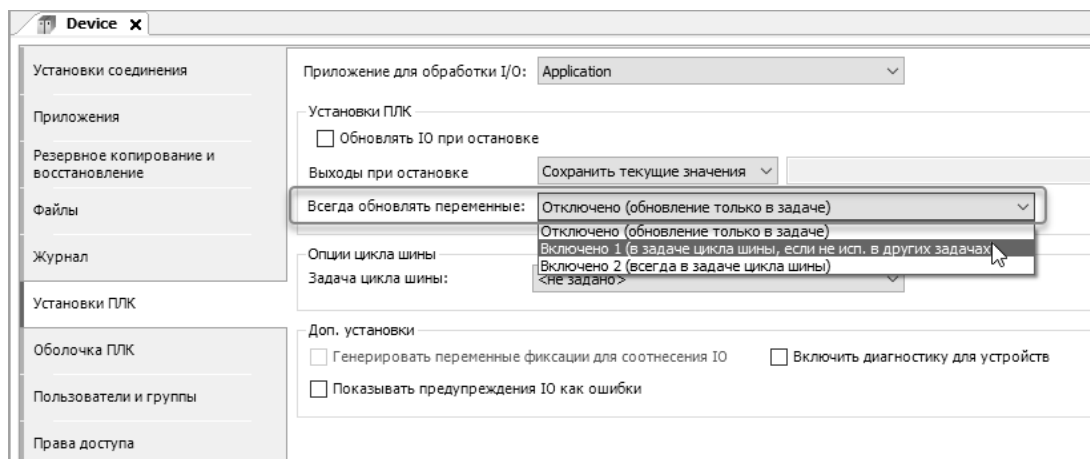





Рисунок 86 – Режимы ввода-вывода данных приложения


	<p>При использовании режима <i>Отключено (обновление только в задаче)</i> невозможен мониторинг значений входных и выходных каналов в IDE МЭК 61131-3, на которые нет ни одной явной ссылки ни в одной задаче приложения.</p>
--	---

Для выяснения наличия и просмотра явных ссылок на входные и выходные каналы в задачах приложения следует перейти на вкладку **Размещение задачи (Task Deployment)** в редакторе устройства типа (*CPMXXX-ZZ Programmable Automation Controller*), показанную на рисунке 87.

В столбцах с именами задач приложения и приоритетами задач в скобках справа от имен, расположенных справа от столбца **Канал**, для каналов ввода-вывода, на которые имеются ссылки в коде, вызываемом из соответствующих задач, отображается символ . Символ , отображаемый для каналов, обновляемых в задаче цикле шины, ничего не значит в приложениях для контроллеров Fastwel.

2. Режим *Включено 1 (в задаче цикла шины, если не исп. в других задачах) (Enabled 1 (use bus cycle task if not used in any task))* – в данном режиме ввод и вывод данных каналов ввода-вывода, на которые имеются явные ссылки в задачах приложения, выполняется таким же образом, как и режиме *Отключено (обновление только в задаче)*, с сохранением возможности мониторинга (просмотра текущих значений) в IDE МЭК 61131-3 остальных каналов ввода-вывода и записи в каналы вывода, на которые нет ни одной явной ссылки в задачах приложения.

Более подробная информация о мониторинге переменных приведена в п. 3.6.4 и п. 4.3.4.

	<p>Рекомендуется использовать режим <i>Включено 1</i> только во время отладки приложения, поскольку обновление большого количества неиспользуемых каналов является ресурсоемкой операцией.</p>
---	--

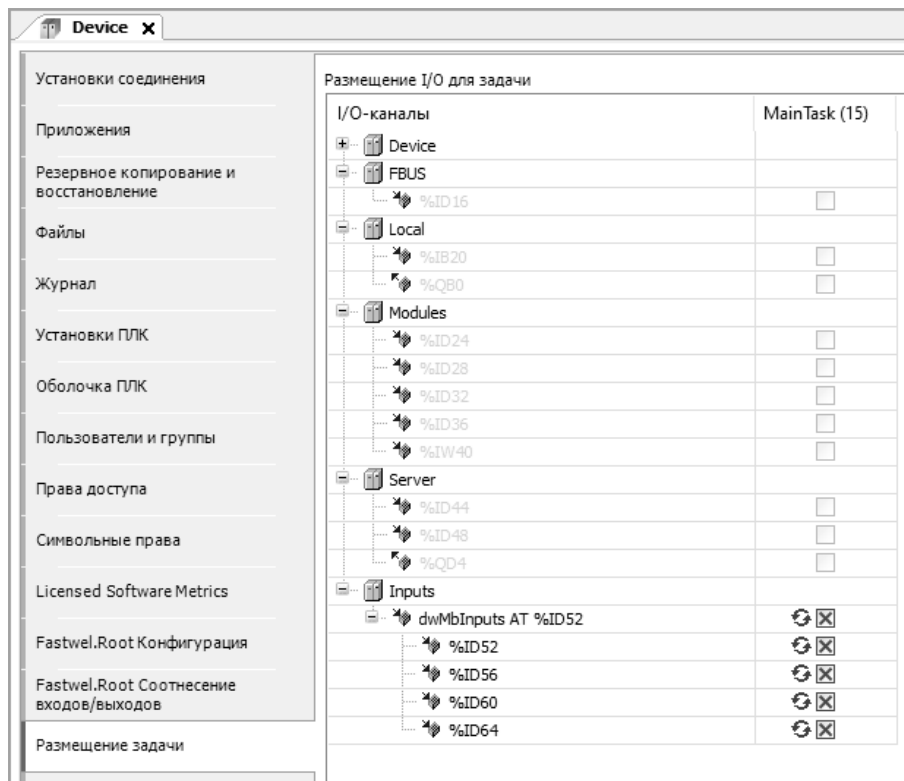


Рисунок 87 – Отображение информации о ссылках на входные и выходные данные в задачах приложения

3. Режим *Включено 2* (всегда в задаче цикла шины) (*Enabled 2 (always in bus cycle task)*) – на контроллерах Fastwel допускается использовать данный режим только при наличии в приложении единственной циклической или свободно-исполняемой задачи.



Рекомендуется никогда не устанавливать *Включено 2*, поскольку при последующих модификациях приложения может потребоваться использовать более одной задачи, и высока вероятность, что при внесении таких изменений необходимость изменения режима ввода-вывода может быть упущена из виду.

#### 4.3.4. Мониторинг значений переменных, соотнесенных с %I-адресами во входном образе процесса

IDE МЭК 61131-3 при подключении к контроллеру командой **Онлайн – Логин** обеспечивает возможность просмотра (мониторинга) текущих значений переменных приложения в области деклараций переменных и непосредственно в области представления текста программного кода.

При мониторинге переменных, соотнесенных с входными каналами внешних устройств, а также значений на каналах в редакторе образа процесса, отображаемом по команде контекстного меню **Изменить I/O-соотнесение** над описанием устройств в дереве проекта, следует учитывать следующие особенности, присущие системе исполнения приложений контроллеров Fastwel:

1. Если в контроллере выполняется приложение IDE МЭК 61131-3 и после подключения к контроллеру командой **Онлайн – Логин** в строке состояния среды разработки отображается статус **ЗАПУСК**, то значения переменных прямого представления (АТ %I), соотнесенных с входными каналами внешних устройств, обновляются с периодом не менее установленного параметром **Период выполнения сервисной задачи** на вкладке **Fastwel.Root.Конфигурация** в редакторе устройства (*CPMXXX-ZZ Programmable Automation Controller*), показанного на рисунке 88.

Таким образом, если система исполнения контроллера не остановлена на точке останова, то сервис мониторинга переменных среды разработки получает значения входных каналов из глобального сегмента входных данных. Данные входных каналов вводятся в глобальный сегмент только системной сервисной задачей, период выполнения которой определен параметром, показанным на рисунке 88.



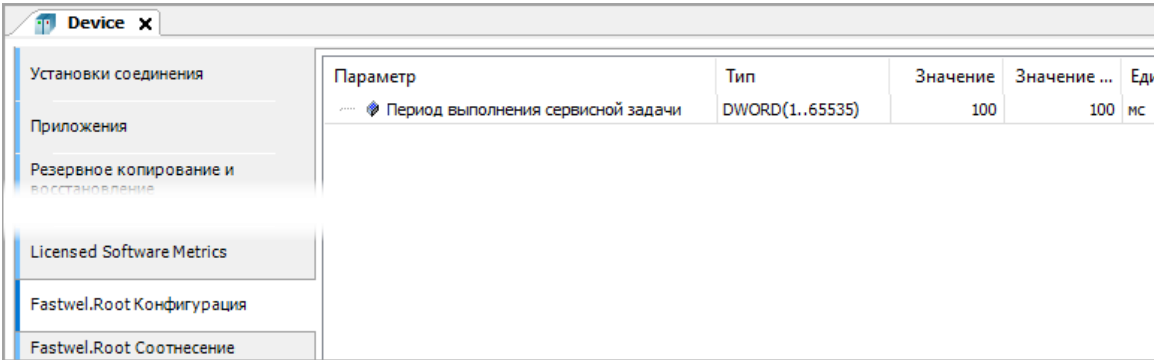


Рисунок 88 – Параметр "Период выполнения сервисной задачи"

- Если на вкладке **Установки ПЛК** задан режим обновления данных *Отключено* (обновление только в задаче), то, согласно п. 4.3.3, будет возможен мониторинг только входных и выходных каналов, на которые имеются явные ссылки в коде программных единиц приложения.
- Если в приложении имеется более одной задачи и хотя бы одна задача имеет явные ссылки на некоторый входной канал, то значения на данном канале, отображаемые при мониторинге в среде разработки со статусом **ЗАПУСК**, будут отличаться от значений этого же канала в персональных сегментах каждой задачи.

Пусть, например, в приложении установлен период сервисной задачи 5000 мс, и две программы, *PROG1* и *PROG2*, выполняющиеся по управлению циклических задач *Task1* и *Task2* с одинаковыми значениями периода 10 мс, имеют идентичный код, показанный на рисунке 89.

```

1  PROGRAM PROG1
2  VAR
3      act_direct : ACTIVITY_CHECKER;
4      dwDirectTxCount AT %ID56 : DWORD;
5  END_VAR

1  act_direct( ActivityCounter := dwDirectTxCount,
2              CheckPeriod := T#500MS);

```

Рисунок 89 – Исходный текст программы с явной ссылкой на входной канал по смещению 56

Предположим, что реальный период обновления значения на входном канале по адресу %ID56 составляет 10 мс.

После загрузки приложения в контроллер в области реализации каждой программы будут одновременно отображаться значения переменной блока *act\_direct.ActivityCounter* и переменной прямого представления *dwDirectTxCount*, как показано на рисунке 90.

Несмотря на то, что *dwDirectTxCount* присваивается переменной *act\_direct.ActivityCounter* в строке 1 при вызове экземпляра функционального блока, во время мониторинга видно, что переменные имеют разные значения, при этом значение *act\_direct.ActivityCounter* обновляется часто, а значение *dwDirectTxCount* обновляется один раз в 5 секунд.

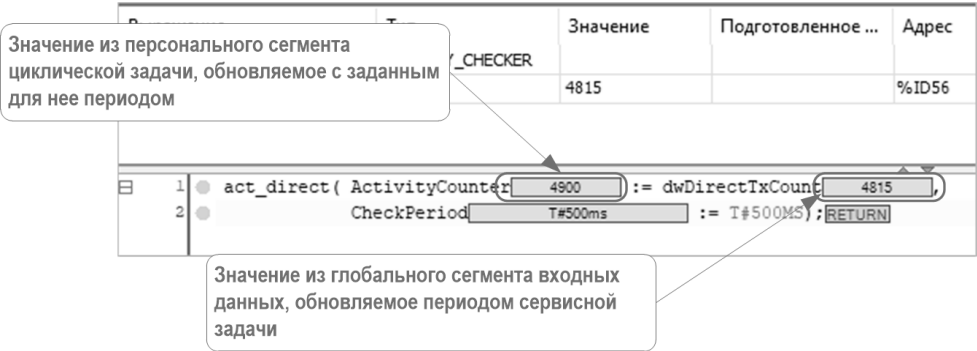
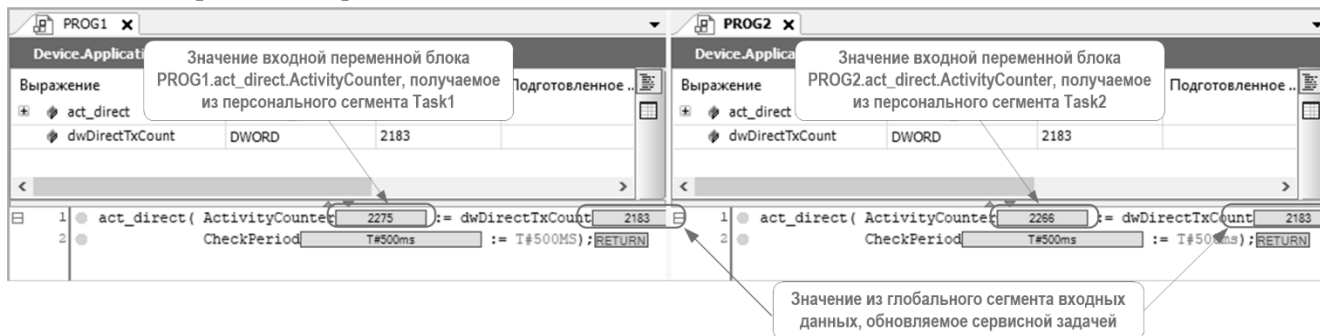


Рисунок 90 – Мониторинг переменной прямого представления в персональном сегменте задачи и глобальном входном образе процесса

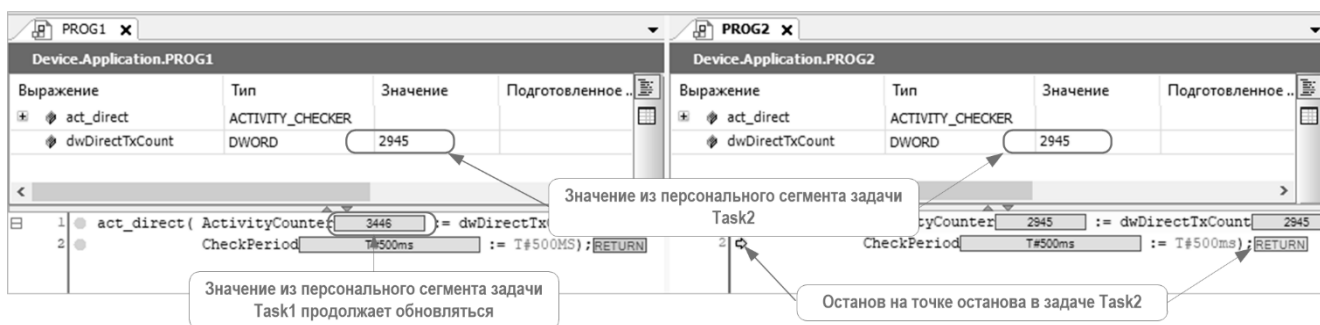
Если период задачи *Task2* отличается от *Task1* и составляет, скажем, 500 мс, то переменные *PROG1.act\_direct.ActivityCounter* и *PROG2.act\_direct.ActivityCounter* при мониторинге будут иметь разные значения, как показано на рисунке 91, и обновляться с разными периодами.



**Рисунок 91 – Мониторинг переменной прямого представления в персональных сегментах двух разночастотных задач и в глобальном входном образе процесса**

- Если одна из нескольких задач приложения, имеющая явную ссылку на некоторый входной канал, остановлена на точке останова, то значения на данном канале, отображаемые при мониторинге в IDE МЭК 61131-3 со статусом **HALT ON BP**, будут получены средой разработки из персонального входного сегмента остановленной задачи.

Если в примере, рассмотренном на рисунках 89 – 91, остановить задачу *Task2* на точке останова, установленную на выходе из *PROG2*, то отображаемое во время мониторинга значение на канале %ID56 будет считываться IDE МЭК 61131-3 из персонального сегмента задачи *Task2*, а значение этого же канала в персональном сегменте задачи *Task1* будет продолжать обновляться, как показано на рисунке 92.



**Рисунок 92 – Мониторинг переменной прямого представления в персональных сегментах двух задач и в глобальном входном образе процесса при останове одной из задач на точке останова**

#### 4.4. Выполнение задач приложения

##### 4.4.1. Общие сведения

Код приложения IDE МЭК 61131-3 выполняется под управлением задач, добавляемых в конфигурацию приложения в список *Конфигурация задач* в дереве проекта.

Для каждой задачи устанавливается приоритет, тип задачи, задаются параметры, специфические для выбранного типа задачи, настраивается сторожевой таймер задачи и определяется список программных единиц типа PROGRAM, которые будут вызываться последовательно друг за другом при очередном наступлении условий запуска задачи.

Информация о приоритете и сторожевом таймере задач всех типов приведена в п. 4.2.4 настоящего документа.

Система исполнения приложений МЭК 61131-3 контроллеров Fastwel поддерживает четыре типа задач:

- Циклические.
- Ациклические по событию (тип *Событие*).
- Ациклические по состоянию (тип *Состояние*).

## 4. Свободно-исполняемые.

Настоящий подраздел содержит информацию о принципе работы задач разных типов и рекомендации по выбору типа задачи и настройке параметров задач.

При принятии решения об использовании в приложении более одной задачи рекомендуется придерживаться следующих общих правил:

1. Если нет ясного понимания принципов работы многозадачных приложений или отсутствует техническое обоснование для использования более одной задачи в приложении, рекомендуется оставить в приложении одну задачу.
2. Если требуется использовать в приложении более одной задачи, не следует вызывать одни и те же программные единицы типа *PROGRAM* (программа) из разных задач.
3. В алгоритмах, вызываемых из разных задач, необходимо избегать доступа к одним и тем же глобальным переменным. Даже если для задач установлено одинаковое значение приоритета, впоследствии это может измениться, а информация о наличии доступа к общим глобальным переменным из разных задач может быть забыта.
4. Запрещается формировать в разных задачах значения битовых выходных переменных прямого представления, расположенных в смежных битовых позициях выходного образа процесса. Это же правило относится к глобальным переменным, соотнесенным с каналами в выходном образе процесса способом 2 по п. 4.3.1.
5. В алгоритмах, вызываемых из задач с разными приоритетами, не следует использовать одни и те же глобальные переменные, соотнесенные с каналами во входном образе процесса способом 2 по п. 4.3.1.

Глобальная переменная размещается компилятором в сегменте внутренних переменных приложения, и, в случае соотнесения ее с входным каналом (с адресом во входном образе процесса), компилятор создает *явные ссылки* (см. 4.3.3) на этот канал во всех задачах, использующих соотнесенную с каналом глобальную переменную.


Кроме того, компилятор генерирует скрытый код копирования значения из персонального сегмента каждой задачи в глобальную переменную после ввода данных в персональный сегмент. По этой причине в некоторых случаях после соотнесения глобальных переменных с образом процесса в менеджере библиотек оказывается подключенной системная библиотека SysMem.

После загрузки приложения в контроллер обновление глобальной переменной, соотнесенной с каналом во входном образе процесса, происходит следующим образом: некоторая задача, использующая глобальную переменную, в начале очередного цикла читает значение канала из окружения в свой персональный сегмент согласно алгоритму, описанному в п. 4.3.2, а затем вызывает сгенерированный скрытый код, копирующий прочитанное значение на канале из своего персонального сегмента в глобальную переменную.

Таким образом, алгоритм менее приоритетной задачи, ссылающийся на эту же глобальную переменную может работать неправильно в двух случаях:

- 1) если более приоритетная задача прервала (вытеснила) менее приоритетную в момент копирования менее приоритетной задачей значения канала (битового или имеющего размер более 4 байт) из своего персонального сегмента в глобальную переменную, после чего обновила глобальную переменную значением из своего персонального сегмента, в результате чего, после возобновления прерванной задачи, значение глобальной переменной может быть испорчено, поскольку возобновленная задача продолжит копирование и скопирует часть значения из своего персонального сегмента;
- 2) если более приоритетная задача прервала (вытеснила) менее приоритетную, когда та выполняла операцию, расположенную в коде между двумя другими операциями, использующими глобальную переменную, в результате чего после возобновления прерванной задачи значение глобальной переменной будет отличаться от значения этой же переменной до прерывания, что приведет к нарушению принципа неизменности входных данных в пределах цикла, на котором основана программная модель ГОСТ Р МЭК 61131-3, и, с высокой вероятностью, это приведет к неправильной работе алгоритма, управляемого менее приоритетной задачей.

6. Для обмена данными между задачами рекомендуется использовать механизм связывания задач по данным, предоставляемый функцией `F_IecTasks_linkVariables` из библиотеки `FastwelCore` (см. п. 6.4.3.8).
7. Если все-таки требуется читать и записывать значения одних и тех же глобальных переменных из разных задач, для синхронизации доступа рекомендуется использовать функции атомарного доступа из системной библиотеки `SysCpuHandling`.

	<p>Запрещено использовать семафоры из библиотеки <code>SysSem</code>.</p>
---	---

8. Не следует назначать приоритет от 0 до 15 задачам, которые управляют продолжительными по времени алгоритмами. В наибольшей степени это правило относится к свободно-исполняемым задачам, а также к задачам, в которых осуществляется чтение или запись файлов на дисковые накопители контроллера.
9. Нежелательно назначать приоритет от 0 до 7 более чем одной задаче, при этом меньшее значение приоритета должно устанавливаться для наиболее короткой по времени выполнения задачи с наименьшим периодом вызова.
10. При необходимости контроля выполнения некоторой задачи сторожевым таймером при наличии в приложении нескольких других более приоритетных задач следует устанавливать значение интервала сторожевого таймера с 5 – 10-кратным запасом от максимальной расчетной длительности выполнения алгоритма, управляемого данной задачей.

#### 4.4.2. Циклические задачи

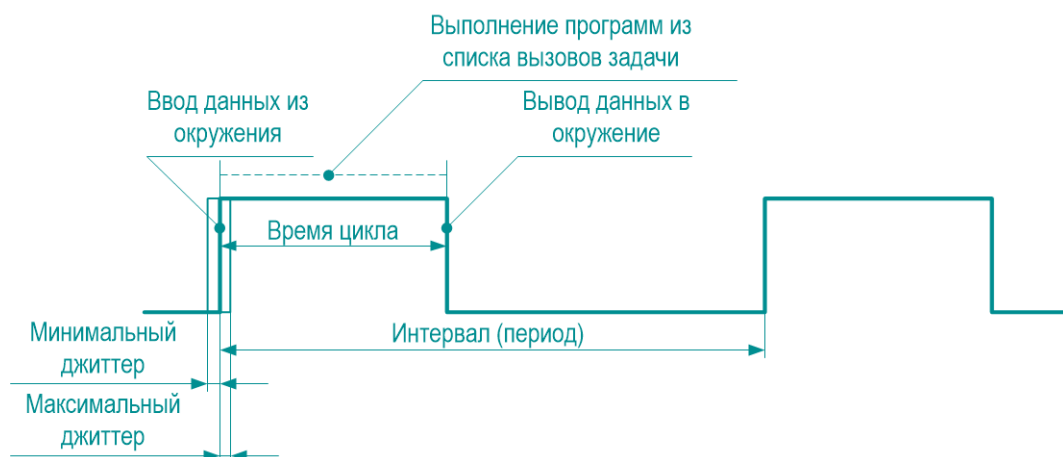


Рисунок 93 – Циклограмма выполнения циклической задачи

Циклограмма функционирования циклической задачи представлена на рисунке 93.

Циклическая задача вызывается с периодом, определяемым параметром **Интервал** в редакторе конфигурации задачи.

При соединении среды разработки с контроллером редактор **Конфигурация задач** имеет вложенную вкладку **Мониторинг**, позволяющую просматривать статистику выполнения задач приложения, как показано на рисунке 94.

Конфигурация задач												
Мониторинг	Использование переменной		Системные события		Свойства							
Задача	Статус	Счётчик МЭК-...	Счётчик циклов	За...	Посл. (µs)	Сред. время цикла (µs)	Макс. время цик...	Мин. время ц...	Джитт...	Мин. джитте...	Макс. джит...	
ClientTask	Valid	16710	16718	10 ms	115	95	720	28	-72	-657	665	
MainTask	Valid	16710	16718	10 ms	114	121	828	32	-267	-916	863	



Рисунок 94 – Просмотр статистики выполнения задач приложения

Столбец **Счетчик МЭК-циклов** содержит количество циклов задачи, в которых выполнялся код программных единиц из списка вызовов данной задачи, а столбец **Счетчик циклов** – общее количество циклов задачи. Если приложение останавливалось или подвергалось отладке, данные счетчики будут иметь разные значения.

Столбцы **Посл.(us)**, **Сред. время цикла (us)**, **Макс. время цикла (us)** и **Мин. время цикла (us)** содержат текущее, среднее, максимальное и минимальное значения времени выполнения кода программных единиц из списка вызовов задачи в микросекундах за цикл.

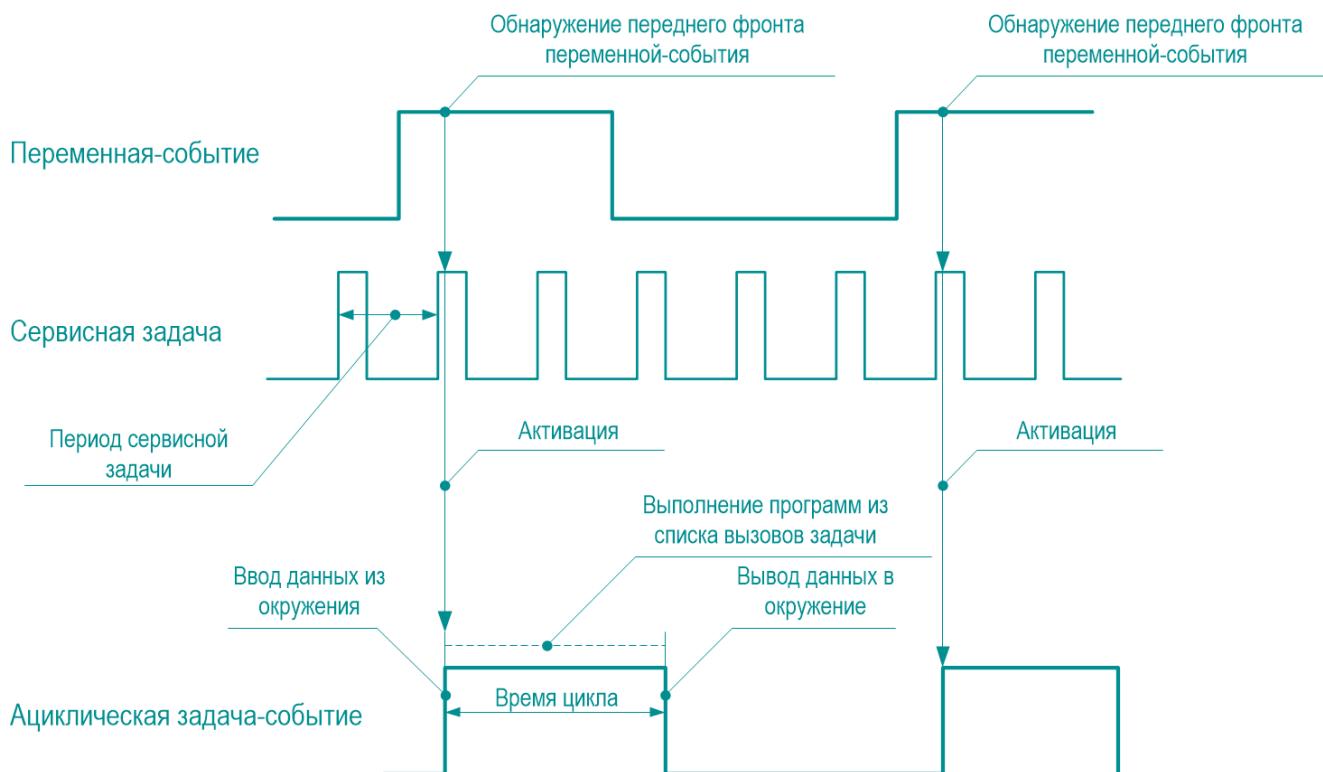
Столбцы **Джиттер (us)**, **Мин. джиттер (us)** и **Макс. джиттер (us)** содержат текущее, минимальное и максимальное значения времени (в микросекундах) запаздывания или опережения начала очередного цикла выполнения кода программных единиц из списка выполнения задачи относительно момента времени, запланированного на предыдущем цикле. Положительное значение джиттера соответствует запаздыванию, а отрицательное – опережению. Опережение возникает в ситуации, когда требуется выровнять временную сетку циклов задачи, если на предыдущем цикле произошло запаздывание.

Для сброса статистики для какой-либо задачи выберите задачу в списке мониторинга, щелкните правой кнопкой мыши и выберите команду **Reset** в контекстном меню. Сбор статистики для выбранной задачи начнется заново

	<p>При запаздывании запуска очередного цикла (положительное значение в столбце <b>Макс. джиттер</b>) планировщик задач системы исполнения скомпенсирует это запаздывание, начав следующий цикл ранее ожидаемого момента времени, равного сумме текущего момента с периодом (интервалом) задачи. При этом интервал опережения будет равен времени запаздывания на предыдущем цикле.</p> <p>Таким образом обеспечивается поддержание запуска циклов задачи в рамках фиксированной временной сетки и устраняется накопление фазовой ошибки.</p>
	<p>Уровень приоритета циклических задач должен устанавливаться на основе уровня важности и срочности алгоритмов, которые должны выполняться циклически под управлением некоторой задачи – <u>более важным, срочным и коротким алгоритмам</u> должно соответствовать меньшее значение приоритета, задаваемое в поле <b>Приоритет</b> конфигурации задачи.</p>



#### 4.4.3. Ациклические задачи типа Событие

Ациклическая задача типа *Событие* начинает выполнение по переднему фронту (при переходе из FALSE в TRUE) булевой *переменной чувствительности*, заданной в поле **Событие** в редакторе конфигурации задачи.



**Рисунок 95. Циклограмма выполнения ациклической задачи, вызываемой по фронту переменной-события**

Циклограмма выполнения ациклической задачи типа *Событие* показана на рисунке 95.

	<p>Ациклические задачи типа <i>Событие</i> следует использовать для управления короткими по времени алгоритмами однократного действия, которые должны выполняться только в момент наступления некоторого условия, олицетворяющего событие в контролируемом технологическом процессе.</p> <p>Ациклическим задачам типа <i>Событие</i> рекомендуется устанавливать более высокий приоритет (меньшее абсолютное значение в поле <b>Приоритет</b> в конфигурации задачи) среди задач других типов, имеющих приоритет реального времени (абсолютные значения от 0 до 15).</p>
	<p>Обнаружение фронта переменной чувствительности производится сервисной задачей системы исполнения, период которой определяется параметром <b>Период выполнения сервисной задачи</b> на вкладке <b>Fastwel.Root.Конфигурация</b> в редакторе устройства (<i>CPMXXX-ZZ Programmable Automation Controller</i>). В связи с этим при необходимости реагировать на часто возникающие события в ациклических задачах типа <i>Событие</i> следует соответствующим образом уменьшить период сервисной задачи.</p>

При использовании входных каналов модулей дискретного ввода в качестве источников данных для переменных чувствительности ациклических задач типа *Событие* необходимо убедиться в отсутствии дребезга дискретных входных сигналов, который может быть причиной ложных срабатываний алгоритмов, управляемых ациклическими задачами, и, при наличии дребезга, устранить его программным способом или внесением изменений в схему подключения внешних цепей или организации полевого электропитания.

#### 4.4.4. Ациклические задачи типа Статус

Ациклическая задача типа *Статус* становится активной при равенстве TRUE булевой *переменной чувствительности*, заданной в поле **Событие** в редакторе конфигурации задачи.

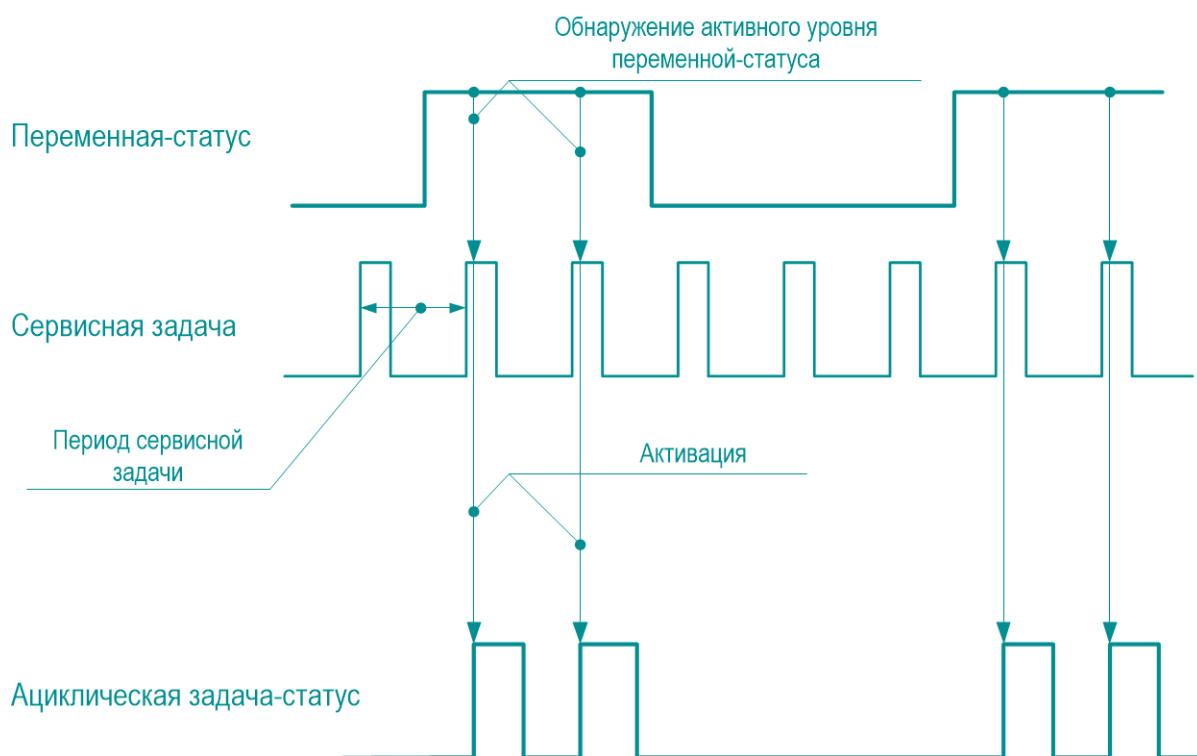


Рисунок 96 – Циклограмма выполнения ациклической задачи, вызываемой по уровню TRUE статусной переменной

Циклограмма выполнения ациклической задачи типа *Статус* показана на рисунке 96.

При использовании входных каналов модулей дискретного ввода в качестве источников данных для переменных чувствительности ациклических задач типа *Статус* необходимо убедиться в отсутствии дребезга, ложных срабатываний или пропаданий дискретных входных сигналов, что может стать

причиной ложных срабатываний или отсутствия срабатывания алгоритмов, управляемых ациклическими задачами типа *Статус*.

	<p>Ациклические задачи типа <i>Статус</i> могут использоваться для управления короткими по времени циклическими алгоритмами, которые должны выполняться только при сохранении некоторого условия, олицетворяющего состояние контролируемого технологического процесса или технических средств системы.</p> <p>Уровень приоритета ациклических задач типа <i>Статус</i> должен устанавливаться на основе уровня важности и срочности действий, которые должны начать выполняться при нахождении контролируемого процесса в некотором состоянии, олицетворяемом переменной чувствительности – более важным и срочным действиям должно соответствовать меньшее значение приоритета, задаваемое в поле <b>Приоритет</b> конфигурации задачи. Однако это справедливо только для коротких по времени последовательностей действий.</p>
	<p>Проверка текущего значения переменной чувствительности и активация ациклической задачи типа <i>Статус</i> производится сервисной задачей системы исполнения, период которой определяется параметром <b>Период выполнения сервисной задачи</b> на вкладке <b>Fastwel.Root.Конфигурация</b> в редакторе устройства (<i>CPMXXX-ZZ Programmable Automation Controller</i>).</p> <p>В связи с этим при необходимости обеспечения более частых вызовов задачи типа <i>Статус</i> во время равенства TRUE переменной чувствительности следует соответствующим образом уменьшить период сервисной задачи.</p>

#### 4.4.5. Свободно-исполняемые задачи

Задачи типа *Свободное выполнение* выполняются циклически, но не имеют заданного периода исполнения. Можно считать, что свободно исполняемые задачи теоретически имеют нулевой период исполнения, т.е. выполнение следующего цикла начинается сразу после завершения текущего цикла.

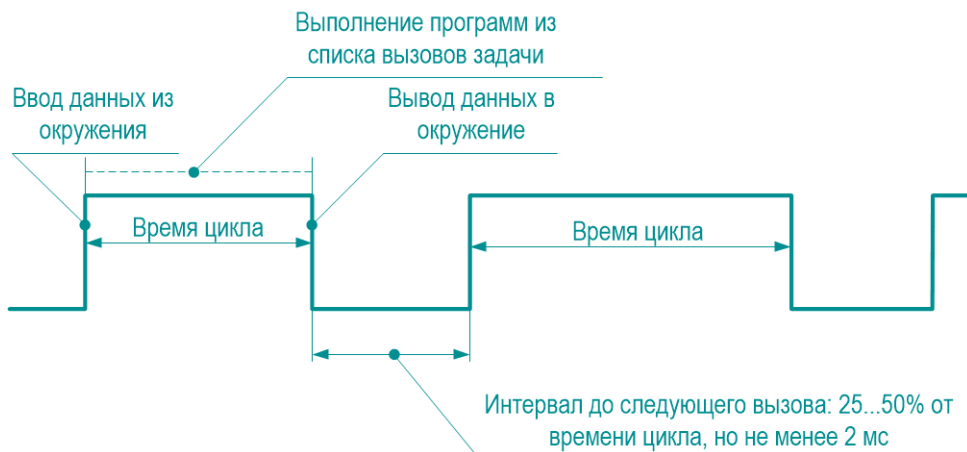




Рисунок 97 – Циклограмма выполнения свободно-исполняемой задачи

Циклограмма выполнения свободно-исполняемой задачи показана на рисунке 97.

Практически интервал между вызовами составляет от 25 до 50% времени очередного цикла, и не может составлять менее 2 мс.

	<p>Свободно-исполняемые задачи рекомендуется использовать для выполнения длительных операций, таких как сложные ресурсоемкие вычисления, чтение и запись данных в файлы или обмен данными через сетевые сокет, проверка статуса батареи резервного питания, обращения к системным часам и т.п.</p>
---	--



	<p>Для свободно-исполняемых задач запрещается устанавливать абсолютное значение приоритета, меньшее 16.</p> <p>Нарушение этого правила может привести к переходу контроллера в безопасный режим по сторожевому таймеру контроля выполнения других задач приложения, к невозможности функционирования сетевых сервисов контроллера, а также к невозможности установить связь между средой разработки и контроллером.</p> <p>В такой ситуации восстановление работоспособности контроллера возможно только путем его перезапуска в режиме с исходными (заводскими) настройками и последующей полной загрузкой обновлённого приложения, в котором исправлено назначение приоритетов.</p>
---	---

## 4.5. Обработка системных событий

### 4.5.1. Общие сведения

Системное событие представляет собой вызов функции, выполняемый системой исполнения при изменении состояния системы исполнения или при выполнении некоторых системных операций.

Обработка системных событий может потребоваться в следующих случаях:

1. При реализации развитых алгоритмов в приложениях и библиотеках, требующих доступа к файлам, сокетам и другим системным ресурсам. В таких случаях часто требуется самостоятельное освобождение и повторный доступ к указанным системным ресурсам.
2. При необходимости учета в алгоритме приложения возможности его полной или частичной модификации методом полной загрузки или загрузкой онлайн-изменений, при которых некоторые переменные алгоритма должны получать значения, отличные от устанавливаемых или сохраняемых по умолчанию системой исполнения или компилятором.
3. При необходимости самостоятельного управления состоянием приложения и его составных частей. Например, если при полной загрузке обновленного приложения в контроллер или после сброса командой **Онлайн – Сброс** требуется автоматически начать работу приложения, сделать это можно только в обработчике системного события.
4. Функции некоторых системных библиотек могут вызываться только при обработке определенных системных событий. Например, функция `F_IecTasks_linkVariables` (см. п. 6.4.3.8) из библиотеки `FastwelCore` должна вызываться только из обработчика системного события *LegacyOnInit*.

Система исполнения приложений контроллеров `Fastwel` поддерживает большинство событий из набора IDE МЭК 61131-3, а также содержит ряд собственных системных событий, имена которых начинаются с префикса *Legacy*. События с префиксом *Legacy* введены в систему для облегчения миграции приложений `CoDeSys 2.3` для контроллеров `Fastwel I/O` на контроллеры `Fastwel` с системой исполнения IDE МЭК 61131-3.

Установка функций обработки событий в пользовательском приложении выполняется в редакторе **Конфигурация задач (Task Configuration)** на вкладке **Системные события**, показанной на рисунке 98.

Для добавления обработчика системного события следует:

1. На вкладке **Системные события** нажать кнопку **Добавить обработчик событий (Add Event Handler)**. На экран монитора будет выведена диалоговая панель **Новый обработчик события (Add Event Handler)**, показанная на рисунке 99.
2. Выбрать тип события, которое требуется обрабатывать, в выпадающем списке **Событие (Event)**.
3. Ввести имя создаваемой функции обработки события в поле **Вызываемая функция (Function to call)**.



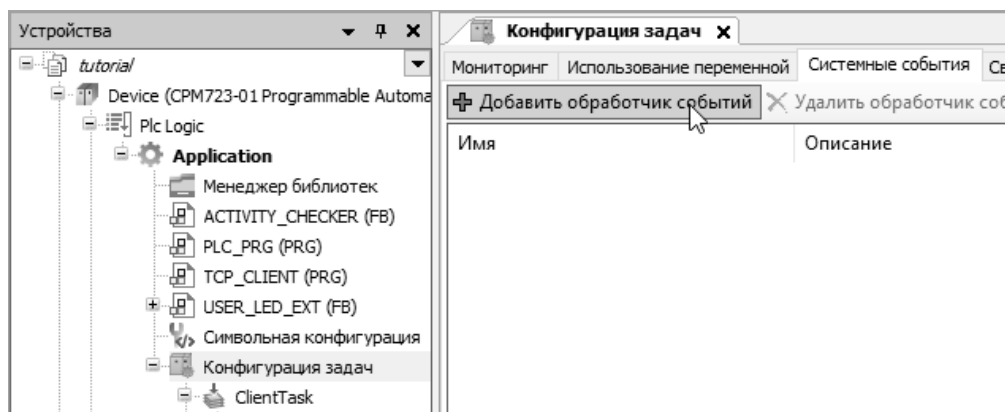


Рисунок 98 – Добавление обработчика системного события

4. Выбрать область действия создаваемой функции переключателем **Область действия (Scope)**. При выборе области действия **Приложение (Application)** функция обработки системного события будет создана в текущем выбранном приложении.

Если выбрать область действия **POU**, то функция будет создана в области действия всего проекта на вкладке **POU**, в результате чего будет возможность использовать создаваемую функцию обработки события в приложениях для всех контроллеров, входящих в проект.

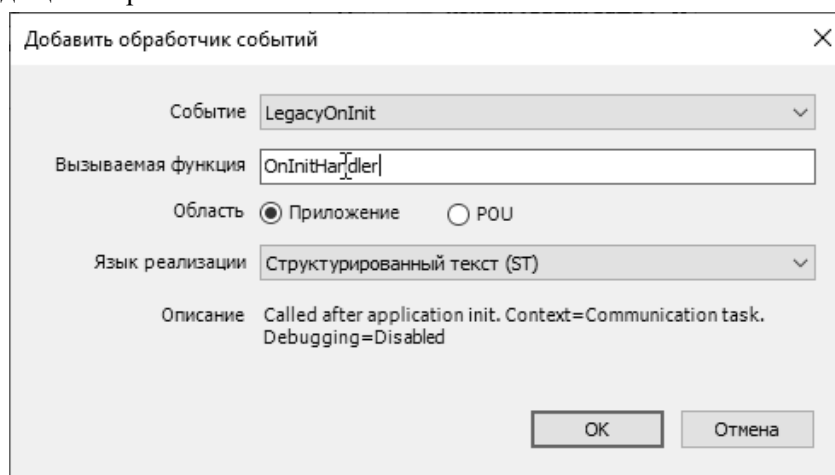


Рисунок 99 – Диалоговая панель создания функции обработки системного события

Поле **Описание (Description)** содержит описательную информацию о системном событии, при возникновении которого произойдет вызов функции, а также признак возможности пошаговой отладки с использованием точек останова в исходном тексте функции: *Debugging=Enabled*, если отладка возможна, и *Debugging=Disabled* – при отсутствии возможности отладки.

Поле описания *Context* содержит тип задачи, которая будет вызывать функцию обработки: *IEC Task* – задача приложения, созданная пользователем; *Communication task* – коммуникационная задача, обслуживающая информационный обмен со средой разработки и другими клиентами протокола CODESYS V3, или сервисная задача.

5. После нажатия **ОК** в диалоговой панели **Добавить обработчик событий** функция обработки события появится в списке обрабатываемых событий на вкладке **Системные события**, а также в списке программных единиц приложения, если при создании обработчика была выбрана область действия **Приложение**, либо в списке программных единиц на вкладке **POU**.

Если во время работы приложения установить соединение между средой разработки и контроллером командой **Онлайн – Логин**, то в списке событий на вкладке **Системные события** в столбце **Количество вызовов** будет отображено количество вызовов установленных обработчиков системных событий, как показано на рисунке 100. При этом кнопка **Онлайн-сброс** позволяет сбросить в 0 количество вызовов обработчиков для перезапуска сбора статистики.

Конфигурация задач							
Мониторинг		Использование переменных		Системные события		Свойства	
Добавить обработчик событий		Удалить обработчик событий		Информация события...		Открыть функцию события	
				Онлайн-сброс			
Имя	Описание	Контекст	Отладка	Вызываемая функция	Активный	Сост...	Количество вы...
LegacyOnInit	Called after application init	Communication task	✗	OnInitHandler	☑	0	1
AfterReadingInputs	Called after reading inputs	IEC task	☑	AfterReadingInputsHandler	☑	0	160954
BeforeReadingInputs	Called before reading inputs	IEC task	☑	BeforeReadingInputsHandler	☑	0	160955
AfterWritingOutputs	Called after writing outputs	IEC task	☑	AfterWritingOutputsHandler	☑	0	160955
BeforeWritingOutputs	Called before writing outputs	IEC task	☑	BeforeWritingOutputsHandler	☑	0	160955
CodeInitDone	Event is sent after CodeInit...	Communication task	✗	CodeInitDoneHandler	☑	0	0
ConfigAppStartedDone	Event is sent after the confi...			ConfigAppStartedDoneHan...	☑	0	1
ConfigAppStoppedDone	Event is sent after the confi...			ConfigAppStoppedDoneHan...	☑	0	0
DebugLoop	Event is sent cyclically in the...	IEC task	✗	DebugLoopHandler	☑	0	0
DownloadDone	Called after application onlin...	Communication task	✗	DownloadDoneHandler	☑	0	1
Exception	Event is sent, if an exceptio...	Exception task or ...	Depends on the ...	ExceptionRaisedHandler	☑	0	0

Рисунок 100 – Список событий при наличии соединения среды разработки с контроллером

Примеры применения обработки системных событий, поставляемых в Fastwel PLC Application Toolkit, в таблице 4 отмечены символом <sup>Е</sup>.

#### 4.5.2. Перечень поддерживаемых системных событий

Перечень стандартных системных событий IDE МЭК 61131-3, поддерживаемых системой исполнения контроллеров Fastwel, представлен в таблице 5.

Таблица 5 – Поддерживаемые стандартные системные события IDE МЭК 61131-3

Тип события	Контекст вызова	Описание
<i>PrepareStart</i>	Коммуникационная задача	Вызывается перед запуском приложения
<i>StartDone</i>	Коммуникационная задача	Вызывается после запуска приложения
<i>ConfigAppStartedDone</i>	Коммуникационная задача	Вызывается после <i>StartDone</i>
<i>PrepareStop</i>	Коммуникационная задача	Вызывается перед остановом приложения
<i>PrepareConfigAppStopped</i>	Коммуникационная задача	Вызывается после <i>PrepareStop</i>
<i>StopDone</i>	Коммуникационная задача	Вызывается после останова приложения
<i>ConfigAppStoppedDone</i>	Коммуникационная задача	Вызывается после <i>StopDone</i>
<i>PrepareReset</i>	Коммуникационная задача	Вызывается перед сбросом приложения
<i>ResetDone</i>	Коммуникационная задача	Вызывается после сброса приложения
<i>PrepareOnlineChange</i>	Коммуникационная задача	Вызывается перед онлайн-изменением приложения
<i>OnlineChangeDone</i>	Коммуникационная задача	Вызывается после онлайн-изменения приложения
<i>PrepareDownload</i>	Коммуникационная задача	Вызывается перед загрузкой приложения
<i>UpdateConfigurationDone</i>	Коммуникационная задача	Вызывается после обновления конфигурации
<i>DownloadDone</i>	Коммуникационная задача	Вызывается после загрузки приложения
<i>PrepareUpdateConfiguration</i>	Коммуникационная задача	Вызывается перед обновлением конфигурации
<i>PrepareExit</i>	Коммуникационная задача	Вызывается перед завершением работы приложения
<i>CodeInitDone</i>	Коммуникационная задача	Вызывается после инициализации сегмента кода в процессе онлайн-изменения приложения.
<i>Exception</i>	Специальная системная задача обработки исключений	Вызывается при возникновении исключения в коде приложения
<i>BeforeReadingInputs</i>	Пользовательская задача приложения	Вызывается перед чтением в персональный сегмент входных данных задачи приложения
<i>AfterReadingInputs</i>		Вызывается после чтения в персональный сегмент входных данных задачи приложения
<i>BeforeWritingOutputs</i>		Вызывается перед выводом данных, относящихся к задаче, из глобального выходного образа процесса в окружение.
<i>AfterWritingOutputs</i>		Вызывается после вывода данных, относящихся к задаче, из глобального выходного образа процесса в окружение.
<i>DebugLoop</i>	Коммуникационная задача	Вызывается циклически при останове пользовательской задачи на точке останова.

Перечень реализованных дополнительных системных событий перечислен в таблице 6.

Таблица 6 – Дополнительные системные события

Тип события	Контекст вызова	Описание
<i>LegacyBeforeReset</i>	Коммуникационная задача	Вызывается перед сбросом приложения по команде <b>Онлайн – Сброс</b> .
<i>LegacyOnDownload</i>	Сервисная задача	Вызывается перед началом загрузки приложения и перед онлайн-изменением приложения.
<i>LegacyOnInit</i>	Коммуникационная задача	Вызывается после <i>StartDone</i> . Данные и код приложения готовы к работе.
<i>LegacyOnLogin</i>	Сервисная задача	Вызывается при авторизации удаленного приложения в процессе установления соединения с системой исполнения.
<i>LegacyOnLogout</i>	Сервисная задача	Вызывается при завершении сеанса связи удаленного приложения с системой исполнения.
<i>LegacyOnPowerOn</i>	Сервисная задача	Вызывается при запуске контроллера после полного перезапуска или при включении питания.
<i>LegacyOnProgramChange</i>	Коммуникационная задача	Вызывается перед онлайн-изменением приложения.
<i>LegacyOnStart</i>	Сервисная задача	Вызывается при запуске приложения по команде <b>Отладка – Старт IDE МЭК 61131-3</b> .
<i>LegacyOnStop</i>	Сервисная задача	Вызывается после останова приложения по команде <b>Отладка – Стоп IDE МЭК 61131-3</b> .
<i>LegacyOnTimer</i>	Сервисная задача	Вызывается циклически с периодом, установленным параметром <b>Fastwel.Root Конфигурация – Период выполнения сервисной задачи</b> .

#### 4.5.3. Порядок вызова обработчиков системных событий

Обработчики системных событий разных типов вызываются в контексте коммуникационных задач, обслуживающих взаимодействие среды разработки и других приложений с системой исполнения контроллера, в контексте сервисной задачи системы исполнения и в контексте специальной задачи обработки исключений в приложении пользователя.

При запуске контроллера после включения питания или полного перезапуска используется следующий порядок вызова обработчиков системных событий:

```

PrepareUpdateConfiguration
UpdateConfigurationDone
LegacyOnDownload
LegacyOnPowerOn
DownloadDone
LegacyOnInit
PrepareStart
StartDone
ConfigAppStartedDone
LegacyOnStart

```

При сбросе приложения командой **Онлайн – Сброс** используется следующий порядок вызова обработчиков системных событий:

```

PrepareStop
PrepareConfigAppStopped
StopDone
ConfigAppStoppedDone
PrepareReset
LegacyBeforeReset
PrepareExit
PrepareUpdateConfiguration
UpdateConfigurationDone
...
PrepareUpdateConfiguration
UpdateConfigurationDone
ResetDone

```

При полной загрузке приложения из среды разработки используется следующий порядок вызова обработчиков системных событий:

```

LegacyOnLogin
PrepareDownload

```

```

PrepareExit
PrepareStop
PrepareConfigAppStopped
StopDone
ConfigAppStoppedDone
LegacyOnStop
PrepareUpdateConfiguration
UpdateConfigurationDone
...
PrepareUpdateConfiguration
UpdateConfigurationDone
DownloadDone
LegacyOnInit
PrepareStart
StartDone
ConfigAppStartedDone

```

При обновлении приложения из среды разработки методом онлайн-изменения используется следующий порядок вызова обработчиков системных событий:

```

LegacyOnLogin
PrepareOnlineChange
LegacyOnDownload
LegacyOnProgramChange
CodeInitDone
OnlineChangeDone
LegacyOnInit

```



Обработчики событий *PrepareUpdateConfiguration* и *UpdateConfigurationDone* вызываются по одному разу на каждую подсистему ввода-вывода, чья конфигурация имеется в загруженном приложении контроллера.

## 4.6. Загрузка приложения в контроллер

### 4.6.1. Способы загрузки приложения в контроллер

Среда разработки IDE МЭК 61131-3 загружает в контроллер исполняемый бинарный код процессора, на базе которого реализовано главное вычислительное устройство контроллера, а также бинарную конфигурацию устройств и сетевых протоколов, описания которых созданы пользователем в проекте.

В главном меню **Компиляция (Build)** IDE МЭК 61131-3 имеется команда **Генерировать код (Generate code)**, которая выполняет компиляцию, а затем генерирует исполняемый код целевого процессора и другие бинарные данные приложения, готовые к загрузке в контроллер.

Данная операция выполняется автоматически в самом начале загрузки приложения в контроллер при выполнении команды **Онлайн (Online) – Логин (Login)**. После успешной загрузки приложения в контроллер, в каталоге размещения файла проекта генерируются служебные файлы с информацией о трансляции, имеющие расширения *compileinfo* и *bootinfo*.

После внесения изменений в исходный текст программных единиц проекта транслятор среды разработки предпримет попытку сгенерировать и загрузить в контроллер бинарный код только измененных программных единиц без останова выполнения приложения (**Online Change** или **Логин с онлайн-изменением**).



Если при фактическом отсутствии ошибок во время работы над проектом в панели сообщений среды разработки появились сообщения об ошибках, можно выполнить команду **Компиляция (Build) – Очистить все (Clean All)**, а затем повторить команду **Генерировать код**.

Команда очистки служит для удаления ранее сформированной средой разработки информации о компиляции и загрузке приложения в контроллер с тем, чтобы иметь возможность сгенерировать код приложения "с нуля".



После выполнения команды **Очистить** загрузка изменений в контроллер (**Логин с онлайн-изменением**) становится невозможной, и приложение может быть передано в контроллер только путем полной загрузки.

Отличие команды **Очистить все** от **Очистить** состоит в том, что **Очистить все** удаляет ранее сформированную информацию о компиляции для всех приложений в данном проекте, а команда **Очистить** – только для *активного* приложения.

Большинство операций в главном меню среды разработки выполняется в отношении текущего *активного* приложения. Проект IDE МЭК 61131-3 может содержать приложения для нескольких контроллеров, и в этом случае для трансляции необходимо выбрать требуемое приложение в качестве активного, как показано на рисунке 101.

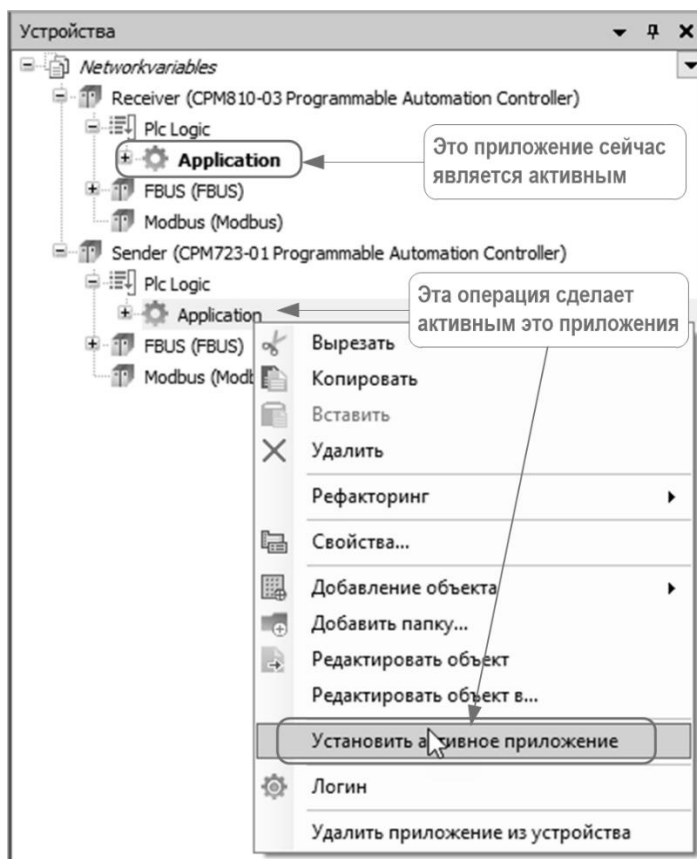


Рисунок 101 – Выбор активного приложения в проекте с приложениями для двух контроллеров

Приложение может быть загружено в контроллер из среды разработки следующими основными способами:

1. Метод полной загрузки – загрузка из среды разработки по команде **Онлайн – Логин**, если в контроллере нет загруженного приложения или приложение в контроллере значительно отличается от приложения в проекте, открытом в IDE МЭК 61131-3.
2. Метод загрузки онлайн-изменением – загрузка из среды разработки по команде **Онлайн – Логин**, если в IDE МЭК 61131-3 открыт проект с приложением, которое ранее было загружено в контроллер, и приложение в среде разработки отличается от ранее загруженного в контроллер исходным текстом отдельных программных единиц, декларациями переменных, версиями используемых библиотек, профиля визуализации и другими элементами проектной информации.
3. Метод множественной загрузки – загрузка из среды разработки нескольких приложений, входящих в один проект, в несколько контроллеров по команде **Онлайн – Множественная загрузка**.

Настоящий подраздел содержит сведения об особенностях использования перечисленных способов загрузки приложения в контроллер. Указания по развертыванию приложений вместе с конфигурацией системных сервисов на нескольких контроллерах приведены в п. 5.5 настоящего документа.

#### 4.6.2. Полная загрузка приложения

Перед загрузкой приложения в контроллер следует настроить сетевой путь доступа среды разработки к контроллеру в соответствии с указаниями п. 3.5.2 (через сервисный порт USB или RS-232C) или п. 3.5.3 (по протоколу TCP или UDP).

При наличии пути доступа к контроллеру вкладка **Установки соединения** редактора устройств выглядит, как показано на рисунке 44 или на рисунке 47.

Если требуется полностью обновить ранее загруженное в контроллер приложение, необходимо выполнить команду **Компиляция – Очистить** для приложения, подлежащего загрузке.

После настройки пути доступа для загрузки приложения в контроллер выполните команду **Онлайн – Логин**. Если в контроллере выполняется ранее загруженное приложение, на экран монитора будет выведена диалоговая панель, показанная на рисунке 102.

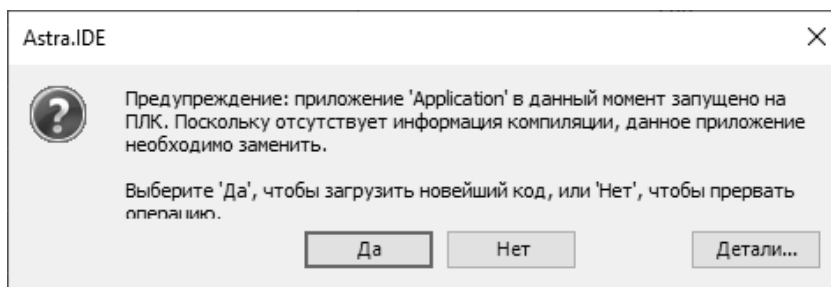



Рисунок 102 – Предложение о загрузке приложения при наличии в контроллере функционирующего приложения


После нажатия кнопки **Да** в диалоговой панели IDE МЭК 61131-3 выполнит генерацию исполняемого кода и, при отсутствии ошибок трансляции, будет выполнена полная загрузка приложения в контроллер. После успешной загрузки приложение в контроллере будет находиться в состоянии **останов**, о чем свидетельствует статус **стоп** в строке состояния среды разработки. Для запуска приложения следует нажать клавишу F5 или выполнить команду меню **Отладка – Старт**.

Если при генерации кода в панели сообщений среды разработки отображена информация об наличии некоторого количества ошибок компиляции, но отсутствуют конкретные сообщения об ошибках, следует выполнить команду **Компиляция – Очистить** или **Компиляция – Очистить все**, а затем повторить загрузку приложения в контроллер. В некоторых случаях перед трансляцией проекта может также потребоваться в свойствах проекта установить текущую версию компилятора и, при наличии в проекте форм визуализации, профиля визуализации в соответствии с указаниями п. 3.7.2.

	<p>После очередной полной загрузки модифицированного приложения, ранее загруженного в контроллер, все переменные приложения, кроме энергонезависимых сохраняемых переменных (RETAIN PERSISTENT), получают начальные значения. Более подробная информация об энергонезависимых переменных приведена в п. 4.2.8 настоящего документа.</p>
---	---

При полной загрузке модифицированного приложения может потребоваться его автоматический запуск сразу по завершении загрузки. Автоматический запуск приложения после загрузки возможен либо путем использования метода множественной загрузки согласно п. 4.6.4, либо путем использования программного запуска приложения из обработчика системного события *LegacyOnInit*.

Проект PlatformControl.project, поставляемый в Fastwel PLC Application Toolkit (см. таблицу 4), содержит пример реализации автоматического запуска приложения при загрузке приложения и сбросе. Для обеспечения доступа к функциям управления приложением необходимо подключить к проекту системную библиотеку CmpApp.

	<p>При полной загрузке модифицированного приложения, ранее загруженного в контроллер, перед заменой текущего приложения может потребоваться освободить ресурсы, занятые приложением до загрузки, выключить пользовательские светодиоды и т.п. К таким ресурсам относятся системные идентификаторы (хэндлы) открытых файлов, последовательных портов, сокетов, объектов системных библиотек и т.п.</p>
---	---

```

FUNCTION OnInitHandler : DWORD
VAR_IN_OUT
    EventPrm: FastwelCore.EVTPARAM_CmpFastwelCoreDummy;
END_VAR

VAR
    pApp : POINTER TO CmpApp.APPLICATION;
    OpResult : RTS_IEC_RESULT;
END_VAR

IF bStartStopApplication THEN
    // Если до предыдущего перезапуска или загрузки новой версии приложения была
    // установлена переменная bStartStopApplication, находим единственное приложение
    // по имени
    pApp := CmpApp.AppFindApplicationByName('Application', ADR(OpResult));

    IF OpResult <> 0 OR pApp = 0 THEN
        // если не получилось, пробуем иначе
        pApp := CmpApp.AppGetFirstApp(ADR(OpResult));
    END_IF

    IF OpResult = 0 AND pApp <> 0 THEN
        // приложение найдено, пробуем запустить:
        OpResult := CmpApp.AppStartApplication(pApp);

        IF OpResult = 0 THEN
            // запуск приложения выполнен успешно, увеличиваем счетчик
            dwAppStartOkCounter := dwAppStartOkCounter + 1;
        END_IF
    END_IF
END_IF

```

Функциональные блоки с переменными, сохраняющими значения системных идентификаторов внешних ресурсов, необходимо снабдить методом *FB\_Exit*, который будет вызван для каждого экземпляра блока перед завершением работы приложения. Более подробная информация о методе *FB\_Exit* приведена в подразделе **Руководство по программированию – Методы 'FB\_Init', 'FB\_Reinit', 'FB\_Exit'** интерактивной справочной системы IDE МЭК 61131-3.

Если системные идентификаторы ресурсов, подлежащих освобождению при полной загрузке приложения, хранятся в глобальных переменных или переменных, принадлежащих программным единицам типа *PROGRAM* (программа), необходимо добавить к таким программам действия (action), выполняющие освобождение ресурсов, а затем установить обработчик системного события *PrepareExit*, в котором вызвать действия по освобождению ресурсов требуемых программах.

Если после выключения и повторного включения питания или сброса контроллера требуется запустить ранее загруженное в него приложение, может потребоваться подключиться к контроллеру средой разработки командой **Онлайн – Логин** и выполнить команду **Онлайн – Создать загрузочное приложение**. Как правило, в выполнении данной команды нет необходимости, поскольку загрузочное приложение в проектах для платформ Fastwel создается в контроллере автоматически после полной загрузки. Однако при переносе приложения IDE МЭК 61131-3, созданного для другой платформы (контроллера другого производителя), у переносимого приложения может быть не установлена опция **Установки загрузочного приложения – Создавать неявное загрузочное приложение при загрузке**, доступная в свойствах приложения. В этом случае нужно либо включить данную опцию в свойствах приложения, либо выполнять команду **Онлайн – Создать загрузочное приложение** вручную.

#### 4.6.3. Загрузка онлайн-изменений

Большая часть изменений в исходном тексте приложения могут быть переданы в контроллер методом онлайн-изменения (Online Change, онлайн-замены). Данный способ модификации приложения в контроллере возможен только в том случае, если в каталоге размещения файла проекта также размещена актуальная информация о предыдущей успешной компиляции и загрузке приложения в контроллер. Данная информация формируется средой разработки во время последней загрузки приложения в контроллер и сохраняется в файле с именем:

*<имя проекта>. <имя устройства в приложении>. <имя приложения>. <ID приложения>. compileinfo*

При онлайн-изменении приложения в контроллер передаются код и данные изменившихся и новых программных единиц, в том числе код инициализации и присвоения начальных значений имеющимся и



новым переменным. Перед переключением задач приложения на исполнение кода обновленного приложения, код программных единиц, которые не изменились, копируется в новую область сегмента кода без изменений, а код изменившихся и новых программных единиц – в последующее свободное пространство памяти в сегменте кода. Изменившиеся данные программных единиц и глобальные переменные размещаются в свободном пространстве памяти в сегменте данных. Затем система исполнения, при необходимости, разрешает ссылки на код и данные в сегменте кода измененного приложения, после чего, по завершении очередных циклов задач, переключает текущий активный сегмент кода на область памяти, в которой размещены неизменные, измененные и новые программные единицы.

Перед загрузкой приложения в контроллер следует настроить сетевой путь доступа среды разработки к контроллеру в соответствии с указаниями п. 3.5.2 (через сервисный порт USB или RS-232C) или п. 3.5.3 (по протоколу TCP или UDP). При наличии пути доступа к контроллеру вкладка **Установки соединения** редактора устройств выглядит, как показано на рисунке 44 или на рисунке 47

После настройки пути доступа для загрузки приложения в контроллер выполните команду **Онлайн – Логин**. Если в контроллере выполняется ранее загруженное приложение, для которого среде разработки доступна актуальная информация о компиляции, на экран монитора будет выведена диалоговая панель, показанная на рисунке 103.

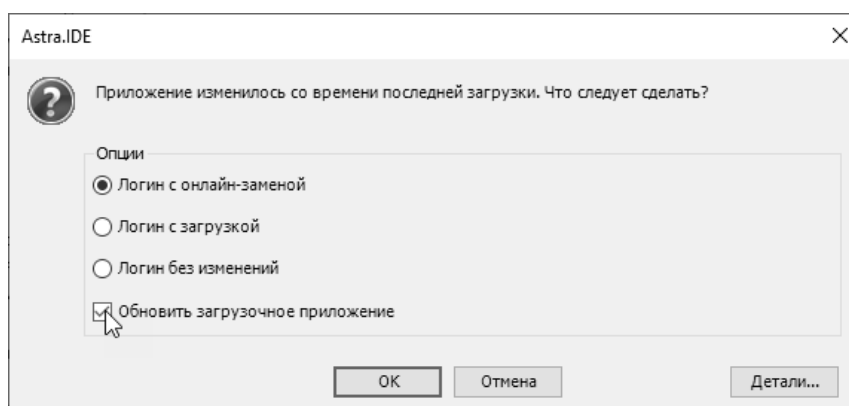



Рисунок 103 – Предложение об онлайн-изменении загруженного приложения



	<p>Перед началом загрузки изменений требуется отметить флажок <b>Обновить загрузочный проект</b> в диалоговой панели, показанной на рисунке 103, если после следующего перезапуска контроллера требуется запустить приложение с изменениями, которые сейчас будут переданы в контроллер методом онлайн-изменения.</p>
---	---

Для начала передачи в контроллер измененного приложения следует нажать **ОК** в диалоговой панели, показанной на рисунке 103.

	<p>После онлайн-изменения приложения, ранее загруженного в контроллер, все переменные приложения, которые были в нем до онлайн-изменения, сохраняют свои значения. В некоторых случаях это может привести к неправильной работе всего алгоритма или отдельных функциональных блоков, если их функционирование зависит от предыдущего состояния алгоритма.</p> <p>Функциональные блоки с переменными, подлежащими повторной инициализации при онлайн-изменении приложения, необходимо снабдить методом <i>FB_Reinit</i>, который будет вызван для каждого экземпляра блока в момент копирования переменных состояния экземпляра блока по новым адресам памяти при онлайн-изменении. Более подробная информация о методе <i>FB_Reinit</i> приведена в подразделе <b>Руководство по программированию – Методы 'FB_Init', 'FB_Reinit', 'FB_Exit'</b> интерактивной справочной системы IDE МЭК 61131-3.</p>
---	--

Если переменные, подлежащие повторной инициализации или специальной обработке при онлайн-изменении, принадлежат программным единицам типа *PROGRAM* (программа), необходимо добавить к таким программам действия (action), выполняющие требуемую повторную инициализацию, а затем установить обработчик системного события *OnlineChangeDone* (и быть может *LegacyOnInit*), в котором вызвать действия по повторной инициализации переменных в требуемых программах.



	<p>Если при генерации кода перед началом онлайн-изменения среда разработки обнаружила, что модификация носит существенный характер, на экран монитора выводится предупреждение о возможных непредвиденных последствиях онлайн-изменения. К таким случаям относится изменение размеров типов данных, изменение местоположения переменных и экземпляров функциональных блоков в памяти, изменение состава и типов входных, выходных и внутренних переменных программ и функциональных блоков, определяющих их состояние между вызовами, при которых алгоритм может оказаться в неожиданном участке кода или вычислить неправильные выходные данные.</p> <p>Перед нажатием кнопки <b>Да</b> для продолжения онлайн-изменения, следует нажать кнопку <b>Детали</b> в диалоговой панели предупреждения для просмотра изменений в приложении, которые среда разработки считает существенными.</p>
	<p>Очередная попытка загрузки приложения методом онлайн-изменения может завершиться неудачно из-за исчерпания системных ресурсов либо выполняться в течение времени, значительно превышающем время критического цикла алгоритма.</p>

#### 4.6.4. Множественная загрузка приложений в несколько контроллеров

При работе над проектом, который содержит приложения для нескольких контроллеров, все приложения могут быть транслированы и загружены во все контроллеры одной командой **Онлайн – Множественная загрузка**. Перед выполнением данной команды на вкладке **Установка соединения** редактора каждого целевого устройства в проекте, соответствующего контроллеру, необходимо задать сетевой путь в соответствии с указаниями п. 3.5 настоящего документа.

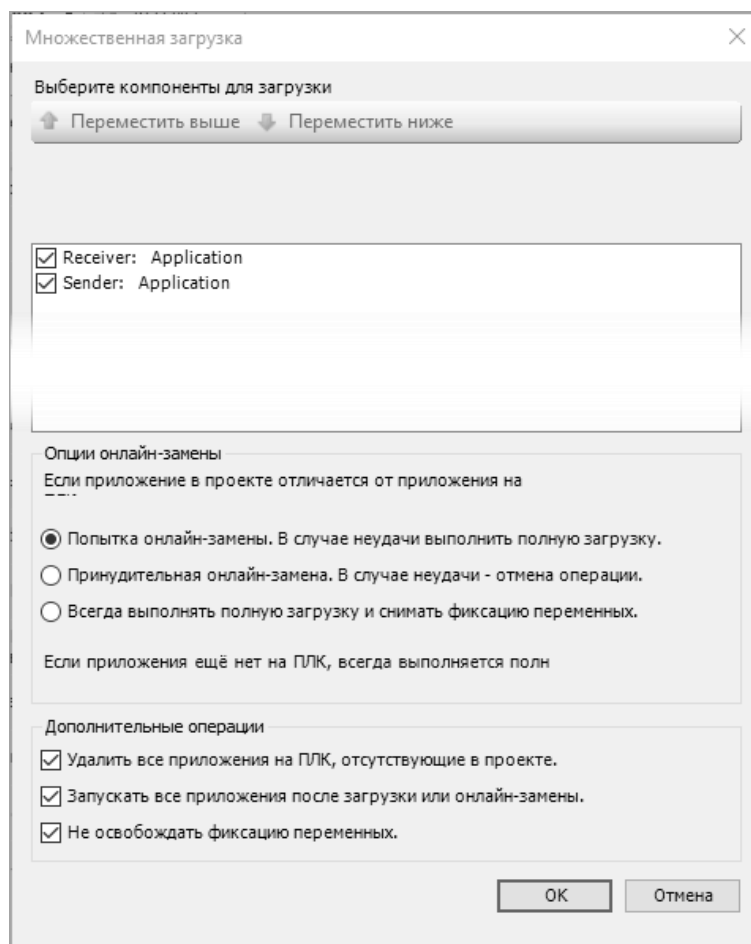




Рисунок 104 – Окно управления множественной загрузкой

После настройки сетевых путей ко всем контроллерам при выполнении команды меню **Онлайн – Множественная загрузка** на экран монитора будет выведено окно управления множественной загрузкой, показанное на рисунке 104.

Список **Выберите компоненты для загрузки** позволяет установить порядок загрузки приложений путем выбора некоторого приложения и нажатия кнопки  для загрузки его ранее остальных, расположенных ниже в списке, либо кнопки  – для загрузки приложения позднее остальных, расположенных выше.

Селектор **Опции онлайн-замены** предназначен для выбора способа множественной загрузки:

1. **Попытка онлайн-замены. В случае неудачи выполнить полную загрузку** – данный способ обеспечивает автоматическое онлайн-изменение приложений во всех контроллерах. Если онлайн-изменение приложения в каком-либо контроллере не может быть выполнено, производится полная загрузка приложения в данный контроллер.
2. **Принудительная онлайн-замена. В случае неудачи - отмена операции** – данный способ предназначен для выполнения загрузки только онлайн-изменением и, в случае неудачи, загрузка приложения в данный контроллер прерывается.
3. **Всегда выполнять полную загрузку** – множественная загрузка будет произведена только методом полной загрузки.

Параметр **Дополнительные операции – Запускать все приложения после загрузки или онлайн-изменения** обеспечивает автоматический запуск приложений на всех контроллерах после множественной загрузки.

## 4.7. Сервер OPC UA

### 4.7.1. Общие сведения

Системное программное обеспечение контроллеров Fastwel с системой исполнения IDE МЭК 61131-3, начиная с версии 3.0.x.x, содержит реализацию сервера OPC Unified Architecture (далее OPC UA) с поддержкой профиля Micro Embedded Device Server Profile согласно OPC Unified Architecture. Specification. Part 7: Profiles. Более подробная информация о спецификации OPC UA может быть получена на веб-узле <https://opcfoundation.org/>.

Сервер OPC UA обеспечивает выполнение следующих функций:

1. Соединение и обмен данными с клиентами OPC UA с использованием бинарного протокола обмена поверх TCP с количеством одновременных сеансов связи с клиентами не менее 10. Количество одновременных сеансов клиентов с сервером не ограничено, однако не рекомендуется использовать более 32 сеансов.
2. Обзор адресного пространства и предоставление информации об опубликованных типах данных, программных единицах и переменных приложения, исполняющегося на контроллере, клиентам OPC UA.
3. Обмен данными с клиентами OPC UA через сервисы мониторинга, подписки, чтения и записи атрибутов.
4. Аутентификацию соединений клиентов с сервером с использованием имен и паролей встроенных и дополнительных учетных записей пользователей системы исполнения.
5. Обмен данными с клиентами через защищенные каналы с использованием политики безопасности Basic256Sha256.



При количестве переменных в символьной конфигурации около 1000 установка связи между клиентом OPC UA и сервером OPC UA контроллера может занимать длительное время.

При количестве переменных более 10000, включенных в символьную конфигурацию контроллеров Fastwel, устойчивый обмен данными не гарантируется.

По умолчанию сервер OPC UA выключен в системных параметрах контроллеров. Кроме того, приложение IDE МЭК 61131-3 изначально не содержит информацию о переменных, доступных клиентам через сервер OPC UA.

Настоящий подраздел содержит указания по активации и настройке сервера OPC UA, а также по созданию и использованию открытых и безопасных сеансов связи с клиентскими приложениями.

Для обмена данными между клиентами OPC UA и сервером системы исполнения контроллера требуется выполнить следующие действия:

1. Активировать сервер OPC UA и настроить его параметры в веб-конфигураторе контроллера на странице **Параметры OPC UA**.
2. Если необходимо, использовать защищенные (безопасные) сеансы связи клиентов с сервером, требуется импортировать цифровой сертификат для сервера OPC UA или сгенерировать собственный самоподписанный сертификат. Это можно сделать в веб-конфигураторе на странице **Сертификаты безопасности** или в IDE МЭК 61131-3 на вкладке **Безопасность – Device**, которая становится доступной после установки пакета расширения CODESYS Security Agent, входящего в Fastwel PLC Application Toolkit.
3. В проекте IDE МЭК 61131-3 добавить в приложение символьную конфигурацию с включенной опцией **Поддержка функций OPC UA**, а затем включить в символьную конфигурацию переменные и программные единицы, подлежащие публикации и/или модификации через OPC UA (см. п. 4.2.6).
4. Установить связь клиентских приложений OPC UA с сервером контроллера (создать сеансы связи) и на клиентах создать подписки на данные, публикуемые сервером. Поиск публикуемых сервером данных осуществляется через стандартные сервисы обзора адресного пространства OPC UA.
5. Если необходимо использовать безопасные сеансы связи клиентов с сервером, при первой попытке установления соединения потребуются обменяться сертификатами безопасности между каждым клиентом и сервером, а затем переместить сертификаты из

карантинной зоны в доверенную. Это можно сделать в веб-конфигураторе на странице **Сертификаты безопасности** или в IDE МЭК 61131-3 на вкладке **Безопасность – Device**, упомянутой выше.

6. Если требуется перенести приложение контроллера вместе с системными параметрами, значениями энергонезависимых переменных и сертификатами безопасности на другой контроллер при помощи файла развертывания `port.url`, генерируемого командой `saveapp` оболочки ПЛК, при выполнении данной команды необходимо использовать дополнительный параметр `cert=all`. Более подробная информация о развертывании приложений приведена в п. 5.5 настоящего документа.



Для обеспечения переносимости самоподписанных собственных сертификатов перед генерацией собственных сертификатов безопасности необходимо определить параметр **Имя хоста** на странице **Параметры сети** в веб-конфигураторе.

При изложении последующих указаний по установлению соединения и обмену данными с клиентами OPC UA использовано приложение UaExpert компании Unified Automation GmbH, <https://www.unified-automation.com/>.

#### 4.7.2. Активация и настройка параметров сервера OPC UA

Активация сервера OPC UA выполняется в веб-конфигураторе контроллера на странице **Параметры OPC UA** включением опции **Разрешить OPC UA**, показанной на рисунке 105.

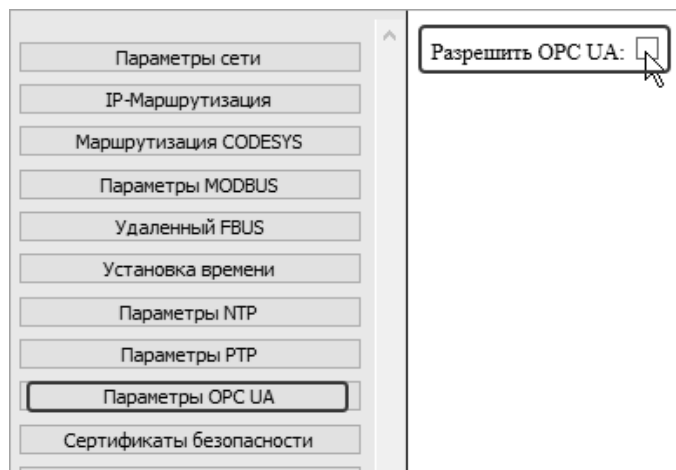


Рисунок 105 – Активация сервера OPC UA в системных параметрах контроллера

После включения данной опции становятся доступными параметры сервера, показанные на рисунке 106.

Разрешить OPC UA: ☒

Локальный IP адрес: По умолчанию ▼

Порт:

Мин. период опроса элемента, мс:

Политика безопасности: Default (None) ▼

Аутентификация: Default (Enabled) ▼

Разрешить простой текстовый пароль: Default (No) ▼

Имя хоста в качестве имени сервера: ☐


Имя сервера:

Рисунок 106 – Параметры сервера OPC UA

Параметр **Порт** определяет порт TCP, используемый сервером для ожидания входящих соединений клиентов OPC UA. По умолчанию используется стандартный порт сервера OPC UA 4840.

Параметр **Мин. период опроса элемента** предназначен для установки минимального возможного значения периода опроса (в мс) значений переменных приложения, публикуемых сервером OPC UA.

Данный параметр соответствует атрибуту *MinimumSamplingInterval* согласно OPC Unified Architecture. Specification. Part 4: Services и означает, что значение уточненного сервером периода опроса значений элементов мониторинга (переменных) *revisedSamplingInterval*, передаваемого сервером клиенту при добавлении элементов мониторинга в подписку, будет не менее *MinimumSamplingInterval*.

	<p>Не рекомендуется устанавливать слишком малые значения данного параметра, т.к. частый опрос и передача значений переменных клиенту OPC UA по сети существенно повышает потребление вычислительных ресурсов на клиенте.</p>
---	--

Параметр **Политика безопасности** позволяет использовать шифрование и цифровую подпись при подключении и обмене данными между клиентами и сервером, а параметры **Аутентификация** и **Разрешить простой текстовый пароль** – учетные данные (имена пользователей и пароли). Более подробная информация о настройке и использовании безопасных сеансов связи клиентов с сервером OPC UA приведена в пп. 4.7.6 – 4.7.8.

Опция **Имя хоста в качестве имени сервера** позволяет выбрать имя хоста, назначенное контроллеру на странице **Параметры сети** веб-конфигуратора, в качестве имени сервера OPC UA, отображаемого первым в иерархии при обзоре адресного пространства сервера и являющегося первым элементом символьного идентификатора путей к элементам данных сервера.


Поле **Имя сервера** (начиная с версии 3.3.x.x системного ПО контроллеров Fastwel) позволяет задать произвольное имя сервера OPC UA.

После включения опции **Разрешить OPC UA** и настройки параметров сервера необходимо в веб-конфигураторе нажать кнопку **Применить конфигурацию** и убедиться, что выполнен повторный запуск контроллера.

После применения конфигурации сервер OPC UA становится доступным на всех сконфигурированных сетевых интерфейсах контроллера.


### 4.7.3. Доступ сервера OPC UA к переменным приложения

Для обеспечения возможности чтения и записи через OPC UA переменных приложения IDE МЭК 61131-3, функционирующего на контроллере, необходимо добавить в приложение элемент проектной информации *Символьная конфигурация* с включенной опцией **Поддержка функций OPC UA**, после чего выбрать переменные приложения, к которым требуется иметь доступ через OPC UA, установить для них тип доступа (чтение, запись, чтение-запись), скомпилировать и загрузить приложение в контроллер.

	<p>Перед загрузкой приложения с символьной конфигурацией обмена с клиентами OPC UA, следует в веб-конфигураторе контроллера включить опцию <b>Разрешить OPC UA</b> и применить конфигурацию.</p> <p>Кроме того, необходимо убедиться, что версия целевого устройства (платформы) <i>CPMXXX-ZZ Programmable Automation Controller</i> в проекте не ниже 3.255.0.0.</p>
---	---

Указания по созданию символьной конфигурации, добавлению переменных и установке вида доступа приведены в п. 4.2.6.

Опция **Поддержка функций OPC UA** включена по умолчанию при создании символьной конфигурации, как показано на рисунке 107. Опция также может быть включена и выключена в меню **Установки** вкладки **Символьная конфигурация**, как показано на рисунке 108.

	<p>Редактор символьной конфигурации, помимо отдельных переменных и массивов переменных встроенных и пользовательских типов данных, позволяет сделать доступными через OPC UA списки глобальных переменных, программные единицы типа PROGRAM и экземпляры функциональных блоков, как показано на рисунке 109.</p>
---	--

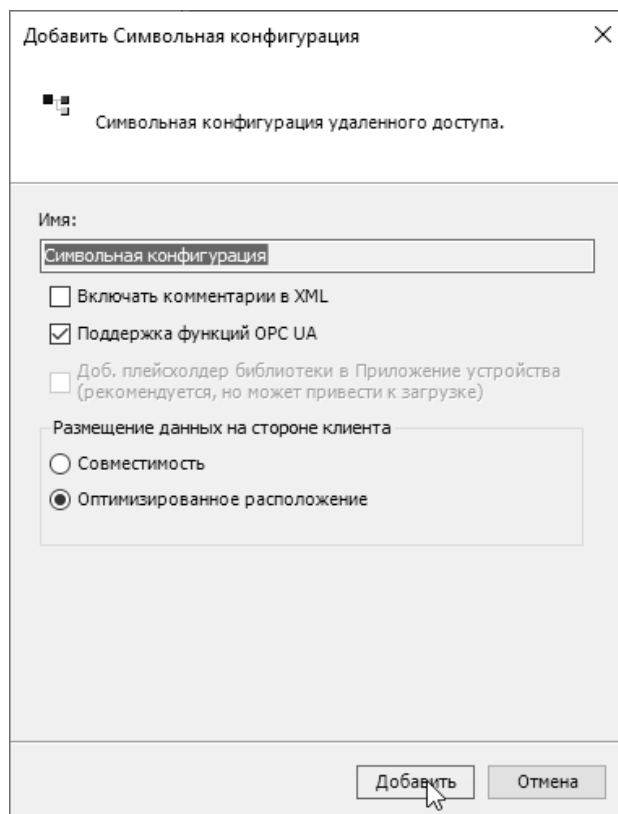


Рисунок 107 – Создание символьной конфигурации с опцией "Поддержка функций OPC UA"

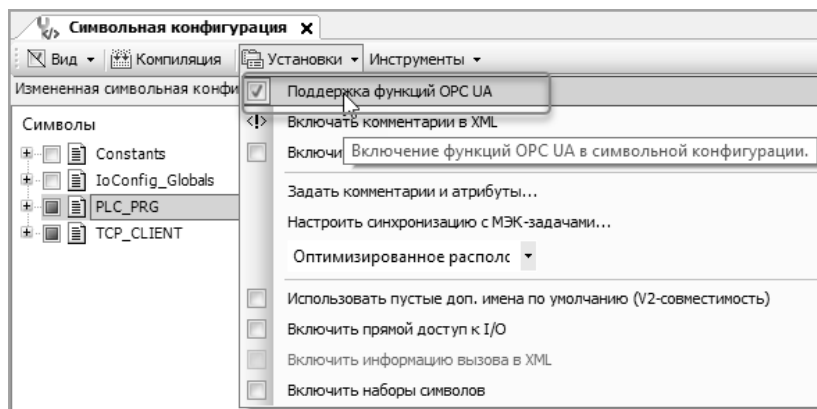


Рисунок 108 – Включение поддержки функций OPC UA в символьной конфигурации

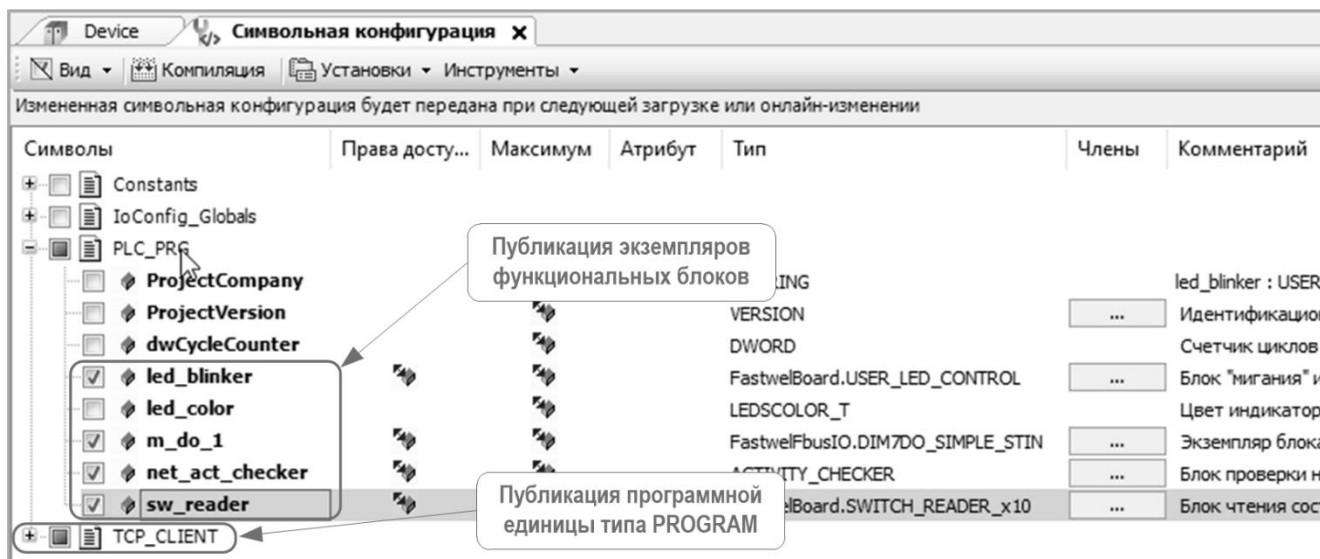



Рисунок 109 – Варианты публикации программных единиц в символьной конфигурации



	<p>Не рекомендуется чрезмерно использовать возможность добавления в символьную конфигурацию массивов переменных и функциональных блоков, т.к. это может привести к значительному расходу оперативной памяти контроллера и к увеличению времени, требуемому клиентам OPC UA для обзора адресного пространства сервера и создания подписок на элементы данных при создании сеансов связи с сервером.</p> <p>При количестве переменных в символьной конфигурации около 1000 установка связи между клиентом OPC UA и сервером OPC UA контроллера.</p> <p>При количестве переменных более 10000, включенных в символьную конфигурацию контроллеров Fastwel, устойчивый обмен данными не гарантируется.</p>
---	---

#### 4.7.4. Соединение клиента с сервером OPC UA

При изложении настоящих указаний предполагается, что в веб-конфигураторе контроллера активирован сервер OPC UA с параметрами по умолчанию (см. п. 4.7.2) без аутентификации и без включенной политики безопасности, и из IDE МЭК 61131-3 в контроллер загружено приложение, содержащее символьную конфигурацию с переменными, публикуемыми через OPC UA (см. п. 4.7.3).


Также предполагается, что в веб-конфигураторе на странице **Параметры сети** установлено значение *10.0.0.0/8* для параметров **IP адрес LAN1 (база)** и **IP адрес LAN2 (база)**, а на переключателях 1 – 8 установлено значение 7, т.е. сетевым интерфейсам назначены IP-адреса 10.0.0.7 и 10.0.0.8 с маской подсети длиной 8 бит (255.0.0.0).

Для установления незащищенного соединения между клиентским приложением UaExpert и сервером OPC UA контроллера:

1. На компьютере запустить UaExpert и выполнить команду меню **Server – Add** или нажать кнопку  в панели инструментов UaExpert.
2. В появившейся диалоговой панели **Add Server** в поле **Configuration Name** ввести имя создаваемой конфигурации соединений клиента с сервером, например, *UAConfig1*, и открыть вкладку **Advanced**.
3. На вкладке **Advanced** в поле **Server Information – Endpoint Url** ввести адрес сервера в следующем формате:  
`opc.tcp://<ip-адрес>:<tcp-порт>`  
где *ip-адрес* – IP-адрес сетевого интерфейса контроллера, через который доступен сервер OPC UA, а *tcp-порт* – номер порта, на котором сервер OPC UA ожидает входящие соединения клиентов.  
В рассматриваемом примере следует ввести:  
`opc.tcp://10.0.0.7:4840`
4. В поле **Session Settings – Session Name** ввести имя сеанса обмена данными с сервером, например, *UnsecuredSession1*.
5. В полях **Security Settings – Security Policy** и **Message Security Mode** установить значение *None*.
6. Переключатель **Authentication Settings** оставить в положении **Anonymous**.  
Содержимое вкладки **Advanced** диалоговой панели **Add Server** приложения UaExpert показано на рисунке 110.
7. Закрыть диалоговую панель **Add Server** нажатием **ОК**. Для того, чтобы UaExpert попытался создать активный сеанс связи (установить соединение) с сервером сразу после закрытия диалоговой панели, перед нажатием **ОК** отметить флажок **Connect Automatically**. В древовидном списке **Project** главного окна UaExpert появится элемент, соответствующий созданному описанию сеанса связи с сервером OPC UA, как показано на рисунке 111. Если перед закрытием диалоговой панели **Add Server** не был отмечен флажок **Connect Automatically**, то сеанс будет находиться в несоединенном состоянии, о чем свидетельствует значок  слева от имени сеанса.



8. Для соединения с сервером щелкнуть правой кнопкой мыши над сеансом в древовидном списке **Project** и выполнить команду **Connect** в контекстном меню, как показано на рисунке 111.

При успешном соединении клиента с сервером слева от имени сеанса будет отображен значок , в области диагностики **Log** главного окна UaExpert будут выведены сообщения об успешном соединении сеанса *UAConfig1*, а в области **Address Space** появится корневой элемент *Root* адресного пространства сервера с категориями объектов *Objects*, *Types* и *Views*, как показано на рисунке 112.

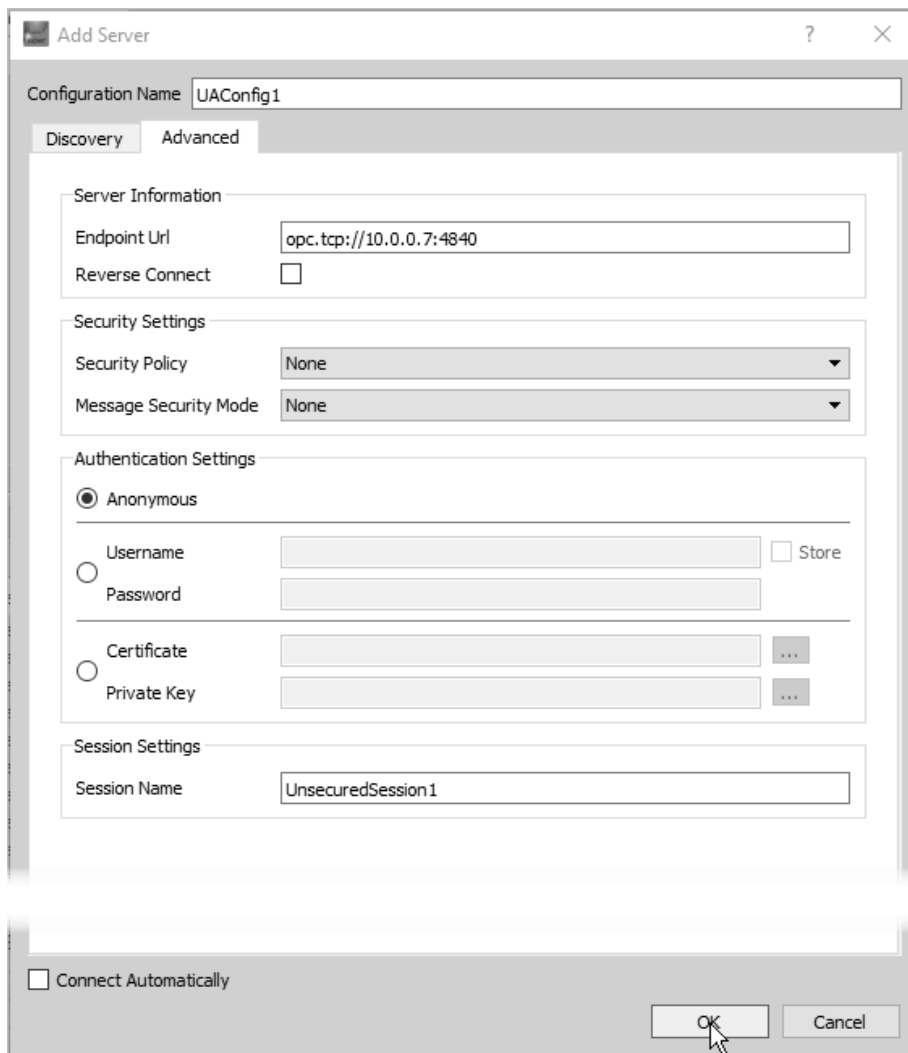


Рисунок 110 – Настройка параметров незащищенного соединения с сервером в UaExpert

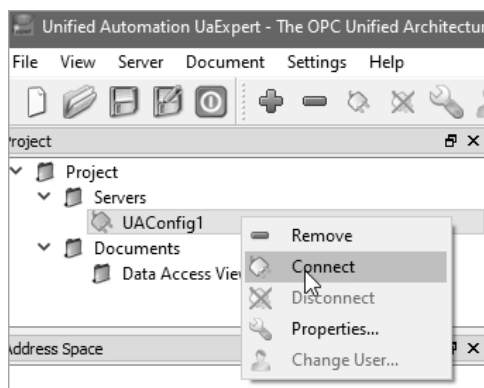


Рисунок 111 – Сеанс связи с сервером OPC UA в UaExpert

Для завершения сеанса связи клиента с сервером следует выполнить команду **Disconnect** в контекстном меню требуемого сеанса в древовидном списке **Project**.

Для установки таймаута сеансов связи UaExpert с серверами OPC UA следует выполнить команду меню **Settings – Configure UaExpert** и в окне **Configure UaExpert** установить требуемое значение параметра *General.SessionTimeout* (в мс), как показано на рисунке 113.



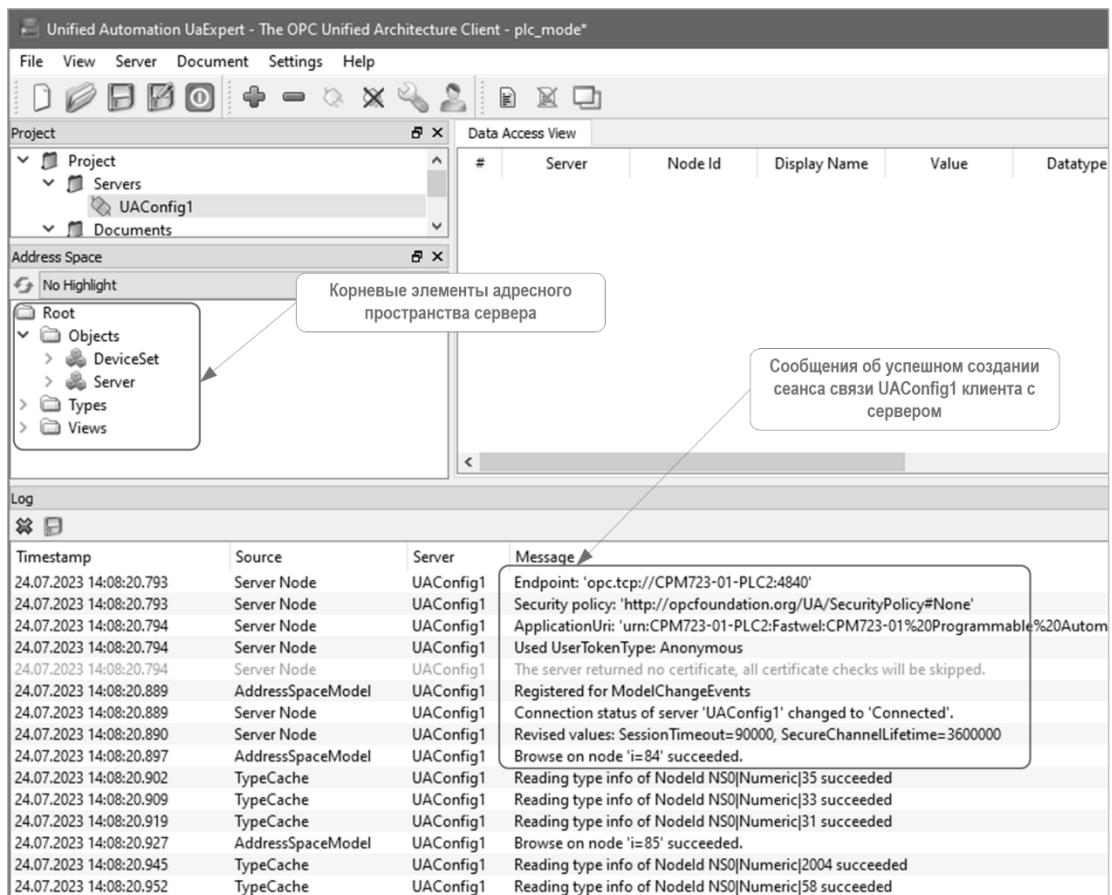


Рисунок 112 – Успешное соединение с сервером OPC UA в UaExpert



При отсутствии активности со стороны клиента или в случае (внезапного) прекращения работы клиента/клиентского приложения без завершения ранее созданных сеансов с сервером, данные сеансы будут оставаться на сервере в течение интервала времени *RequestedSessionTimeout* (в мс), определенного клиентом при установлении соединения. Более подробная информация приведена в спецификации OPC Unified Architecture. Specification. Part 4: Services.

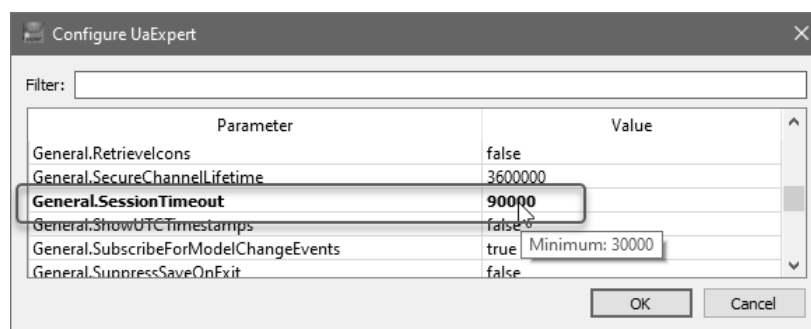


Рисунок 113 – Установка таймаута сеансов связи с серверами OPC UA в UaExpert



При использовании других клиентских приложений и устройств с функцией клиента OPC UA следует обратить внимание на наличие поддержки данного параметра и его значение по умолчанию, т.к. каждый сеанс с клиентом OPC UA на сервере может требовать большого количества системных ресурсов контроллера. Например, в Matrikon OPC UA Explorer версии 100.1 значение таймаута сеансов связи с сервером составляет 5 часов, поэтому при завершении работы данного приложения без завершения сеансов связи с серверами OPC UA все созданные сеансы связи будут оставаться незакрытыми в течение 5 часов.

#### 4.7.5. Обзор адресного пространства и обмен данными с сервером

При успешном соединении клиента с сервером становится доступным для обзора адресное пространство объектов сервера, сопоставленных с различными элементами данных системы исполнения контроллера. Пример обзора корневых элементов адресного пространства сервера OPC UA контроллера СРМ810-03 показан на рисунке 112. Основные элементы структуры адресного пространства сервера OPC UA с раскрытыми высокоуровневыми элементами стандартных типов *DeviceSet* и *Server* представлены на рисунке 114.

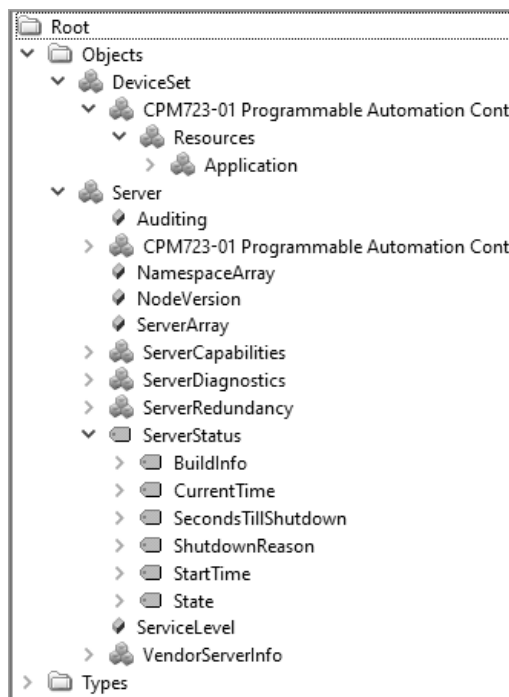



Рисунок 114 – Основные элементы адресного пространства сервера OPC UA контроллера в UaExpert

Указанные высокоуровневые элементы содержат один и тот же элемент *CPMXXX-ZZ Programmable Automation Controller – Resources*, через который в адресном пространстве сервера представлено приложение, загруженное в контроллер из среды разработки IDE МЭК 61131-3.

Имя элемента *Application* под *CPMXXX-ZZ Programmable Automation Controller – Resources* совпадает с именем, заданным для приложения ГОСТ Р МЭК 61131-3 в проекте IDE МЭК 61131-3. Данный элемент адресного пространства соотнесен с приложением IDE МЭК 61131-3, функционирующим в контроллере, и обеспечивает доступ к переменным, добавленным в символьную конфигурацию, и статическим свойствам приложения.

На рисунке 115 представлены некоторые элементы адресного пространства приложения с символьной конфигурацией, показанной на рисунке 109.

Элементы со значком  являются статическими свойствами приложения, значения и другие атрибуты которых отображаются только в панели **Attributes** UaExpert. Из свойств приложения интерес могут представлять *SerialNumber*, содержащий заводской номер контроллера, и *SoftwareRevision*, который содержит версию приложения, введенную пользователем в поле **Версия** на вкладке **Общие** диалоговой панели **Информация проекта**, которая выводится на экран по команде меню **Проект – Информация проекта** IDE МЭК 61131-3.

Категория *GlobalVars* содержит списки глобальных переменных и отдельные глобальные переменные, опубликованные сервером через символьную конфигурацию. В примере на рисунке 115 глобальные переменные не опубликованы.

Категория *Programs* содержит программные единицы типа PROGRAM, переменные программ и экземпляры функциональных блоков, опубликованные сервером.

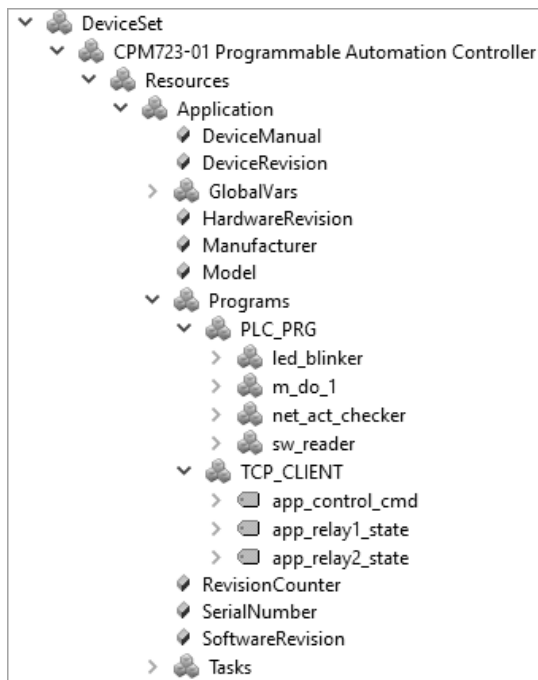


Рисунок 115 – Элементы адресного пространства приложения

Для создания подписки на опубликованные элементы адресного пространства, за исключением статических свойств, следует перетащить их из адресного пространства в представление **Data Access View**, как показано на рисунке 116, и отпустить левую кнопку мыши.

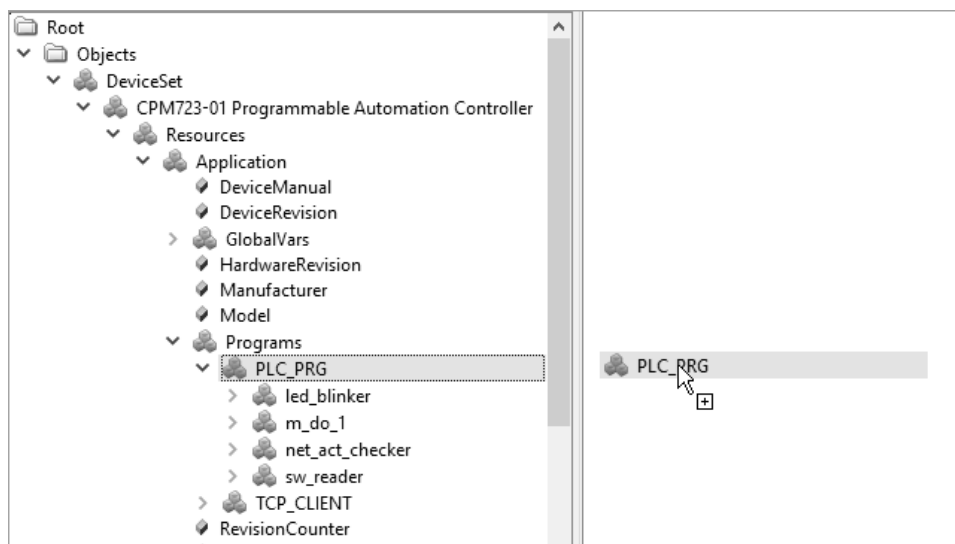


Рисунок 116 – Создание подписки на множество элементов программной единицы PLC\_PRG

UaExpert попытается создать подписку и рекурсивно добавить в нее все подэлементы, содержащиеся в перенесенном элементе и добавленные в символьную конфигурацию приложения, после чего, в случае удачи, в представлении **Data Access View** будут отображаться все добавленные элементы со значениями, признаками качества, типами данных и метками времени, как показано на рисунке 117.



Если элемент данных, добавляемый в подписку, содержит несколько тысяч подэлементов, включая массивы, структурные типы, функциональные блоки и т.п., то добавление элементов в подписку может занимать несколько минут каждый раз при создании данного сеанса связи клиента с сервером.

Data Access View								
#	Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Status
1	UAConfig1	NS4[String]  var ...	Failure	false	Boolean	14:20:01.799	14:20:01.799	Good
2	UAConfig1	NS4[String]  var ...	Period	100	Int64	14:20:01.799	14:20:01.799	Good
3	UAConfig1	NS4[String]  var ...	SwitchStates	{true,true,false,f...	Boolean	14:20:01.799	14:20:01.799	Good
4	UAConfig1	NS4[String]  var ...	next_time_to_run	808296	Int64	14:20:20.709	14:20:20.709	Good
5	UAConfig1	NS4[String]  var ...	sw_mask_prev	3	UInt32	14:20:01.799	14:20:01.799	Good
6	UAConfig1	NS4[String]  var ...	Active	true	Boolean	14:20:01.799	14:20:01.799	Good
7	UAConfig1	NS4[String]  var ...	ActivityCounter	63857	UInt32	14:20:20.709	14:20:20.709	Good
8	UAConfig1	NS4[String]  var ...	CheckPeriod	100	Int64	14:20:01.799	14:20:01.799	Good
9	UAConfig1	NS4[String]  var ...	Error	false	Boolean	14:20:01.799	14:20:01.799	Good
10	UAConfig1	NS4[String]  var ...	ErrorsCounter	0	UInt32	14:20:01.799	14:20:01.799	Good
11	UAConfig1	NS4[String]  var ...	Performance	80	Float	14:20:20.086	14:20:20.086	Good
12	UAConfig1	NS4[String]  var ...	Reset	false	Boolean	14:20:01.799	14:20:01.799	Good
13	UAConfig1	NS4[String]  var ...	act_count_prev	63855	UInt32	14:20:20.707	14:20:20.708	Good
14	UAConfig1	NS4[String]  var ...	err_count_prev	0	UInt32	14:20:01.799	14:20:01.799	Good
15	UAConfig1	NS4[String]  var ...	next_time_to_c...	808296	Int64	14:20:20.707	14:20:20.707	Good
16	UAConfig1	NS4[String]  var ...	reset_prev	false	Boolean	14:20:01.799	14:20:01.799	Good
17	UAConfig1	NS4[String]  var ...	Controls	{false,false,false...	Boolean	14:20:01.799	14:20:01.799	Good
18	UAConfig1	NS4[String]  var ...	ControlsReadba...	{false,false,false...	Boolean	14:20:01.799	14:20:01.799	Good
19	UAConfig1	NS4[String]  var ...	Inputs	Double click to ...	ExtensionObject	14:20:01.799	14:20:01.799	Good
20	UAConfig1	NS4[String]  var ...	Outputs	Double click to ...	ExtensionObject	14:20:01.799	14:20:01.799	Good
21	UAConfig1	NS4[String]  var ...	bDiagnostics	{false,false,false...	Boolean	14:20:01.798	14:20:01.799	Good
22	UAConfig1	NS4[String]  var ...	BlinkPeriod	1000	Int64	14:20:01.798	14:20:01.798	Good
23	UAConfig1	NS4[String]  var ...	Color	1	Int16	14:20:01.798	14:20:01.798	Good
24	UAConfig1	NS4[String]  var ...	ENABLE	true	Boolean	14:20:01.798	14:20:01.798	Good
25	UAConfig1	NS4[String]  var ...	OUT	true	Boolean	14:20:20.396	14:20:20.396	Good
26	UAConfig1	NS4[String]  var ...	TIMEHIGH	500	Int64	14:20:01.798	14:20:01.798	Good
27	UAConfig1	NS4[String]  var ...	TIMELOW	500	Int64	14:20:01.798	14:20:01.798	Good
28	UAConfig1	NS4[String]  var ...	bDiagnostics[1]	false	Boolean	14:20:01.798	14:20:01.798	Good
29	UAConfig1	NS4[String]  var ...	bDiagnostics[2]	false	Boolean	14:20:01.798	14:20:01.798	Good
30	UAConfig1	NS4[String]  var ...	bDiagnostics[3]	false	Boolean	14:20:01.798	14:20:01.798	Good
31	UAConfig1	NS4[String]  var ...	bDiagnostics[4]	false	Boolean	14:20:01.798	14:20:01.798	Good
32	UAConfig1	NS4[String]  var ...	bDiagnostics[5]	false	Boolean	14:20:01.798	14:20:01.798	Good
33	UAConfig1	NS4[String]  var ...	bDiagnostics[6]	false	Boolean	14:20:01.798	14:20:01.798	Good
34	UAConfig1	NS4[String]  var ...	bDiagnostics[7]	false	Boolean	14:20:01.798	14:20:01.798	Good
35	UAConfig1	NS4[String]  var ...	bDiagnostics[8]	false	Boolean	14:20:01.798	14:20:01.798	Good
36	UAConfig1	NS4[String]  var ...	Control	0	Byte	14:20:01.798	14:20:01.798	Good
37	UAConfig1	NS4[String]  var ...	Diagnostics	0	Byte	14:20:01.798	14:20:01.798	Good
38	UAConfig1	NS4[String]  var ...	OutputStates	0	Byte	14:20:01.798	14:20:01.798	Good

Рисунок 117 – Мониторинг элементов данных в подписке UaExpert

Для записи значения в элемент данных, соотношенный с переменной приложения, следует дважды щелкнуть в ее ячейке **Value** в представлении **Data Access View**, ввести новое значение и нажать Enter.

Информация о производителе и версии системного программного обеспечения контроллера, а также о версии системы исполнения приложений контроллера, может быть получена в элементе *Server – ServerStatus – BuildInfo*, как показано на рисунке 118.

The screenshot displays the UaExpert interface. On the left, the 'Project' tree shows 'Servers' > 'UAConfig1' > 'ServerStatus' > 'BuildInfo' selected. The 'Data Access View' table (Figure 117) is visible in the background. The 'Attributes' panel on the right shows the details for the 'BuildInfo' attribute, including fields like 'ProductUri', 'ManufacturerName', 'ProductName', 'SoftwareVersion', 'BuildNumber', and 'BuildDate'. The 'ServerStatus' tree on the left is expanded to show 'BuildInfo'.

Рисунок 118 – Информация о контроллере в адресном пространстве сервера

Атрибут *SoftwareVersion* содержит версию системного программного обеспечения, а *BuildNumber* – версию интегрированной в системное программное обеспечения системы исполнения приложений МЭК 61131-3.

Начиная с версии 3.2.x.x системного ПО контроллера, информация о текущем режиме работы контроллера может быть получена в элементе *Server – ServerStatus – State*. Данный элемент имеет тип Int32 и может принимать следующие значения:

- 0 – нормальный режим;
- 1 – безопасный режим по ошибке в ранее загруженном приложении;
- 2 – безопасный режим без приложения (после заводского сброса из среды разработки).

#### 4.7.6. Общие сведения о безопасном взаимодействии клиентов с сервером OPC UA

Для обеспечения безопасного взаимодействия по протоколу OPC UA между клиентами и сервером могут использоваться следующие механизмы:

1. Аутентификация пользователя при создании сеанса связи клиента с сервером путем передачи от клиента серверу имени пользователя и пароля.
2. Создание защищенного канала между клиентом и сервером, сопровождающееся обменом криптографическими ключами, с последующим симметричным шифрованием и/или цифровой подписью остальных сообщений, передаваемых по защищенному каналу. Для генерации цифровой подписи используется алгоритм SHA256, а для шифрования – BASIC256.

Данные механизмы не зависят друг от друга и могут использоваться вместе или отдельно. При совместном использовании учетные данные, передаваемые в процессе аутентификации, подвергаются дополнительному шифрованию. При использовании аутентификации без шифрования имя пользователя и пароль передаются "открытым текстом".

#### 4.7.7. Аутентификация пользователя

Для использования аутентификации необходимо в веб-конфигураторе контроллера на странице **Параметры OPC UA** включить опцию **Аутентификация: Enforced**. Если параметр **Разрешить простой текстовый пароль** имеет значение *Yes*, как показано на рисунке 119, то пароль пользователя будет передаваться клиентом серверу без шифрования.

The screenshot shows a configuration window for OPC UA parameters. The settings are as follows:

- Разрешить OPC UA:** ☒
- Порт:** 4840
- Мин. период опроса элемента, мс:** 100
- Политика безопасности:** Default (None)
- Аутентификация:** Enforced
- Разрешить простой текстовый пароль:** Yes

Рисунок 119 – Включение обязательной аутентификации при создании сеансов связи с сервером OPC UA

После применения конфигурации, сопровождающейся перезапуском контроллера, создание сеансов клиентов с сервером OPC UA будет возможным только в случае успешного ввода имени пользователя и пароля одной из учетных записей подсистемы безопасности контроллера, которым в IDE МЭК 61131-3 на вкладке **Права доступа** редактора целевого устройства, соответствующего контроллеру, разрешено устанавливать соединение с сервером OPC UA.

Если параметр **Разрешить простой текстовый пароль** имеет значение по умолчанию *No*, то при передаче учетных данных от клиента серверу пароль будет зашифрован с использованием алгоритма SHA256 RSA-OAEP. Чтобы иметь возможность шифрования пароля при аутентификации клиентов OPC UA, для программного компонента *CmpOPCUAServer* системы исполнения контроллера должен быть установлен или сгенерирован действительный цифровой сертификат (см. п. 4.7.10 или п. 4.7.11).

По умолчанию разрешено устанавливать соединение с сервером OPC UA с именами и паролями учетных записей, относящихся к группе *Administrators*, в том числе системных учетных записей *Administrator* и *Everyone*, однако, поскольку учетная запись *Everyone* по умолчанию имеет пустой пароль, она не может использоваться для создания соединений клиентов с сервером OPC UA с включенной аутентификацией.

Более подробная информация о правах доступа приведена в п. 5.3 настоящего документа.

Для создания сеанса связи UaExpert с сервером OPC UA контроллера с использованием аутентификации без шифрования пароля учетной записи пользователя необходимо выполнить следующие действия:

1. В веб-конфигураторе контроллера включить обязательную аутентификацию с разрешением простого текстового пароля, как показано на рисунке 119, применить конфигурацию и убедиться, что контроллер повторно запущен после перезапуска.
2. В UaExpert открыть диалоговую панель свойств ранее созданного сеанса связи с сервером и установить опцию **Authentication Settings – Username/Password**, как показано на рисунке 120, после чего закрыть диалоговую панель нажатием **ОК**.
3. Выполнить команду **Connect** в контекстном меню сеанса.
4. В появившейся диалоговой панели **Enter user credentials** ввести имя пользователя и пароль учетной записи *Administrator*.
5. Нажать кнопку **Ignore** в диалоговой панели **Connect Error**, предупреждающей о недостаточной степени защиты сеанса, и сеанс связи будет установлен.



Рисунок 120 – Включение режима аутентификации для связи с сервером OPC UA в UaExpert

Для создания сеанса связи UaExpert с сервером OPC UA контроллера с использованием аутентификации с шифрованием пароля учетной записи пользователя необходимо:

1. Сгенерировать или импортировать цифровой сертификат для программного компонента *СтрOPCUAServer* системы исполнения контроллера в соответствии с указаниями п. 4.7.10 или п. 4.7.11.
2. В веб-конфигураторе включить обязательную аутентификацию без разрешения простого текстового пароля (**Разрешить простой текстовый пароль:No**), применить конфигурацию и убедиться, что контроллер перезапустился и продолжил работу.
3. В UaExpert открыть диалоговую панель свойств ранее созданного сеанса связи с сервером и установить опцию **Authentication Settings – Username/Password**, как показано на рисунке 120, после чего закрыть диалоговую панель нажатием **ОК**.
4. Выполнить команду **Connect** в контекстном меню сеанса.
5. В появившейся диалоговой панели **Enter user credentials** ввести имя пользователя и пароль учетной записи *Administrator*. Соединение клиента с сервером будет успешно установлено.

#### 4.7.8. Механизм цифровой подписи и шифрования OPC UA

Согласно спецификации OPC Unified Architecture. Specification. Part 2: Security Model, клиент OPC UA при начальном взаимодействии с сервером получает у сервера установленную для него политику безопасности (Security Policy) и режим безопасности сообщений (Message Security Mode), а также цифровой сертификат сервера с публичным (открытым) ключом шифрования.

Затем клиент посылает серверу свой публичный ключ в цифровом сертификате в специальном сообщении о создании защищенного канала с использованием асимметричного шифрования с публичным ключом сервера и асимметричными цифровыми подписями с приватным (закрытым) ключом клиента.

Сервер расшифровывает сообщение своим приватным ключом и проверяет асимметричную цифровую подпись ранее полученным от клиента публичным ключом шифрования. "Секретная" информация клиента совместно с "секретной" информацией сервера используется для формирования набора криптографических ключей, которые затем будут применяться для защиты остальных сообщений, передаваемых по защищенному каналу. В дальнейшем набор сформированных ключей будет периодически меняться.

Установление защищенного соединения клиента с сервером OPC UA контроллера состоит из следующих операций:

1. Включение политики безопасности *Basic256Sha256* и режима безопасности сообщений *Sign* или *Sign & Encrypt* в веб-конфигураторе контроллера.
2. Импорт или генерация собственных сертификатов безопасности, как минимум, для компонента *StrOPCUAServer* в веб-конфигураторе на странице **Сертификаты безопасности** или в IDE МЭК 61131-3 на вкладке **Device** редактора **Безопасность**.
3. Настройка параметров сеанса связи с сервером у клиента OPC UA с использованием подходящей для сервера политики безопасности и режима безопасности сообщений.
4. Первоначальное соединение клиента с сервером для обмена сертификатами и перевод сертификата сервера из карантина в доверенную зону клиента.
5. Перевод сертификата клиента из карантина в доверенную зону сервера в веб-конфигураторе контроллера на странице **Сертификаты безопасности** или в IDE МЭК 61131-3 на вкладке **Device** редактора **Безопасность**.
6. Повторное соединение клиента с сервером для обзора адресного пространства и/или обмена данными.

Операции с 1 по 5 выполняются однократно до тех пор, пока не закончился срок действия сертификатов безопасности, использованных в процессе на шагах с 1 по 5.

#### 4.7.9. Разрешение защищенных сеансов связи с сервером OPC UA

Для разрешения только защищенных сеансов связи с сервером OPC UA контроллера следует:

1. В веб-конфигураторе контроллера на странице **Параметры сети** в поле **Имя хоста** ввести имя контроллера в сети. Имя должно состоять из символов латинского алфавита, цифр от 0 до 9, символов подчеркивания или короткого тире, и иметь длину не более 48 символов, например: *CPM81003-PLC1*.  
Ввод имени хоста нужен для того, чтобы сгенерированный самоподписанный цифровой сертификат не зависел от серийного номера контроллера. В таком случае при выходе из строя данного контроллера будет возможность использовать резервную копию всей его конфигурации, сформированную в файле *norm.upl* командой оболочки ПЛК *saveapp cert-all*, для последующего развертывания на исправном контроллере, заменяющем вышедший из строя.
2. На странице **Параметры OPC UA** установить значение параметра **Политика безопасности**: *Basic256Sha256*.
3. В появившемся выпадающем списке **Режим безопасности сообщений** установить значение *Sign*, как показано на рисунке 121, или *Sign & Encrypt*.  
При установке режима *Sign* будет возможность создания защищенных сеансов связи с использованием либо только цифровой подписи каждого сообщения, либо цифровой подписи и шифрования.  
Режим *Sign & Encrypt* позволяет клиенту использовать только режим обмена сообщениями с цифровой подписью и шифрованием.
4. Нажать **Применить конфигурацию** и убедиться, что контроллер перезапустился и продолжил работу.

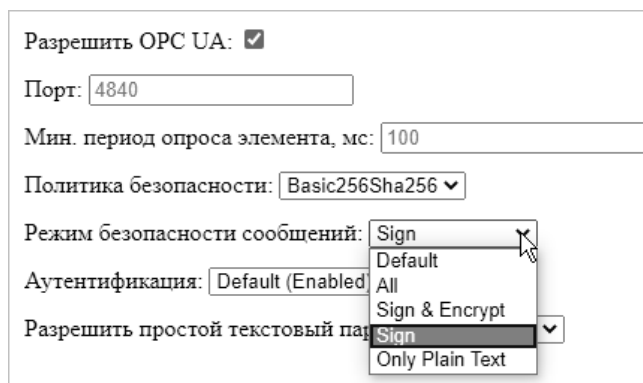


Рисунок 121 – Разрешение только защищенных сеансов связи с сервером OPC UA контроллера

#### 4.7.10. Просмотр и генерация сертификатов безопасности в веб-конфигураторе контроллера

Для генерации самоподписанного сертификата безопасности сервера OPC UA в веб-конфигураторе контроллера следует:

1. На странице **Сертификаты безопасности** выбрать категорию сертификатов *Все* нажать кнопку **Запросить**. В списке сертификатов могут появиться собственные сертификаты, которые были автоматически сгенерированы ранее в подсистеме безопасности контроллера, как показано на рисунке 122. Данные сертификаты в поле **commonName** должны содержать имя хоста, ранее установленное на странице **Параметры сети** веб-конфигуратора. Если в поле **commonName** содержится строка, отличная от имени хоста, следует удалить данный сертификат нажатием кнопки **Удалить**.

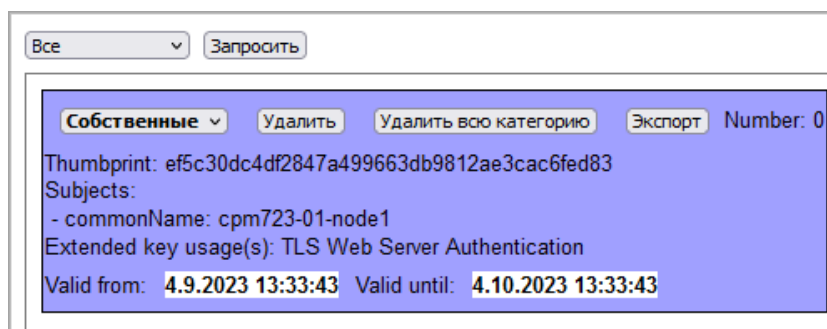


Рисунок 122 – Самоподписанный сертификат CmpFastwelCore для FTPS, SFTP, HTTPS

Собственный сертификат для компоненты CmpFastwelCore генерируется автоматически со сроком действия один месяц при отсутствии непросроченного сертификата для CmpFastwelCore.

Сертификат для CmpFastwelCore предназначен для установления защищенных соединений по протоколам FTSP, SFTP и HTTPS.

Информация о текущих используемых сертификатах будет отображаться на фиолетовом фоне.

2. В выпадающем списке слева от кнопки **Запросить** выбрать категорию сертификатов *Собственные*. Справа от кнопки **Запросить** появятся кнопки **Импорт**, **Генерировать** и выпадающий список с названиями доступных программных компонентов, для которых возможно сгенерировать собственные самоподписанные сертификаты.
3. В выпадающем списке справа от кнопки **Генерировать** выбрать *CmpOPCUAServer* и нажать кнопку **Генерировать**. Над страницей **Сертификаты безопасности** будет отображена заставка, показанная на рисунке 123.



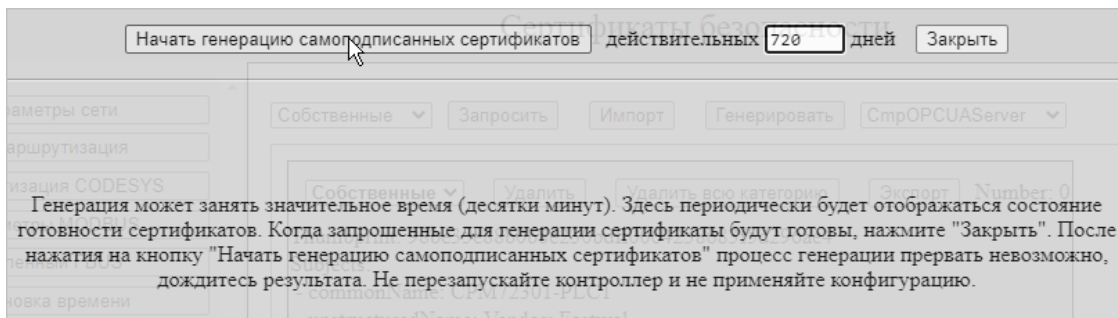



Рисунок 123 – Заставка для генерации одного или нескольких самоподписанных сертификатов безопасности

	<p>Если компонент <i>CmpOPCUAServer</i> отсутствует в выпадающем списке справа от кнопки <b>Генерировать</b>, это означает, что сервер OPC UA контроллера не был ранее активирован на странице <b>Параметры OPC UA</b> в соответствии с указаниями п. 4.7.2.</p>
---	--

- В поле **действительных дней** ввести срок действия сертификата в днях (по умолчанию 365 дней) и нажать **Начать генерацию самоподписанных сертификатов**. Через несколько секунд на заставке появится прямоугольная область со списком доступных программных компонентов и текущим состоянием их сертификатов безопасности, как показано на рисунке 124. Чем большее количество дней указано в поле **действительных дней**, тем больше времени продолжается процесс генерации сертификата.

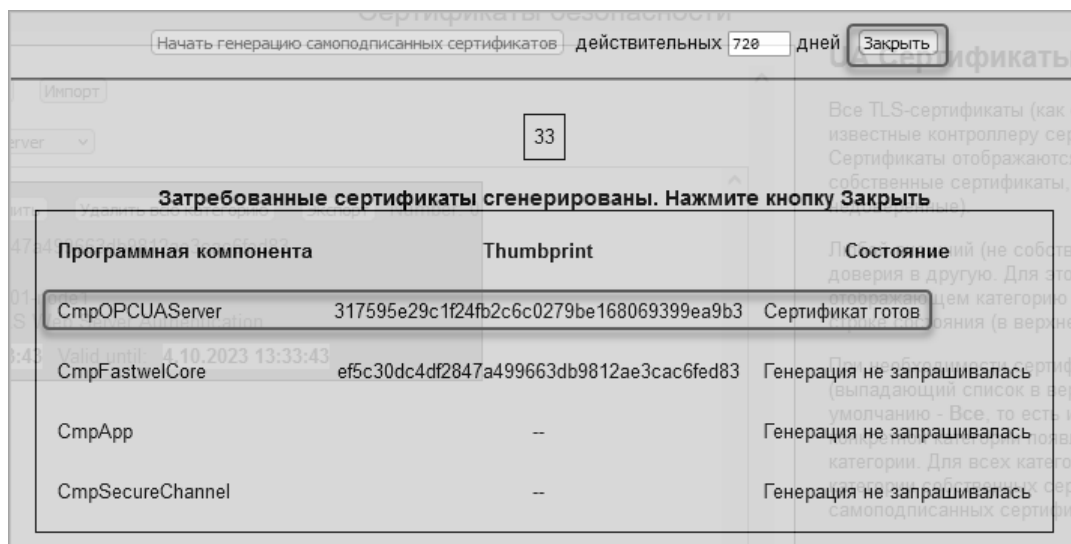



Рисунок 124 – Состояние самоподписанных сертификатов безопасности контроллера

- После появления состояния *Сертификат готов* для всех запрошенных сертификатов, нажать кнопку **Заккрыть**. В списке собственных сертификатов появится только что сгенерированный сертификат сервера OPC UA, как показано на рисунке 125.

	<p>В поле <b>commonName</b> должно отображаться имя компонента в формате <i>OPCUAServer@&lt;имя-хоста&gt;</i>. Начальная и конечная даты действия сертификата отображаются в полях <b>Valid from</b> и <b>Valid until</b> соответственно.</p>
---	---

На странице **Сертификаты безопасности** также возможно просматривать и генерировать самоподписанные сертификаты для работы с защищенным приложением (компонент *CmpApp*) и для использования защищенного подключения к контроллеру из среды разработки IDE МЭК 61131-3 (компонент *CmpSecureChannel*).



В процессе генерации самоподписанных сертификатов не следует выключать питание контроллера или перезапускать контроллер иными способами: кнопкой сброса, командами оболочки ПЛК и т.д.

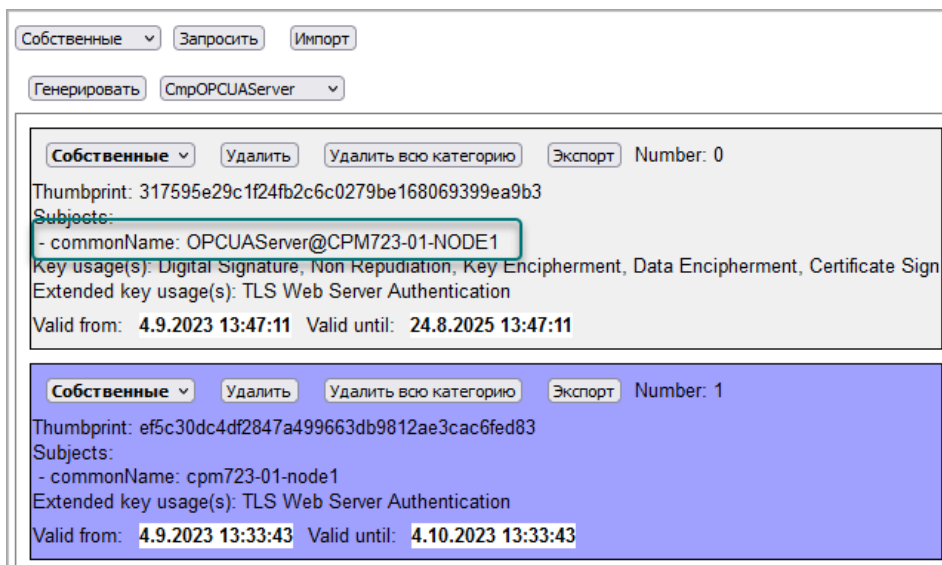


Рисунок 125 – Самоподписанный сертификат безопасности сервера OPC UA

#### 4.7.11. Просмотр и генерация сертификатов безопасности в IDE МЭК 61131-3

Для генерации самоподписанного сертификата безопасности сервера OPC UA в среде разработки IDE МЭК 61131-3 с установленным Fastwel PLC Application Toolkit следует:

1. На вкладке **Установки соединения** редактора устройства, соответствующего контроллеру, нажать **Сканировать сеть**, в окне **Выбор устройства** выбрать контроллер, имеющий установленное ранее имя хоста, и нажать **ОК**. При успешном взаимодействии среды разработки с контроллером на вкладке **Сканировать сеть** будет отображена информация о контроллере, как показано на рисунке 126.

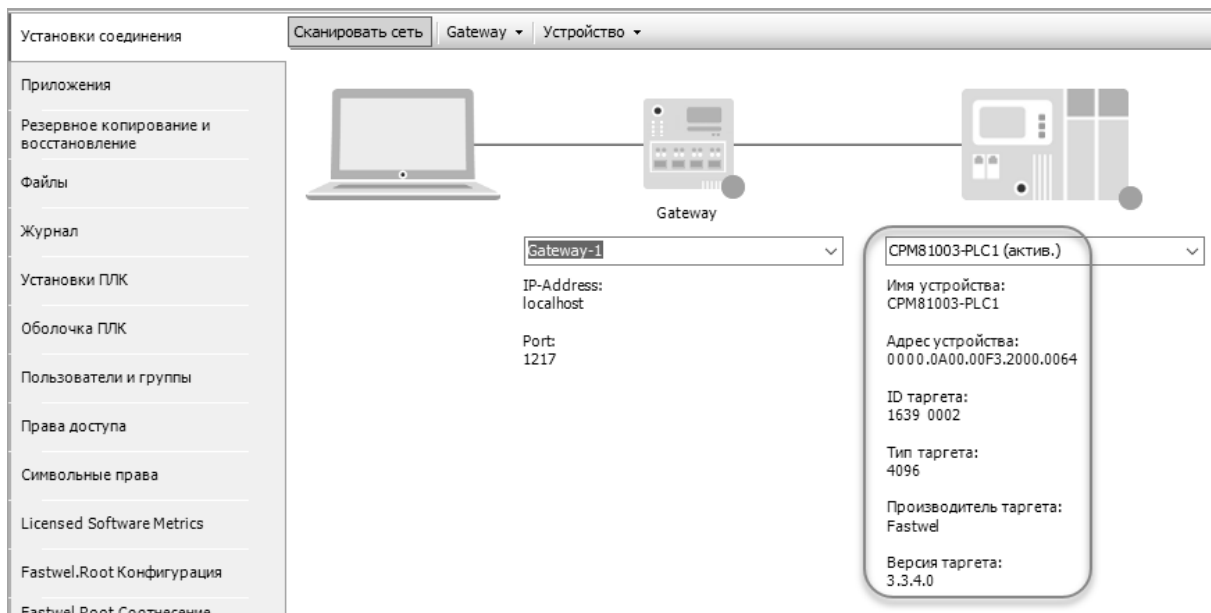



Рисунок 126 – Подготовка к взаимодействию среды разработки с контроллером

2. В главном меню IDE МЭК 61131-3 выполнить команду **Вид – Безопасность**, в появившемся редакторе **Безопасность** выбрать вкладку **Device**, показанную на рисунке 127, после чего нажать кнопку  для чтения информации о сертификатах безопасности контроллера.

3. В древовидном списке *Information* щелкнуть левой кнопкой мыши на корневом элементе, соответствующем требуемому устройству, например, *Device\_1*. Имеющиеся в контроллере сертификаты безопасности программных компонентов будут отображены в таблице сертификатов справа, как показано на рисунке 128. При отсутствии сертификата справа от имени программного компонента будет отображена строка (*not available*).

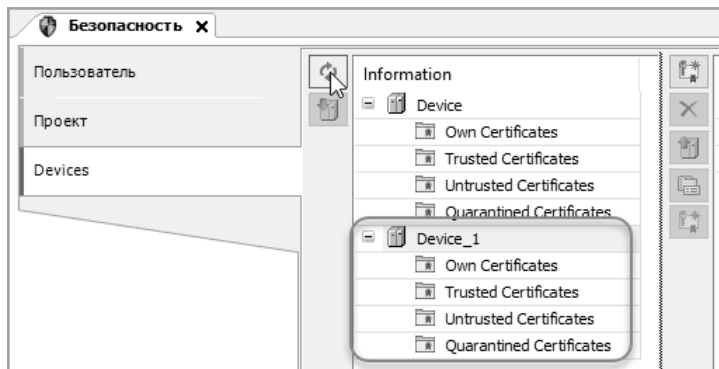



Рисунок 127 – Вкладка Devices редактора Безопасность

Information	Issued for	Issued by	Valid from	Valid until
OPC UA Server (not available)				
Encrypted Application (not available)				
Encrypted Communication	CPM81003-PLC1	CPM81003-PLC1	24.07.2023 15:58:49	23.08.2023 15:58:49 (31 days)

Рисунок 128 – Сертификаты безопасности программных компонентов контроллера

4. Для генерации нового сертификата безопасности для сервера OPC UA контроллера выбрать *OPC UA Server* в таблице сертификатов и нажать кнопку .
5. В появившейся диалоговой панели **Certificate Settings** установить длину ключа шифрования в битах в поле **Key length (bit)** и длительность действия сертификата в днях в поле **Validity period (days)** и нажмите **Ok**. Строка с программным компонентом, для которого запущена генерация сертификата, примет вид, показанный на рисунке 129.


По завершении генерации сертификата в строке с соответствующим программным компонентом появится информация о сертификате, включая: **Issued for** – для какого программного компонента выпущен; **Issued By** – источник выпуска сертификата; **Valid from** и **Valid until** – начальная и конечная даты действия сертификата, **Thumbprint** – бинарный отпечаток сертификата.

Information	Issued for	Iss
OPC UA Server (calculating)		
Encrypted Application (not available)		
Encrypted Communication	CPM81003-PLC1	CPM81003-PLC1

Рисунок 129. Отображение процесса генерации сертификата безопасности для OPC UA Server



В процессе генерации самоподписанных сертификатов не следует выключать питание контроллера или перезапускать контроллер иными способами: кнопкой сброса, командами оболочки ПЛК и т.д.

Кнопка  предназначена для импорта файла цифрового сертификата с компьютера в контроллер для выбранного программного компонента.


Кнопка  используется для экспорта выбранного цифрового сертификата в контроллере в файл на компьютере.

Управление сертификатами в среде разработки и в веб-конфигураторе используют один и тот же механизм в системном программном обеспечении контроллера, поэтому результаты их работы идентичны.

#### 4.7.12. Настройка защищенного сеанса связи в клиентском приложении OPC UA

После разрешения защищенных сеансов связи с сервером OPC UA в системных параметрах контроллера (см. п. 4.7.9) требуется выполнить настройку параметров сеанса связи в клиентском приложении, идентичную выполненной для контроллера.

Для настройки защищенного сеанса в UaExpert в диалоговой панели свойств сеанса в поле **Security Policy** следует установить *Basic256Sha256*, а в поле **Message Security Mode** – *Sign* или *Sign & Encrypt*, как показано на рисунке 130, и нажать **OK**.

	<p>Если в системных параметрах сервера установлен режим безопасности сообщений <i>Sign</i>, то соответствующий параметр сеанса клиента может быть как <i>Sign</i>, так и <i>Sign &amp; Encrypt</i>.</p> <p>Если на сервере установлен режим безопасности сообщений <i>Sign &amp; Encrypt</i>, то соответствующий параметр на клиенте должен быть только <i>Sign &amp; Encrypt</i>.</p>
---	--

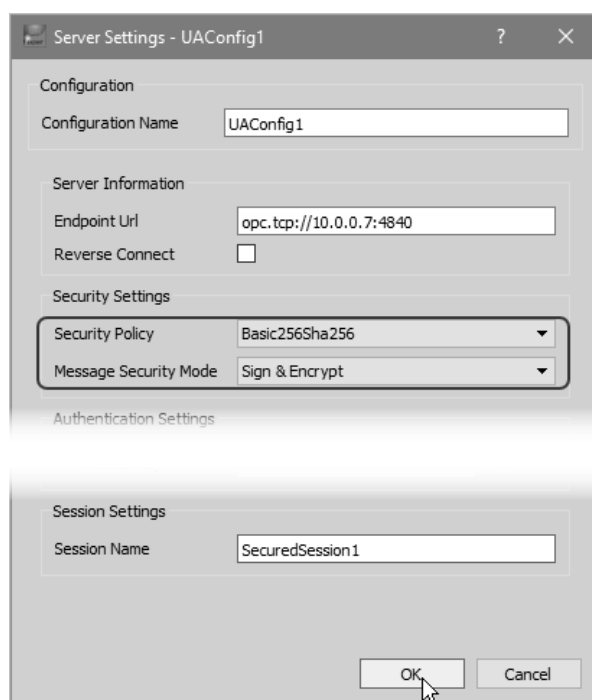


Рисунок 130 – Параметры безопасности сеанса связи с сервером в UaExpert

#### 4.7.13. Обмен сертификатами клиента с сервером и создание сеанса связи через защищенный канал

Перед созданием защищенного канала клиент и сервер OPC UA должны обмениваться цифровыми сертификатами безопасности. Настоящий подраздел содержит указания по обмену сертификатами между клиентом и сервером на примере клиента UaExpert и по созданию сеанса связи клиента с сервером через защищенный канал.

После установки UaExpert версии 1.5.1 и выше при первом запуске на экран выводится сообщение о необходимости генерации сертификата для приложения, показанное на рисунке 131.



Рисунок 131 – Сообщение о необходимости сгенерировать сертификат и ключи при первом запуске UaExpert

При нажатии **ОК** следует в поле **Organization** выведенного на экран окна **New Application Instance Certificate** ввести, как минимум, название предприятия и нажать **ОК**. Впоследствии управление собственными сертификатами и сертификатами серверов OPC UA выполняется в окне **Manage Certificates**, отображаемом по команде **Settings – Manage Certificates** главного меню UaExpert.

После настройки защищенного сеанса связи с сервером в соответствии с указаниями п. 4.7.12:

1. В UaExpert выбрать команду **Connect** в контекстном меню сеанса. На экран будет выведено окно **Certificate Validation**, показанное на рисунке 132, с информацией о полученном от сервера OPC UA цифровом сертификате безопасности, который по умолчанию не является доверенным и помещается в карантин.



Рисунок 132 – Окно UaExpert для проверки и подтверждения подлинности сертификата сервера

2. Нажать кнопку **Trust Server Certificate**, если требуется переместить сертификат сервера в число доверенных (в доверенную зону). Содержимое окна **Certificate Validation** изменится на аналогичное показанному на рисунке 133.

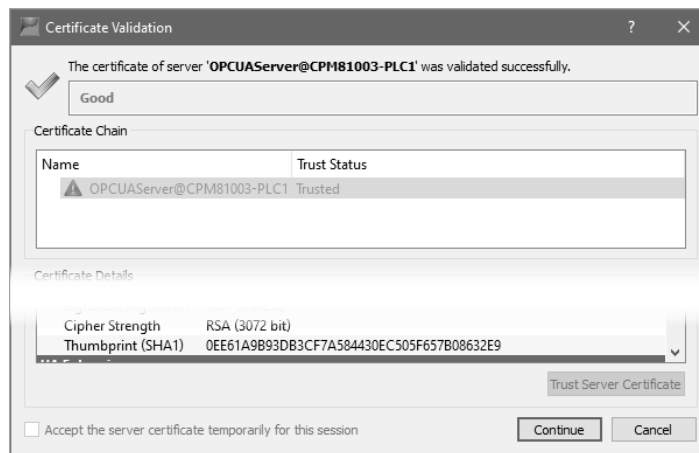


Рисунок 133 – Окно UaExpert после переноса сертификата сервера в число доверенных

3. Нажать **Continue**. Поскольку цифровой сертификат, переданный клиентом серверу, находится на карантине, соединение с сервером не будет установлено, и в области **Log** UaExpert будет выведено сообщение о неудачном завершении создания защищенного канала *"Error 'BadSecurityChecksFailed' was returned during OpenSecureChannel"*.
4. Открыть страницу **Сертификаты безопасности** в веб-конфигураторе контроллера и нажать кнопку **Запросить**. Если цифровой сертификат безопасности клиента был получен сервером, информация о нем будет отображена в списке сертификатов в прямоугольной области желтого цвета, как показано на рисунке 134.

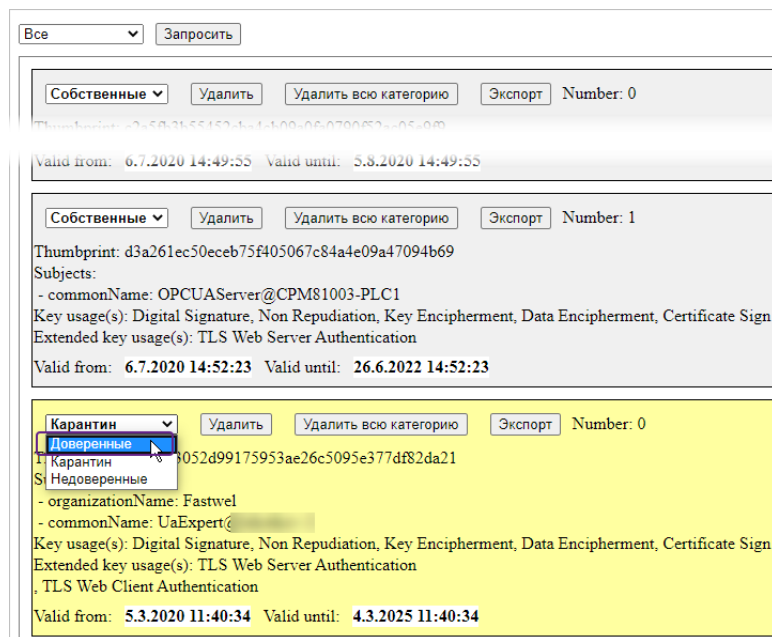


Рисунок 134 – Информация о сертификате UaExpert в карантине сервера

5. Для перемещения сертификата клиента из карантина в число доверенных выбрать **Доверенные** в выпадающем списке в левом верхнем углу прямоугольной области информации о сертификате. При успешном перемещении сертификата в число доверенных прямоугольная область изменит желтый цвет на зеленый, как показано на рисунке 135.

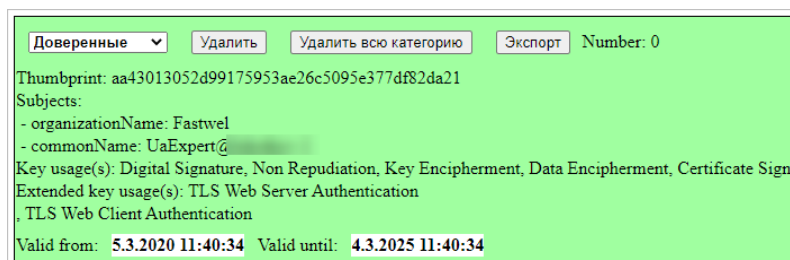


Рисунок 135 – Информация о сертификате UaExpert в доверенной зоне сервера

6. В UaExpert выбрать команду **Connect** в контекстном меню настроенного защищенного сеанса. Если имя хоста, установленное для контроллера, перед первоначальным обменом сертификатами между клиентом и сервером не было сопоставлено с его IP-адресом в сетевой подсистеме операционной системы компьютера, на экран будет выведено сообщение о неправильном имени хоста в сертификате, показанное на рисунке 136.




Рисунок 136 – Информация о сертификате UaExpert в доверенной зоне сервера

7. Нажать кнопку **Ignore**, и сеанс связи клиента с сервером через защищенный канал будет успешно установлен.

Для того, чтобы сообщение о неправильном имени хоста в сертификате не выводилось на экран каждый раз при установлении соединения клиента с сервером, перед первоначальным обменом сертификатами клиента с сервером следует:

1. Добавить сопоставление имени хоста с IP-адресом контроллера в файл `c:\Windows\System32\drivers\etc\hosts`, например:  
**10.0.0.7 CPM81003-PLC1**
2. Затем в поле **Server Information – Endpoint Url**, показанное на рисунке 130, ввести символьный адрес сервера, например:  
**opc.tcp://CPM81003-PLC1:4840**



Перенос сертификата клиента в число доверенных на контроллере может быть выполнен в IDE МЭК 61131-3 на вкладке **Device** редактора **Безопасность** (см. п. 4.7.11). Для этого после обмена сертификатами клиента с сервером нужно на вкладке **Device** выбрать зону (категорию) сертификатов *Quarantined Certificates*, найти в таблице сертификатов только что переданный сертификат клиента и перетащить его мышью в зону *Trusted Certificates*, как показано на рисунке 137.

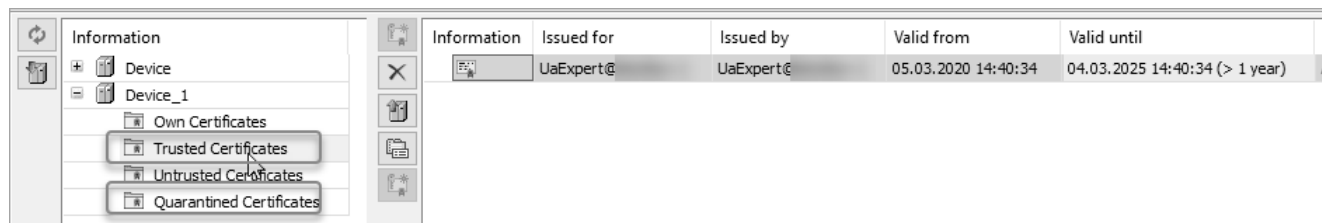


Рисунок 137 – Перенос сертификата UaExpert в доверенную зону сервера в CODESYS V3 на вкладке **Device** редактора **Безопасность**

#### 4.7.14. Развертывание приложения с сохранением возможности безопасного подключения клиентов OPC UA

Развертыванием называется процесс переноса файлов приложения и системных параметров с одного контроллера, называемого эталонным, на другой, называемый целевым, при помощи специального файла `portm.url`, формируемого на эталонном контроллере при помощи команды `saveapp` оболочки ПЛК. Информация о назначении и способах развертывания приложения приведена в п. 5.5 настоящего руководства.

Наиболее очевидным случаем, при котором может потребоваться развертывание приложения, является замена неисправного контроллера на исправный в процессе обслуживания автоматизированной системы управления технологическим процессом. Если в системе для обмена данными между клиентскими приложениями и сервером OPC UA эталонного контроллера используются защищенные сеансы связи (см. п. 4.7.6), то для сохранения возможности создания защищенных сеансов после развертывания приложения на целевом контроллере без повторного обмена сертификатами между клиентами и сервером и переноса сертификатов в доверенную зону целевого контроллера необходимо выполнить следующие дополнительные действия:

1. Перед генерацией сертификата для сервера OPC UA на эталонном контроллере в веб-конфигураторе установить параметр **Имя хоста** на странице **Параметры сети** и нажать **Применить конфигурацию**. Задаваемое имя хоста должно состоять из символов латинского алфавита, цифр от 0 до 9, символов подчеркивания или короткого тире, и иметь длину не более 48 символов.
2. Команда оболочки ПЛК *saveapp* должна выполняться после успешного обмена сертификатами между клиентами и сервером OPC UA эталонного контроллера и переноса сертификатов клиентов в доверенную зону и содержать дополнительный параметр *cert=all*, например:  
**saveapp cert=all**
3. Перед копированием файла *portm.url* с эталонного на целевой контроллер следует убедиться, что системные дата и время на целевом контроллере находятся внутри временного интервала  $[\max(Dmin_1, Dmin_2, \dots, Dmin_N), \min(Dmax_1, Dmax_2, \dots, Dmax_N)]$ , где  $Dmin_i$  – дата и время начала срока действия сертификата с номером  $i$ ,  $Dmax_i$  – дата и время окончания срока действия сертификата с номером  $i$ . Иными словами, системное время на целевом контроллере должно быть не ранее самого позднего времени начала действия и не позднее самого раннего времени окончания действия среди всех переносимых сертификатов.



## 5. Сервисные операции

### 5.1. Общие сведения

Операции по управлению конфигурацией программного обеспечения, служебных данных, файлов и параметров контроллера, выполняемые пользователем и не связанные с разработкой исполняемого кода приложений в IDE МЭК 61131-3, далее называются *сервисными*.

Данный раздел содержит указания по выполнению следующих основных сервисных операций с контроллерами Fastwel в IDE МЭК 61131-3 и иными средствами:

1. Управление съемными дисковыми накопителями, подключаемыми к портам USB или устанавливаемыми в гнездо microSD контроллера.
2. Ограничение доступа к контроллеру по имеющимся сервисным каналам связи, включая связь с IDE МЭК 61131-3, связь по протоколу FTP и по протоколу HTTP с веб-конфигуратором.
3. Передача файлов между контроллером и ПК.
4. Развертывание приложений на нескольких контроллерах.
5. Обновление системного программного обеспечения контроллера и модулей ввода-вывода, входящих в состав контроллера.
6. Оценка загрузки процессора контроллера.

Часть сервисных операций выполняется в **Оболочке ПЛК** путем ввода специальных команд. Вкладка **Оболочка ПЛК** доступна в редакторе целевого устройства в IDE МЭК 61131-3, как показано на рисунке 138, а также на соответствующей странице в веб-конфигураторе контроллера.

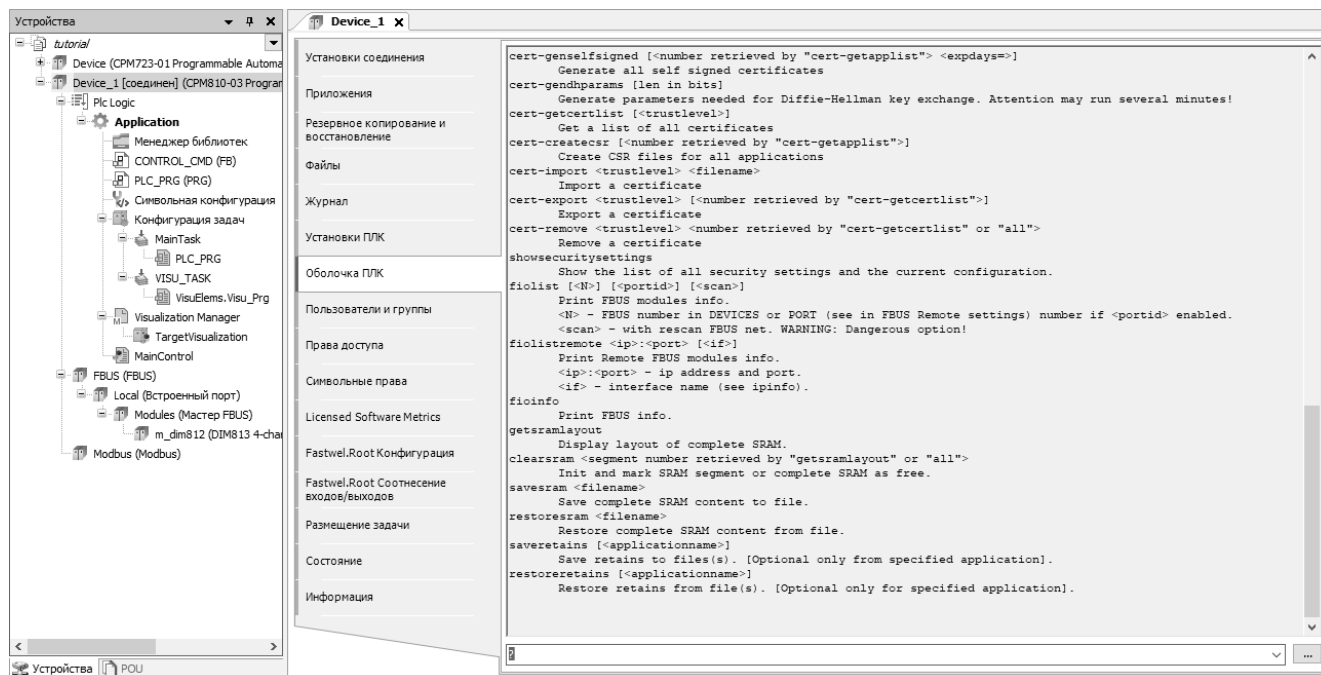


Рисунок 138 – Вкладка Оболочка ПЛК IDE МЭК 61131-3 после выполнения команды ?

### 5.2. Управление съемными дисковыми накопителями

#### 5.2.1. Общие сведения

Контроллеры Fastwel некоторых типов (CPM723, CPM810, CPM823) имеют в своем составе гнезда, к которым допускается подключать и отключать дисковые накопителя без выключения питания контроллера. Накопители с интерфейсом USB или microSD далее называются съемными накопителями.

Настоящий подраздел содержит информацию о просмотре списка подключенных дисковых накопителей и о программном завершении работы с дисковым накопителем перед физическим отключением от контроллера.



Для контроллера CPM810-03 не допускается установка и извлечение карты microSD при включенном питании контроллера.

Перед подключением и использованием съемных накопителей к портам интерфейса USB контроллера CPM810-03 в веб-конфигураторе контроллера должен быть снят флажок **Отключение USB-устройств** на странице **Система** с последующим нажатием кнопки **Применить конфигурацию**.

### 5.2.2. Просмотр информации о подключенных съемных накопителях

Для получения списка всех съемных накопителей, подключенных к контроллеру, выполните команду *disks* на странице **Оболочка ПЛК** веб-конфигуратора или на вкладке **Оболочка ПЛК** в редакторе целевого устройства IDE МЭК 61131-3.

При наличии подключенных к контроллеру съемных накопителей ответ на команду будет выглядеть аналогично следующему:

```
=>0: [/dev/sdb] removable
    0: /dev/sdb1 -> ./user/usb2p1
1: [/dev/sdc] removable
    0: /dev/sdc1 -> ./user/usb1p1
2: [/dev/sdd] removable
    0: /dev/sdd5 -> ./user/usb3p5
    1: /dev/sdd1 -> ./user/usb3p1
```

или

*disks*

```
0: [/dev/mmcblk] removable
    0: /dev/mmcblk0p1 -> ./user/microsd1
    1: /dev/mmcblk0p1 -> ./sd-mmcblk0p1
```

Каждый набор строк, начинающийся с порядкового номера накопителя (начиная с 0, в порядке обнаружения накопителей операционной системой), описывает съемный накопитель и все найденные на нем разделы, которые монтируются в каталог *./user* с именами в формате:

**<тип накопителя><номер накопителя>р<номер устройства, соответствующего разделу накопителя>**

Например, набор строк:

```
=>0: [/dev/sdb] removable
    0: /dev/sdb1 -> ./user/usb2p1
```

описывает накопитель с номером 0 и системным именем */dev/sdb*, на котором имеется один раздел с системным именем */dev/sdb1*, который смонтирован в подкаталог *usb2p1* каталога *./user*.

Для USB-накопителя с двумя разделами, который, наряду с двумя другими накопителями в гнездах USB1 и USB2, установлен в гнездо USB3, информация, выводимая по команде *disks* для данного накопителя, выглядит аналогично следующей:

```
2: [/dev/sdd] removable
    0: /dev/sdd5 -> ./user/usb3p5
    1: /dev/sdd1 -> ./user/usb3p1
```

Для получения информации о съемных накопителях в приложении контроллера следует воспользоваться функциями *SysBoardDeviceGetAllDisks*, *SysBoardDeviceGetAllMountPoints* и *SysBoardDeviceGetDiskInfo* системной библиотеки *FastwelRemovableMedia* (см. п. 6.7), устанавливаемой при установке *Fastwel PLC Application Toolkit*.

### 5.2.3. Извлечение съемных дисковых накопителей

Перед физическим извлечением съемного накопителя из гнезда следует завершить все операции с ним и выполнить команду программного извлечения с размонтированием всех точек монтирования:

```
eject [номер|all]
```

где

<номер> – порядковый номер накопителя, начиная с 0 (в порядке обнаружения операционной системой съемных дисковых устройств), расположенный слева от имени устройства при выполнении команды `disks`;

*all* – данный параметр позволяет подготовить к физическому извлечению все подключенные съемные накопители.

Например:

1. Вывести информацию обо всех съемных дисковых накопителях:  

```
disks
=>0: [/dev/sdb] removable
   0: /dev/sdb5 -> ./user/usb3p5
   1: /dev/sdb1 -> ./user/usb3p1
1: [/dev/sdc] removable
   0: /dev/sdc1 -> ./user/usb2p1
2: [/dev/sdd] removable
   0: /dev/sdd1 -> ./user/usb1p1
```
2. Выполнить программное отключение диска 1 (/dev/sdc)  

```
eject 1
=>Done, disk /dev/sdc successfully ejected!
```
3. Еще раз вывести информация обо всех съемных дисковых накопителях:  

```
=>0: [/dev/sdb] removable
   0: /dev/sdb5 -> ./user/usb3p5
   1: /dev/sdb1 -> ./user/usb3p1
1: [/dev/sdd] removable
   0: /dev/sdd1 -> ./user/usb1p1
```

При отключении съемного накопителя командой *eject* происходит смена номеров накопителей, которые до отключения имели порядковые номера, превосходящие по значению переданный команде *eject*.

Для отключения съемного накопителя из приложения IDE МЭК 61131-3, загруженного в контроллер, следует воспользоваться функцией `SysBoardDeviceEject` системной библиотеки `FastwelRemovableMedia`, описание которой приведено в п. 6.7 настоящего руководства.

### 5.3. Управление доступом

#### 5.3.1. Учетные записи пользователей


Для управления доступом к контроллеру из IDE МЭК 61131-3 через коммуникационный сервис IDE Gateway, а также с использованием других сетевых протоколов, поддерживаемых системным программным обеспечением контроллера, используются две встроенные учетные записи:

1. *Administrator* – для данной учетной записи доступны следующие операции:  
соединение с контроллером из среды разработки,  
настройка системных параметров контроллера в веб-конфигураторе,  
загрузка в контроллер файлов обновления системного программного обеспечения и файла развертывания приложения из веб-конфигуратора,  
доступ к корневому каталогу системы исполнения по протоколу FTP.
2. *Everyone* – для данной учетной записи доступны следующие операции:  
соединение с контроллером из среды разработки,  
просмотр системных параметров в веб-конфигураторе,  
доступ к каталогу пользователя по протоколу FTP.

По умолчанию для учетной записи *Administrator* установлен пароль *Administrator*, а для учетной записи *Everyone* паролем является пустая строка.



Если для учетной записи *Everyone* установить непустой пароль, то при попытке подключения к контроллеру из IDE МЭК 61131-3 на экран монитора будет выводиться запрос ввода имени пользователя и пароля, показанный на рисунке 139.

	<p>При вводе непустого пароля для любой учетной записи следует использовать только символы таблицы ASCII. Запрещается использовать символы кириллицы и других национальных кодировок.</p> <p>Утраченный пароль может быть сброшен только сервисной службой предприятия-изготовителя.</p>
---	--

Возврат исходных паролей встроенных учетных записей возможен путем выполнения команды **Онлайн – Сброс заводской** из IDE МЭК 61131-3 после успешного подключения к контроллеру с правами одной из учетных записей, имеющих возможность выполнять заводской сброс.

Кроме того, возврат исходных паролей происходит при выполнении команды **Восстановить заводские настройки** на странице **Система** в веб-конфигураторе контроллера.

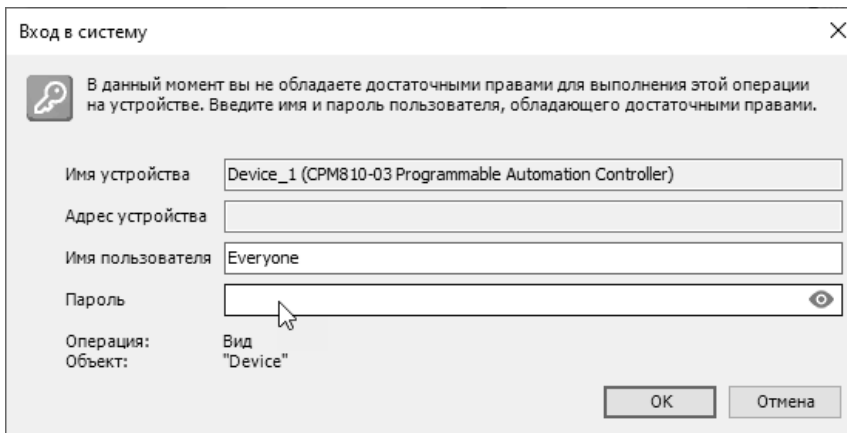



Рисунок 139 – Запрос IDE МЭК 61131-3 о вводе пароля при подключении к контроллеру

Дополнительные учетные записи для доступа к контроллеру могут быть добавлены в IDE МЭК 61131-3 на вкладке **Пользователи и группы** редактора целевого устройства и переданы в контроллер. Дополнительные учетные записи не будут иметь прав доступа по поддерживаемым контроллером сетевым протоколам, отличным от IDE Gateway: HTTP, FTP и т.д.


	<p>Перед созданием дополнительных учетных записей в контроллере следует выгрузить из контроллера имеющиеся встроенные учетные записи в соответствии с указаниями п. 5.3.2.</p> <p>Не рекомендуется использовать команду <b>Онлайн – Безопасность – Добавить пользователя устройства</b> во избежание вероятной потери возможности связаться с контроллером из IDE МЭК 61131-3!</p>
---	--

Информация о создании учетных записей и групп учетных записей в проекте для ограничения прав доступа к операциям над элементами проекта из IDE МЭК 61131-3 приведена в разделе **Защита и сохранение проекта** интерактивной справочной системы IDE МЭК 61131-3.

Информация о глобальном пароле, который может быть установлен для текущего проекта с целью защиты от несанкционированного доступа к исходному тексту приложений в проекте, приведена в п. 3.7.4 настоящего документа.


### 5.3.2. Изменение пароля учетной записи из IDE МЭК 61131-3

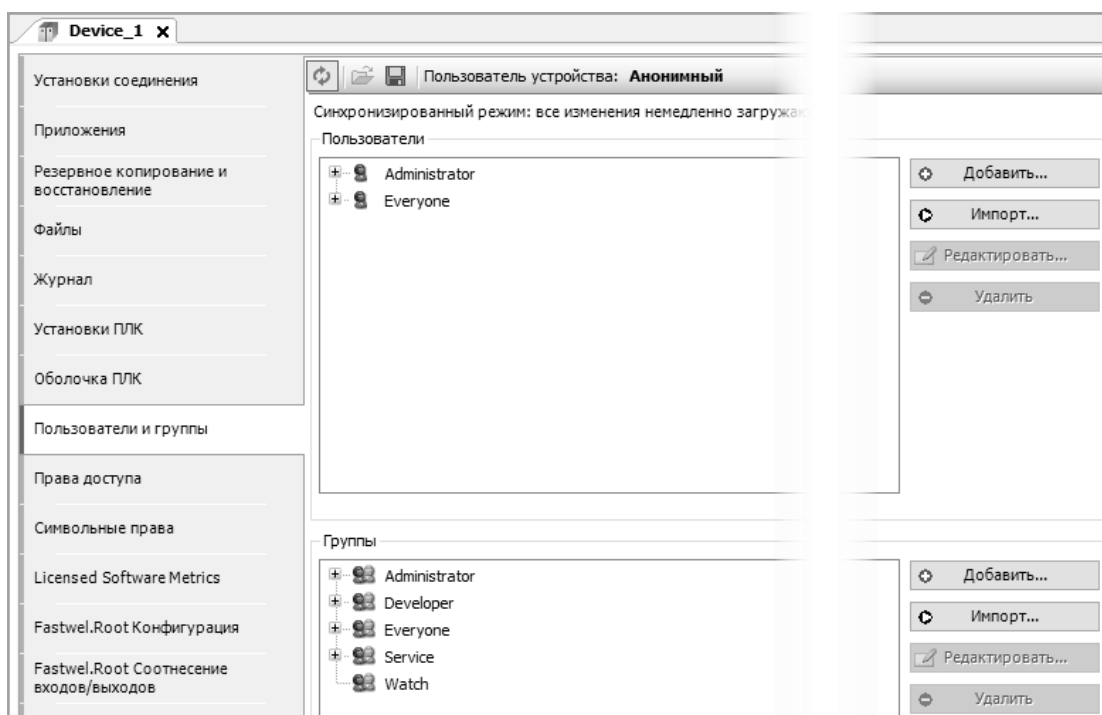
Для изменения пароля учетной записи в контроллере из IDE МЭК 61131-3 следует:

1. Настроить сетевой путь доступа среды разработки к контроллеру в соответствии с указаниями п. 3.5.2 (через сервисный порт USB или RS-232C) или п. 3.5.3 (по протоколу TCP или UDP). При наличии пути доступа к контроллеру вкладка **Установки соединения** редактора устройств выглядит, как показано на рисунке 44 или на рисунке 47.
2. Перейти на вкладку **Пользователи и группы** редактора целевого устройства, показанную на рисунке 140, и нажать кнопку  в панели инструментов вкладки. При наличии связи с контроллером в списке **Пользователи** появятся два элемента,

*Administrator* и *Everyone*, при этом кнопка  будет обведена контуром темного цвета, как показано на рисунке 140.

В данный момент имеется возможность добавления новых учетных записей в базу данных безопасности контроллера с вкладки **Пользователи и группы**, изменения паролей имеющихся учетных записей, а также отнесения учетных записей пользователей к группам учетных записей в списке **Группы** или исключения учетных записей из групп.


	<p><i>Группой</i> является именованная совокупность учетных записей пользователей, которым установлены <i>права доступа (разрешения, permissions)</i> для выполнения следующих основных операций над объектами конфигурации приложения: <i>Добавить/Удалить (Add/Remove)</i>, <i>Изменить (Modify)</i>, <i>Обзор (View)</i>, <i>Выполнить (Execute)</i>.</p> <p>Встроенные учетные записи <i>Administrator</i> и <i>Everyone</i> по умолчанию отнесены к группам <i>Administrator</i> и <i>Everyone</i>.</p> <p>Учетные записи пользователей, входящих в группу <i>Administrator</i>, имеют разрешения на выполнение всех операций над всеми объектами конфигурации приложения.</p> <p>Учетная запись <i>Everyone</i> по умолчанию имеет пустой пароль, в результате чего изначально анонимному пользователю разрешено выполнять любые действия с объектами конфигурации приложения при соединении с контроллером по протоколу IDE Gateway.</p> <p>Для того, чтобы при попытке соединения с контроллером выводился запрос на ввод учетных данных (имени пользователя и пароля), следует установить непустой пароль для учетной записи <i>Everyone</i>. Если при этом требуется ограничить доступ к контроллеру, следует, как минимум, изменить исходный пароль учетной записи <i>Administrator</i>.</p> <p>Если требуется ограничить права доступа к объектам конфигурации приложения контроллера для учетной записи <i>Everyone</i>, необходимо удалить данную учетную запись из группы <i>Administrator</i>.</p>
---	--




**Рисунок 140 – Информация об учетных записях пользователей в контроллере**

- Для смены пароля учетной записи дважды щелкнуть над именем учетной записи или выбрать имя учетной записи и нажать кнопку **Редактировать**, расположенную справа от списка пользователей.
- В появившейся диалоговой панели **Редактировать пользователя <имя пользователя>**, показанной на рисунке 141, ввести новый пароль в поля **Пароль** и **Повторите пароль**.

Кнопка **ОК** будет разблокирована, если в полях **Пароль** и **Повторите пароль** введены одинаковые строки пароля.

	<p>При вводе непустого пароля следует использовать только символы таблицы ASCII. Запрещается использовать символы кириллицы и других национальных кодировок.</p>
---	--

Для передачи изменений в контроллер следует нажать **ОК**.

5. Щелкнуть на кнопке  для прекращения соединения с контроллером.

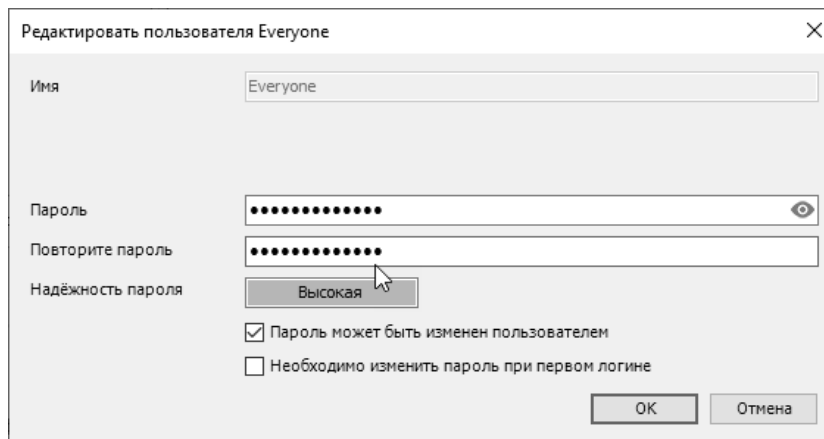


Рисунок 141 – Изменение пароля учетной записи в среде разработки

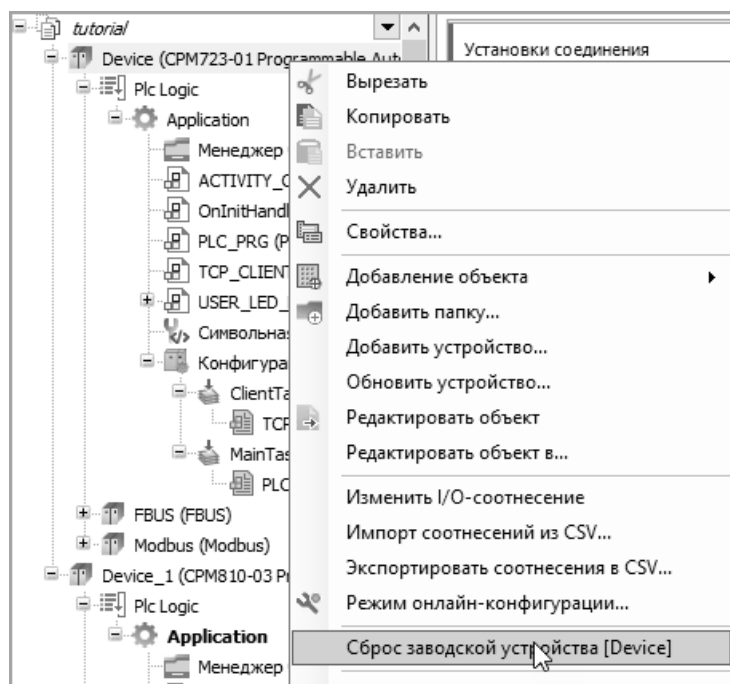




Рисунок 142 – Команда Сброс заводской устройства для целевого устройства

	<p>Для соединения с контроллером с реквизитами требуемой учетной записи пользователя следует сначала выполнить команду <b>Онлайн – Безопасность – Отключить текущего онлайн пользователя</b>, а затем попытаться повторно установить соединение с контроллером.</p>
	<p>Сброс пароля встроенных учетных записей <i>Administrator</i> и/или <i>Everyone</i> может быть выполнен на странице <b>Установка паролей</b> в веб-конфигураторе контроллера.</p> <p>Для возврата конфигурации учетных записей в исходное состояние следует выполнить команду <b>Сброс заводской устройства</b> в контекстном меню целевого устройства, как показано на рисунке 141, и отметить флажок <b>Delete</b> в диалоговой панели запроса подтверждения сброса, показанной на рисунке 142.</p>

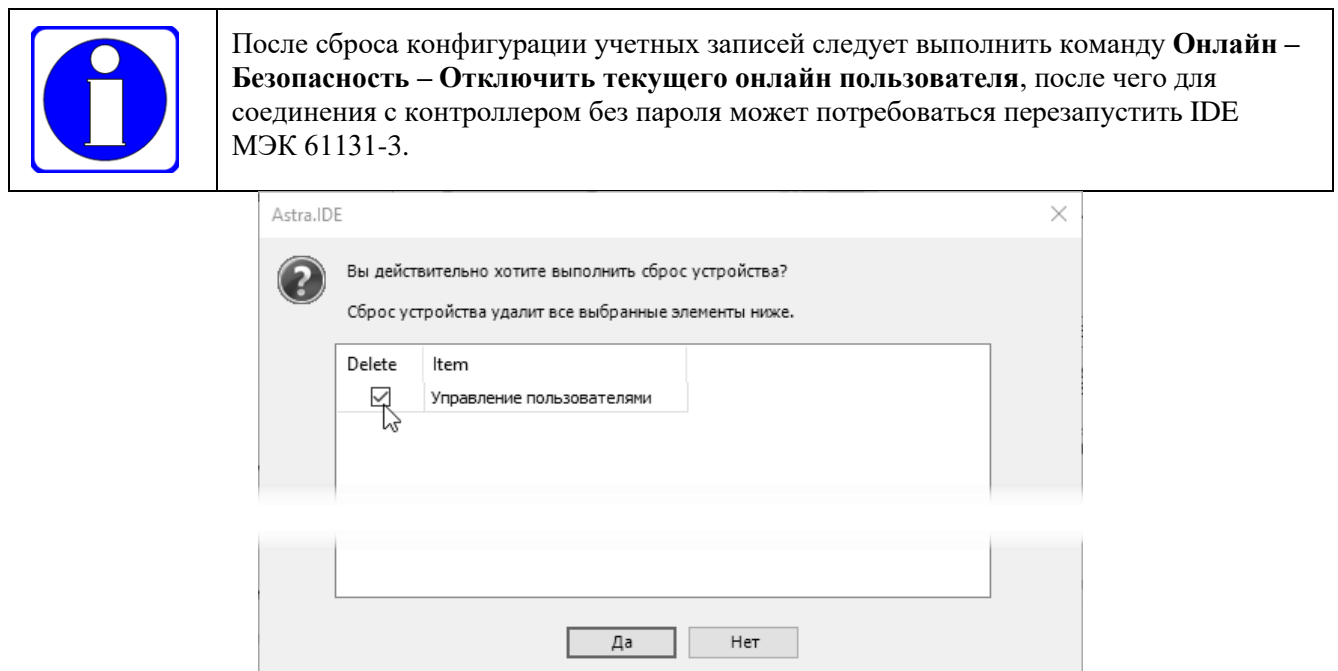
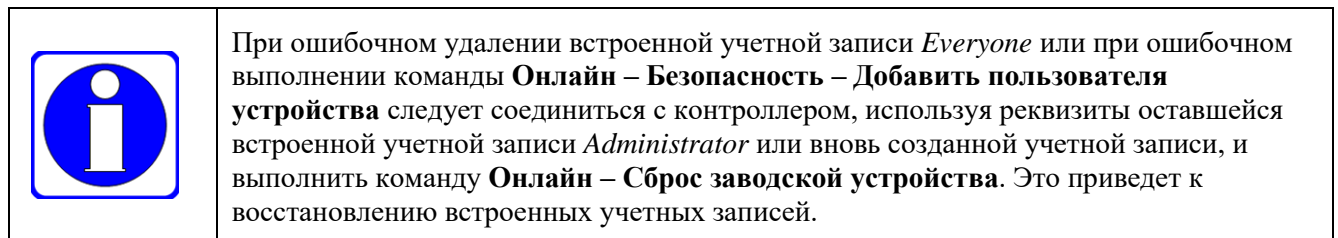


Рисунок 143 – Подтверждение заводского сброса устройства с удалением изменений в учетных записях пользователей



При ошибочном удалении встроенной учетной записи *Everyone* или при ошибочном выполнении команды **Онлайн – Безопасность – Добавить пользователя устройства** следует соединиться с контроллером, используя реквизиты оставшейся встроенной учетной записи *Administrator* или вновь созданной учетной записи, и выполнить команду **Онлайн – Сброс заводской устройства**. Это приведет к восстановлению встроенных учетных записей.

### 5.3.3. Ограничение прав доступа для учетных записей пользователей

В IDE МЭК 61131-3 обеспечивается возможность ограничения доступа к просмотру системной информации, изменению приложения и параметров приложения контроллера для групп учетных записей пользователей на вкладке **Права доступа** редактора устройства, показанной на рисунке 144.

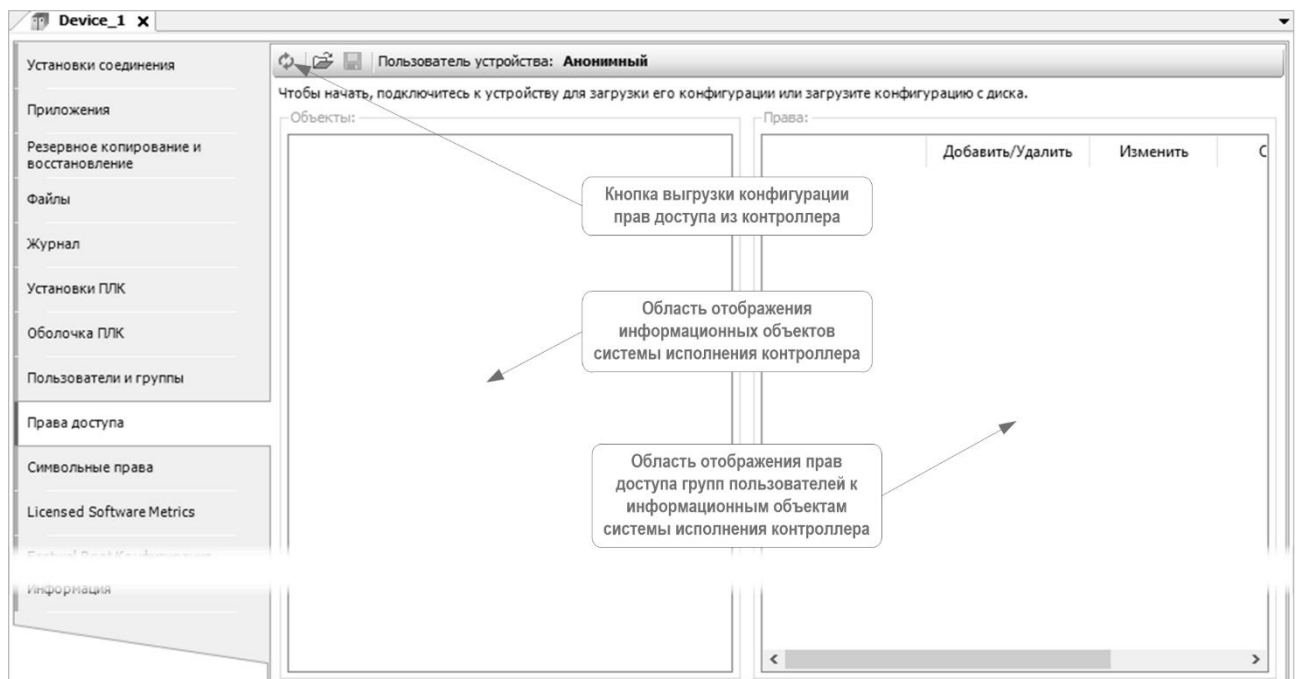


Рисунок 144 – Редактор прав доступа

Например, анонимному пользователю или для вновь созданных учетных записей пользователей может быть запрещено просматривать записи системного журнала контроллера на вкладке **Журнал**





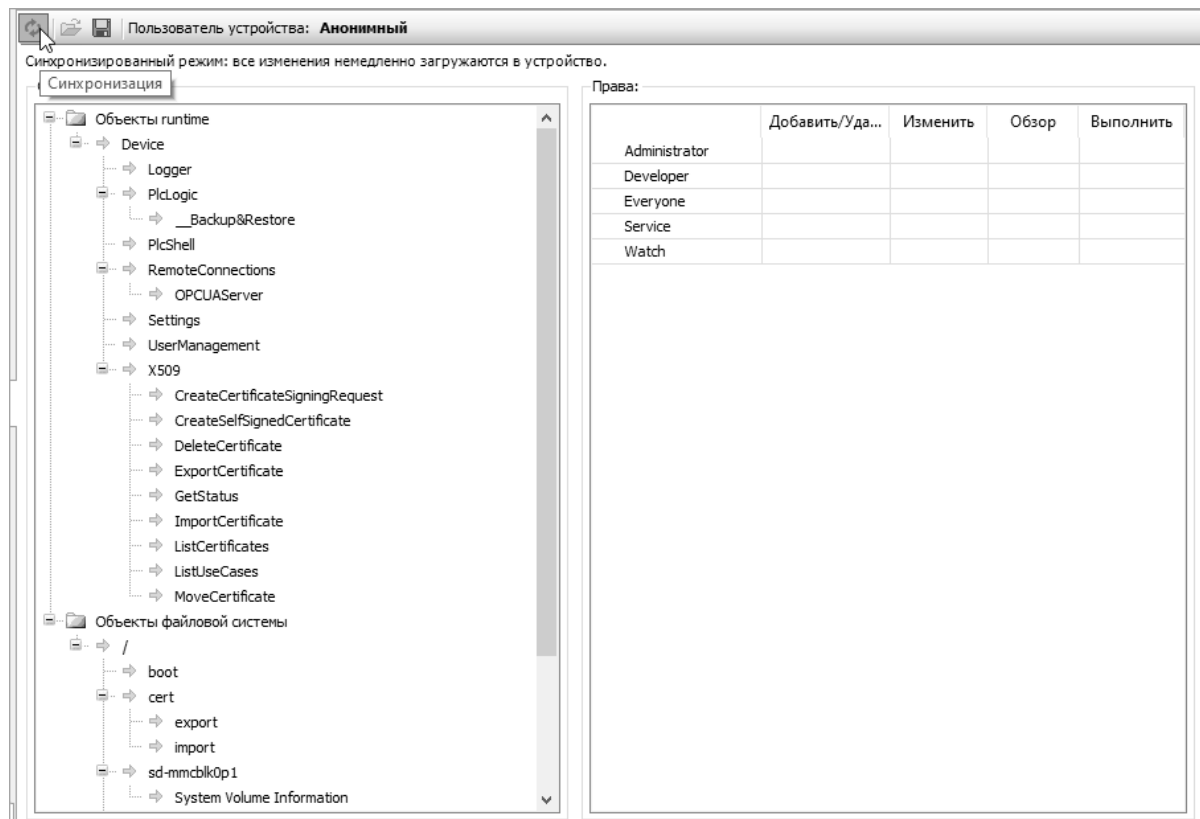


Рисунок 146 – Объекты системы исполнения контроллера с возможностью ограничения доступа

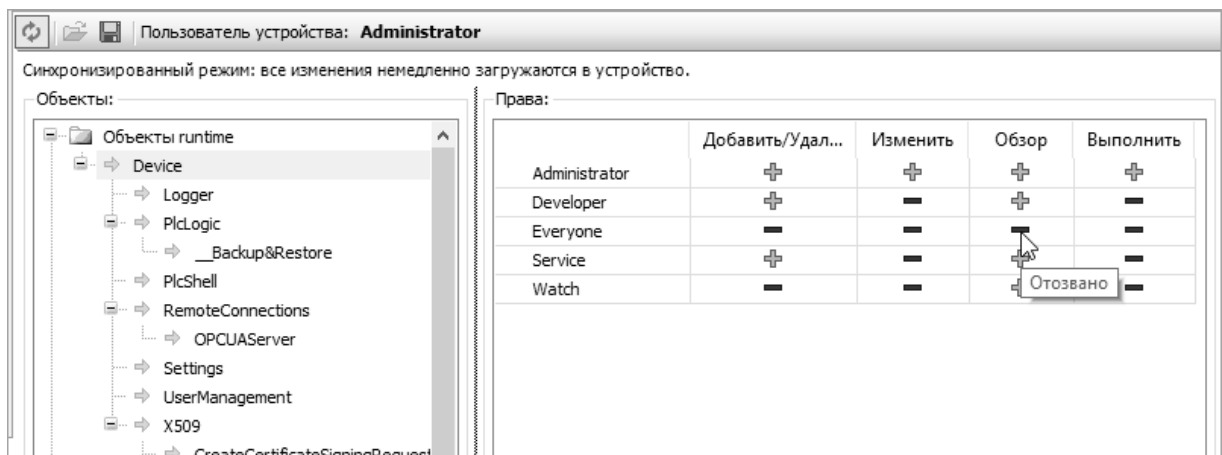






Рисунок 147 – Текущие права доступа к объектам системы исполнения контроллера

10. Для завершения редактирования прав доступа к объектам для учетных записей щелкнуть на кнопке , что приведет к закрытию соединения с контроллером.
11. Для соединения с контроллером с реквизитами учетной записи *Everyone* выполнить команду **Онлайн – Безопасность – Отключить текущего онлайн пользователя**, а затем перейти на вкладку **Установка соединения** и повторно установить соединение с контроллером.
12. В выведенном на экран окне **Вход в систему** в качестве имени пользователя и пароля ввести реквизиты учетной записи *Everyone*. Соединение с контроллером не будет установлено, на экран будет выведено сообщение **Никакое устройство не отвечает на запрос поиска**.

Для восстановления прав доступа к объектам системы исполнения следует установить соединение с контроллером, используя реквизиты учетной записи *Administrator* и на вкладке **Права доступа** редактора устройств для объекта *Объекты runtime–Device* вернуть права доступа для группы *Everyone*, щелкая в ячейках операций доступа до появления символов .

	<p>При необходимости предоставления или ограничения доступа к отдельным объектам конфигурации приложения следует изменять разрешения для объектов из нижней части древовидного списка <i>Device</i> и для каталогов файловой системы контроллера.</p>
	<p>Если при попытке соединения с контроллером на вкладке <b>Установки соединения</b> вместо диалоговой панели <b>Вход в систему</b> выводится сообщение <b>Никакое устройство не отвечает на запрос поиска</b>, следует подождать 20 – 30 с или закрыть и повторно открыть редактор целевого устройства и повторить попытку.</p>

## 5.4. Чтение и запись файлов

### 5.4.1. Способы передачи файлов между контроллером и компьютером

Имеются следующие способы передачи файлов между контроллером и ПК:

1. Загрузка и выгрузка файлов по протоколу FTP или FTPS при помощи клиента FTP или FTPS. В качестве клиента FTP могут использоваться проводник Windows, различные графические утилиты управления файлами вроде Total Commander, а также утилиты командной строки lftp, wget, wput и другие, входящие в состав Cygwin, или в состав операционной системы Linux, установленной на ПК.  
В качестве клиента FTPS могут использоваться графические утилиты управления файлами Total Commander, FileZilla и т.п., а также утилита командной строки lftp, которая может быть установлена в операционной системе Linux или включена в состав Cygwin.  
Для обеспечения возможности обмена файлами по протоколу FTP и FTPS в веб-конфигураторе контроллера на странице **Параметры FTP** должен быть снят флажок **Запретить FTP** и установлен флажок **Использовать SSL**.
2. Загрузка и выгрузка файлов по протоколу SFTP. В качестве клиента SFTP может использоваться графическая утилита управления файлами FileZilla, а также утилита командной строки sftp, которая может быть установлена в операционной системе Linux или включена в состав Cygwin.  
Для обеспечения возможности обмена файлами по протоколу SFTP в веб-конфигураторе контроллера на странице **Параметры FTP** должен установлен флажок **Разрешить SFTP**.
3. Загрузка и выгрузка файлов при помощи IDE МЭК 61131-3 на вкладке **Файлы** редактора целевого устройства.
4. Загрузка системных файлов обновления системного программного обеспечения контроллера и периферийных модулей со страницы **Система** веб-конфигуратора контроллера.

### 5.4.2. Передача файлов по протоколу FTP

#### 5.4.2.1. Передача файлов проводником Windows

Для подключения к контроллеру по протоколу FTP проводником Windows в адресной строке проводника следует ввести:

`ftp://<username>:<password>@<ip-address>`

где `<username>:<password>` – имя пользователя и пароль учетной записи, под которой устанавливается соединение по протоколу FTP,

`<ip-address>` – IP-адрес контроллера,

а затем нажать Enter.

Например, для подключения к контроллеру, имеющему IP-адрес 10.0.0.3, с именем пользователя *Administrator* и паролем *Administrator* следует ввести:

`ftp://Administrator:Administrator@10.0.0.3`

При успешном подключении к контроллеру с именем пользователя *Administrator* в окне клиента FTP будет отображен корневой каталог системы исполнения, показанный на рисунке 148.

Для подключения к контроллеру с IP-адресом 10.0.0.3 под именем *Everyone* и пустым паролем следует ввести:

```
ftp://Everyone:@10.0.0.3
```

При успешном подключении к контроллеру с именем пользователя *Everyone* в окне клиента FTP будет отображен каталог пользователя *user*, содержащий файлы и подкаталоги, созданные из приложения функциями библиотеки SysFile, а также каталоги доступа к подключенным съемным дисковым накопителям.

После успешного подключения проводником для загрузки файла из контроллера следует перейти в требуемый подкаталог файловой системы контроллера и перетащить мышью нужный файл в целевой каталог ПК.

Для загрузки (передачи) файла в контроллер следует перетащить нужный файл из каталога файловой системы ПК в требуемый целевой каталог файловой системы контроллера.

#### 5.4.2.2. Передача файлов утилитами командной строки Cygwin

Утилита командной строки *ftp*, входящая в состав Windows, не может использоваться для чтения и записи файлов по протоколу FTP, поскольку не поддерживает так называемый пассивный режим, в котором при взаимодействии клиента FTP с сервером инициатором операций является клиент.

В связи с этим для автоматизации передачи файлов между контроллером и ПК по протоколу FTP на ПК с операционной системой Windows может быть установлен пакет Cygwin с поддержкой командной оболочки *bash* (или аналогичной), а также соответствующих утилит, выполняющих функции клиента FTP. В настоящем подразделе вкратце рассматривается использование утилиты загрузки файлов *wget* и утилиты загрузки файлов *wput*.



Если в Windows добавить путь к каталогу *bin*, содержащему утилиты Cygwin, в системную переменную окружения *Path*, то запуск утилит Cygwin возможен из командной оболочки Windows *cmd.exe*.

Для загрузки файла, расположенного в каталоге файловой системы контроллера, в целевой каталог ПК может использоваться команда:

```
wget ftp://<username>:<password>@<ip-address>/<file-path> -P <pc-path>
```

где *<username>:<password>* – имя пользователя и пароль учетной записи, с правами которой устанавливается соединение по протоколу FTP,

*<ip-address>* – IP-адрес контроллера,

*<file-path>* – относительный путь и имя файла в контроллере,

*<pc-path>* – путь к каталогу ПК, в который будет помещен выгружаемый файл.

#### Пример 1:

Пусть в контроллере не менялись исходные пароли учетных записей *Administrator* и *Everyone*, и из контроллера с IP-адресом 10.0.0.3 требуется выгрузить файл *safedump.txt* из корневого каталога системы исполнения и все файлы с расширением *\*.bin* из каталога пользователя в папку *d:\local* на ПК.

Для загрузки *safedump.txt* следует использовать следующую команду, запускаемую в окне Cygwin:

```
wget ftp://Administrator:Administrator@10.0.0.3/safedump.txt -P /cygdrive/d/local/
```

Если путь к каталогу *bin* пакета Cygwin добавлен в системную переменную окружения *Path*, то можно в окне *cmd.exe* выполнить следующую команду:

```
wget ftp://Administrator:Administrator@10.0.0.3/safedump.txt -P d:\local\
```

Для загрузки всех файлов с расширением *\*.bin* можно использовать команду в окне Cygwin:

```
wget -r ftp://Everyone:@10.0.0.3/*.bin -P /cygdrive/d/local/
```

или в окне *cmd.exe*:

```
wget -r ftp://Everyone:@10.0.0.3/*.bin -P d:\local\
```

### Пример 2:

Пусть в контроллере не менялись исходные пароли учетных записей *Administrator* и *Everyone*, и требуется выгрузить все содержимое каталога пользователя контроллера с IP-адресом 10.0.0.3 в папку d:\local на ПК.

Если глубина вложенности файлов и подкаталогов в каталоге пользователя не превышает пяти, для выгрузки можно использовать следующую команду в окне Cygwin:

```
wget -r ftp://Everyone:@10.0.0.3/* -P /cygdrive/d/local/
```

В результате в каталоге d:\local на ПК будет создан подкаталог 10.0.0.3 со всем содержимым каталога пользователя контроллера с адресом 10.0.0.3.

Для загрузки файла, в целевой каталог на контроллера может использоваться команда:

```
wput --basename=<pc-path> <pc-path>/<file> ftp://<username>:<password>@<ip-address>/<rem-path>
```

где <username>:<password> – имя пользователя и пароль учетной записи, с правами которой устанавливается соединение по протоколу FTP,

<ip-address> – IP-адрес контроллера,

<pc-path> – путь к подлежащему передаче файлу на ПК,

<file> – имя файла на ПК,

<rem-path> – относительный путь в файловой системе на контроллере.

### Пример 3:

Пусть пароль учетной записи *Administrator* был изменен на *Admin*, и требуется передать файл обновления системного программного обеспечения *norm.dnl* в корневой каталог системы исполнения контроллера из каталога d:\tmp ПК. Для загрузки файла можно использовать команду в окне Cygwin:

```
wput --basename=/cygdrive/d/tmp/ /cygdrive/d/tmp/norm.dnl ftp://Administrator:Admin@10.0.0.3/
```

или в окне cmd.exe:

```
wput --basename=d:\tmp\ d:\tmp\norm.dnl ftp://Administrator:Admin@10.0.0.3/
```

### 5.4.2.3. Передача файлов по протоколу SFTP

Для обеспечения возможности передачи файлов между ПК и контроллером по протоколу SFTP в веб-конфигураторе контроллера на странице **Параметры FTP** следует установить флажок **Разрешить SFTP**, нажать кнопку **Применить конфигурацию** и убедиться в успешном перезапуске контроллера.



Встроенная учетная запись *Everyone* по умолчанию имеет пустой пароль. В связи с этим доступ к каталогу user контроллера по протоколу FTP, SFTP и FTPS для пользователя *Everyone* возможен без пароля.

Рекомендуется установить непустой пароль для встроенной учетной записи *Everyone* и изменить исходный пароль для встроенной учетной записи *Administrator* в соответствии с указаниями п. 5.3.2.

Перед началом взаимодействия между компьютером и контроллером с адресом <ipaddr> по протоколу SFTP с реквизитами встроенной учетной записи <user> следует обменяться ключами шифрования, для чего ввести следующую команду в командной строке Linux или Cygwin:

```
sftp <user>@<ipaddr>
```

и нажать Enter.

При первом соединении компьютера с контроллером по SFTP в окне консоли будет выведено сообщение, аналогичное следующему:

```
The authenticity of host '<ipaddr> (<ipaddr>)' can't be established.  
RSA key fingerprint is SHA256:6shT/06ZlQSEFHEWbAv9zZbTjMrKXQD++DrY1ySfynA.  
Are you sure you want to continue connecting (yes/no)?
```

Если есть полная уверенность, что обмен ключами выполняется именно с требуемым контроллером, то для продолжения сеанса следует ввести *yes* и нажать Enter.

При успешном установлении соединения с контроллером в окна консоли появится приглашение для ввода консоли *sftp*:

```
sftp>
```

Начиная с этого момента, имеется возможность просматривать содержимое файловой системы контроллера в текущем каталоге, создавать и удалять файлы и каталоги, а также копировать файлы между компьютером и контроллером с использованием команд SFTP, перечень которых может быть получен командой ?:

```
sftp> ?
Available commands:
bye                Quit sftp
cd path            Change remote directory to 'path'
chgrp grp path     Change group of file 'path' to 'grp'
chmod mode path    Change permissions of file 'path' to 'mode'
chown own path     Change owner of file 'path' to 'own'
df [-hi] [path]    Display statistics for current directory or
                  filesystem containing 'path'

exit              Quit sftp
get [-afPpRr] remote [local] Download file
reget [-fPpRr] remote [local] Resume download file
reput [-fPpRr] [local] remote Resume upload file
help             Display this help text
lcd path         Change local directory to 'path'
lls [ls-options [path]] Display local directory listing
lmkdir path      Create local directory
ln [-s] oldpath newpath Link remote file (-s for symlink)
lpwd             Print local working directory
ls [-lafhlNrSt] [path] Display remote directory listing
lumask umask     Set local umask to 'umask'
mkdir path       Create remote directory
progress         Toggle display of progress meter
put [-afPpRr] local [remote] Upload file
pwd             Display remote working directory
quit            Quit sftp
rename oldpath newpath Rename remote file
rm path         Delete remote file
rmdir path      Remove remote directory
symlink oldpath newpath Symlink remote file
version         Show SFTP version
!command        Execute 'command' in local shell
!              Escape to local shell
?              Synonym for help
```

Для завершения сеанса связи по SFTP следует ввести команду *quit* и нажать Enter.

Передача файлов между компьютером и контроллером может выполняться одной командой, вводимой в терминале Linux или Cygwin.

Для передачи файла <file> в каталог <dst> контроллера с IP-адресом <ipaddr> из каталога <src> компьютера с учетными данными пользователя <user> следует ввести следующую команду в командной строке Linux или Cygwin:


```
sftp <user>@<ipaddr>:<dst> '$'put <src>/<file>'
```

Например, для передачи файла *m.txt* из каталога *dir1* относительно текущего каталога компьютера в каталог *user* контроллера с адресом *10.0.0.4*, следует ввести команду:

```
sftp Everyone@10.0.0.4: <<< '$'put ./dir1/m.txt'
Connected to 10.0.0.4.
Changing to: /codesys/user/.
sftp> put ./dir1/m.txt
Uploading ./dir1/m.txt to /codesys/user/m.txt
./dir1/m.txt                               100% 1024   204.9KB/s   00:00
```

В данном случае на контроллере пароль учетной записи *Everyone* остается пустым, поэтому не был запрошен ввод пароля учетной записи *Everyone*.



	<p>Однострочные команды с символами &lt;&lt;&lt;, требующие ввода пароля при выполнении, работают не во всех терминальных программах, в частности, в Cygwin.</p>
---	--


Для выгрузки файла <file> из каталога <src> контроллера с IP-адресом <ipaddr> в каталог <dst> компьютера с учетными данными пользователя <user> следует ввести следующую команду в командной строке Linux или Cygwin:

```
sftp <user>@<ipaddr>:<src>/<file> <dst>
```

Например, для выгрузки файла normdump.txt из корневого каталога системы исполнения контроллера с адресом 10.0.0.4 в текущий каталог компьютера следует ввести команду:

```
sftp Administrator@10.0.0.4:normdump.txt ./
Administrator@10.0.0.4's password:
Connected to 10.0.0.4.
Fetching /codesys/normdump.txt to ./normdump.txt
/codesys/normdump.txt          100% 1024   171.1KB/s   00:00
```


При появлении в окне терминала подсказки Administrator@10.0.0.4's password: потребуется ввести пароль учетной записи Administrator и нажать Enter.

	<p>Утилита sftp в терминале Cygwin может работать неправильно, особенно при выполнении однострочных команд передачи файлов в контроллер с реквизитами учетной записи, требующей ввода пароля.</p> <p>Для сброса ключей шифрования сеансов взаимодействия с контроллером на компьютере следует выполнить команду:</p> <pre>ssh-keygen -R &lt;ipaddr&gt;</pre>
---	--



### 5.4.3. Передача файлов в IDE МЭК 61131-3

Передача файлов между контроллером и ПК средствами IDE МЭК 61131-3 осуществляется на вкладке **Файлы** редактора целевого устройства, показанной на рисунке 148.

Для просмотра содержимого корневого каталога системы исполнения и передачи файлов между ПК и контроллером следует:

1. Открыть проект, из которого было загружено приложение контроллера, или создайте новый проект для платформы, соответствующей используемому контроллеру.
2. В дереве проекта дважды щелкнуть на элементе, соответствующем используемому контроллеру.
3. Настроить сетевой путь доступа среды разработки к контроллеру в соответствии с указаниями п. 3.5.2 (через сервисный порт USB или RS-232C) или п. 3.5.3 (по протоколу TCP или UDP). При наличии пути доступа к контроллеру вкладка **Установки соединения** редактора устройств выглядит, как показано на рисунке 44 или на рисунке 47.
4. Перейти на вложенную вкладку **Файлы** редактора устройства и нажать кнопку , как показано на рисунке 148.  
При успешном соединении с контроллером и чтении информации о файлах и каталогах в области **Исполнение** вкладки **Файл** будут отображены файлы и подкаталоги корневого каталога системы исполнения контроллера.  
При необходимости следует перейти в каталог пользователя ./user, в котором создаются файлы и каталоги при обращении приложения к соответствующим функциям системной библиотеки SysFile, дважды щелкнув мышью на имени каталога.
5. В области **Хост** выбрать каталог файловой системы ПК, который содержит файлы, подлежащие передаче в контроллер, либо который будет использоваться в качестве целевого каталога для выгрузки файлов из контроллера на ПК.



6. Для передачи одного или нескольких файлов или каталогов из контроллера в текущий выбранный каталог ПК следует выбрать имена файлов и/или каталогов в области **Исполнение** и нажать кнопку .
- Для выбора нескольких файлов следует щелкнуть левой кнопкой мыши над их именами, удерживая нажатой клавишу Ctrl.
7. Для передачи одного или нескольких файлов из текущего выбранного каталога ПК в текущий каталог файловой системы контроллера следует выбрать имена файлов в области **Хост** и нажать кнопку .

Информация об использовании средств IDE МЭК 61131-3 для передачи файлов приведена в подразделе **Копирование файлов на/из ПК** интерактивной справочной системы IDE МЭК 61131-3.

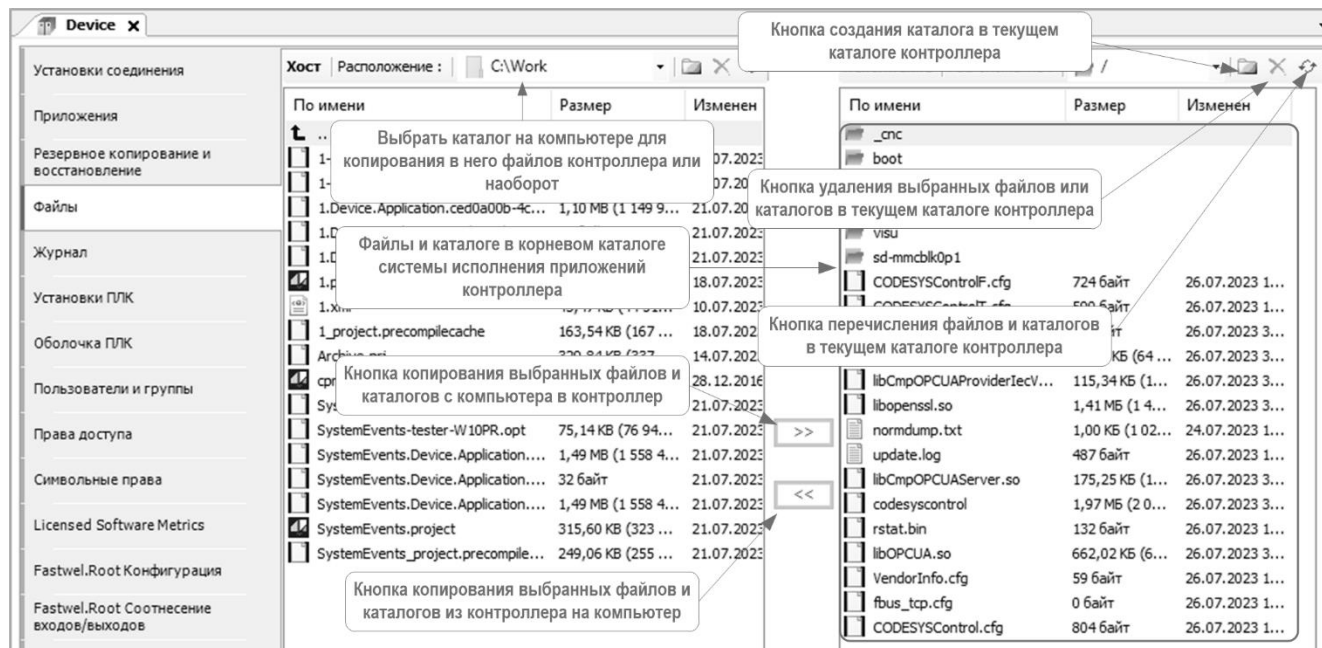


Рисунок 148. Вкладка **Файлы** для передачи файлов между контроллером и компьютером

## 5.5. Развертывание приложений

### 5.5.1. Общие сведения

*Развертыванием* называется процесс переноса файлов приложения и системных параметров с одного контроллера, называемого *эталонным*, на множество других контроллеров, называемых *целевыми*, при серийном производстве или обслуживании автоматизированных систем сбора данных и управления. По окончании развертывания на всех контроллерах применяются системные параметры и начинает функционировать приложение эталонного контроллера.

При серийном производстве и обслуживании систем на базе программируемых контроллеров запись приложения в контроллер из IDE МЭК 61131-3 и настройка системных параметров при помощи веб-конфигуратора требуют высокой квалификации производственного и обслуживающего персонала и могут занимать значительное время.

В данном подразделе приведено описание процедуры развертывания, позволяющей существенно сократить временные затраты на запись системных параметров и прикладного программного обеспечения в множество контроллеров, а также автоматизировать процесс записи.

Процесс развертывания состоит из следующих операций:

1. Формирование файла развертывания *norm.upl* в каталоге пользователя (user) эталонного контроллера или на съемном накопителе (USB или microSD), установленном в эталонный контроллер.
2. Запись файла развертывания в один или несколько целевых контроллеров.
3. Перезапуск целевых контроллеров.

Файл развертывания, сформированный по окончании разработки приложения, отладки и пусконаладочных работ, может использоваться в качестве резервной копии полной конфигурации прикладного программного обеспечения контроллера, состоящей из бинарных файлов приложения, файлов системных параметров, значений энергонезависимых переменных, базы данных безопасности и других системных файлов, находящихся в подкаталоге boot корневого каталога системы исполнения. Если в эталонный контроллер из среды разработки загружен исходный текст приложения командой **Онлайн – Загрузка исходного кода в подсоединенное устройства** или автоматически (см. п. 3.7.3), то файл исходного текста также будет включен в состав конфигурации приложения в файле развертывания.



При развертывании конфигурации одного приложения на целевых контроллерах, которые должны быть подключены к одной или двум подсетям в составе системы, уникальность IP-адресов разных контроллеров должна обеспечиваться установкой переключателей 1–8 или использованием динамического назначения адресов.

## 5.5.2. Операции процесса развертывания

### 5.5.2.1. Формирование файла развертывания на эталонном контроллере

Для формирования файла развертывания на эталонном контроллере:

1. Включить питание эталонного контроллера, если это не было сделано ранее.
2. В IDE МЭК 61131-3 открыть проект, содержащий приложение наиболее актуальной версии, загруженное в контроллер.
3. Настроить сетевой путь доступа среды разработки к контроллеру в соответствии с указаниями п. 3.5.2 (через сервисный порт USB или RS-232C) или п. 3.5.3 (по протоколу TCP или UDP). При наличии пути доступа к контроллеру вкладка **Установки соединения** редактора устройств выглядит, как показано на рисунке 44 или на рисунке 47.
4. В редакторе устройства, соответствующего целевой платформе, перейти на вкладку **Оболочка ПЛК (PLC Shell)**.
5. Для формирования файла развертывания в каталоге пользователя user выполнить команду оболочки ПЛК:

```
saveapp
```

При успешном завершении команды в окне оболочки ПЛК будет выведено сообщение:  
**user/norm.upl saved !**

Данное сообщение свидетельствует о формировании в каталоге пользователя user файла развертывания norm.upl, включающего в себя текущие значения энергонезависимых переменных.

Для формирования файла развертывания на карте microSD, установленной в эталонный контроллер, следует выполнить команду оболочки ПЛК:

```
saveapp microsd1
```

При успешном завершении команды в окне оболочки ПЛК будет выведено сообщение:  
**user/microsd1/norm.upl saved !**

Для формирования файла развертывания на USB-накопителе, установленном в гнездо USB1 эталонного контроллера, следует выполнить команду оболочки ПЛК:

```
saveapp usb1p1
```

При успешном завершении команды в окне оболочки ПЛК будет выведено сообщение:  
**=>user/usb1p1/norm.upl saved!**

Если файл развертывания не должен содержать текущие значения энергонезависимых переменных, при выполнении команды *saveapp* следует использовать дополнительный параметр *noretain*:

```
saveapp noretain
```

или

```
saveapp microsd1 noretain
```

Если файл развертывания не должен содержать системные параметры, то при выполнении команды *saveapp* следует использовать дополнительный параметр *noconfig*:

```
saveapp noconfig
```

Параметры *noconfig* и *noretain* могут следовать в любом порядке, но путь для формирования файла развертывания должен указываться после имени команды, например:

**saveapp noconfig noretain**

Если на эталонном контроллере активирован сервер OPC UA, выполнена настройка защищенных сеансов клиентских приложений с сервером, произведен обмен цифровыми сертификатами и перенос сертификатов клиентов в доверенную зону, то для сохранения возможности создания защищенных сеансов с этими же клиентами на целевом контроллере при выполнении команды *saveapp* следует использовать дополнительный параметр *cert=all*:

**saveapp cert=all**

6. Если файл развертывания предполагается записывать в целевые контроллеры с ПК, следует выгрузить файл *norm.upl* из каталога пользователя или со съемного накопителя по протоколу FTP, SFTP или FTPS (см. п. 5.4.2), или в IDE МЭК 61131-3 (см. п. 5.4.2.3).  
Если файл развертывания предполагается записывать в целевые контроллеры со съемного накопителя, необходимо либо завершить работу со съемным накопителем командой *eject* оболочки ПЛК (см. п. 5.2.3), либо выключить питание эталонного контроллера и извлечь съемный накопитель.  
Если файл развертывания предполагается записывать в целевые контроллеры с USB-накопителя, необходимо сначала включить поддержку устройств USB на целевых контроллерах (см. п. 5.2.1).



При значительной вычислительной загрузке процессора выполнение команды *saveapp* может занимать до 1 минуты и завершиться таймаутом связи между средой разработки и контроллером. В такой ситуации возможно использовать команду *saveapp microsd1* и перед повторным выполнением команды проверить наличие сформированного файла *norm.upl* на целевом накопителе.

#### 5.5.2.2. Развертывание с карты microSD

Для развертывания приложения и системных параметров с карты microSD на целевом контроллере:

1. Выключить питание целевого контроллера.
2. Установить карту microSD с файлом *norm.upl* в целевой контроллер.
3. Включить питание целевого контроллера.
4. Убедиться, что через некоторое время индикатор APP светится желтым цветом, а затем, после успешного развертывания, индикаторы RUN, APP, I/O и USR1 (USER) кратковременно светятся зеленым цветом.



Если после развертывания карта microSD с файлом развертывания остается в гнезде целевого контроллера, то повторное развертывание из этого файла на карте будет произведено после извлечения и повторной установки карты в гнездо.


В связи с этим рекомендуется не оставлять карту с файлом развертывания *norm.upl* в гнезде контроллера, поскольку факт наличия файла развертывания на карте может быть забыт, что в будущем приведет к нежелательному развертыванию приложения с карты microSD.

#### 5.5.2.3. Развертывание с USB-накопителя

Для развертывания приложения и системных параметров с USB-накопителя на целевом контроллере:

1. Включить питание целевого контроллера
2. В веб-конфигураторе на странице **Система** включить поддержку устройств USB, нажать **Применить конфигурацию** и дождаться повторного запуска контроллера.
3. Присоединить USB-накопитель с файлом *norm.upl* к одному из гнезд целевого контроллера.
4. Перезапустить целевой контроллер любым способом.

5. Убедиться, что через некоторое время индикатор APP светится желтым цветом, а затем, после успешного развертывания, индикаторы RUN, APP, I/O и USR1 кратковременно светятся зеленым цветом.

	<p>Если после развертывания USB-накопитель с файлом развертывания остается установленным в целевой контроллер, то повторное развертывание из этого файла с накопителя будет произведено после извлечения и повторной установки данного накопителя в контроллер.</p> <p>В связи с этим рекомендуется не оставлять USB-накопитель с файлом развертывания <i>norm.upl</i> в гнезде USB контроллера, поскольку факт наличия файла развертывания на накопителе может быть забыт, что в будущем приведет к нежелательному развертыванию приложения с USB-накопителя.</p>
---	--

#### 5.5.2.4. Развертывание с компьютера

Для развертывания приложения и системных параметров с ПК:

1. Загрузить файл развертывания *norm.upl* в один или несколько целевых контроллеров. Загрузка файла в корневой каталог системы исполнения по протоколу FTP, SFTP или FTPS выполняется согласно указаниям п. 5.4.2. При использовании утилит командной строки возможно автоматически загрузить файл в несколько контроллеров. Загрузка файла в корневой каталог системы исполнения из IDE МЭК 61131-3 выполняется в соответствии с указаниями п. 5.4.2.3.
2. Выполнить полный перезапуск целевого контроллера выключением и повторным включением питания, нажатием кнопки сброса, выполнением команды *reboot* оболочки ПЛК в IDE МЭК 61131-3 или нажатием кнопки **Перезапустить контроллер** на странице **Система** веб-конфигуратора.
3. Убедиться, что после полного перезапуска через некоторое время индикатор APP светится желтым цветом, а затем, после успешного развертывания, индикаторы RUN, APP, I/O и USR1 кратковременно светятся зеленым цветом.

Развертывание данным способом, независимо от результата, выполняется однократно, после чего файл *norm.upl* удаляется

#### 5.6. Обновление системного программного обеспечения

Файлы обновления системного программного обеспечения контроллеров Fastwel *norm.dnl* находятся в подкаталога, соответствующих типам контроллеров, по адресу:

[ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel\\_IO/Version3/Firmware](ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Firmware)

Например, файл обновления системного программного обеспечения контроллера CPM810-03 находится по адресу:

[ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel\\_IO/Version3/Firmware/CPM810-03](ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version3/Firmware/CPM810-03)

Файл обновления микропрограмм модулей ввода-вывода *ffw.dnl* находится по адресу:

[ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel\\_IO/Version2/Firmware/Modules](ftp://ftp.fastwel.ru/pub/Hardware/Fastwel/Fastwel_IO/Version2/Firmware/Modules)




Обновление системного программного обеспечения или микропрограмм модулей ввода-вывода может быть выполнено с ПК, с карты microSD или с USB-накопителя:

1. Обновление с ПК выполняется путем загрузки соответствующего файла в корневой каталог системы исполнения (см. п. 6.5.1) контроллера. Загрузка файла из веб-конфигуратора выполняется на странице Система нажатием кнопки **Загрузить системный файл**. Загрузка файла в корневой каталог системы исполнения по протоколу FTP, FTPS или SFTP выполняется согласно указаниям п. 5.4.2. При использовании утилит командной строки возможно автоматически загрузить файл в несколько контроллеров. Загрузка файла в корневой каталог системы исполнения из IDE МЭК 61131-3 выполняется в соответствии с указаниями п. 5.4.2.3.

2. Для обновления с карты microSD следует записать файл обновления на карту, выключить питание, установить карту в контроллер, подлежащий обновлению, и включить питание.
3. Для обновления с USB-накопителя следует записать файл обновления на накопитель, присоединить USB-накопитель к одному из гнезд контроллера, включить поддержку USB на странице **Система** в веб-конфигураторе и применить конфигурацию.

После загрузки файла обновления в корневой каталог системы исполнения или после установки USB-накопителя с файлом обновления в гнездо контроллера необходимо перезапустить контроллер выключением и повторным включением питания, нажатием кнопки сброса, выполнением команды *reboot* оболочки ПЛК IDE МЭК 61131-3 или нажатием кнопки **Перезапустить контроллер** на странице **Система** веб-конфигуратора.

В процессе обновления системного программного обеспечения может произойти более одного перезапуска контроллера, после чего контроллер вновь запустится с ранее загруженным приложением и системными параметрами. Обновление системного программного обеспечения реализовано таким образом, что случайное пропадание питания во время развертывания обновления не приведет к неисправности контроллера.

	Обновление системного программного обеспечения контроллера может выполняться в течение нескольких минут.
	В процессе обновления модулей ввода-вывода индикатор I/O контроллера светится желтым цветом. Обновление будет выполнено для всех модулей ввода-вывода локальной и удаленных шин, присутствующих в конфигурации приложения, загруженного в контроллер.
	В процессе обновления микропрограмм модулей ввода-вывода при пропадании питания контроллера, удаленных адаптеров шины FBUS или промежуточных модулей питания шины, как минимум, один модуль может выйти из строя!.

### 5.7. Оценка загрузки процессора

Система исполнения приложений МЭК 61131-3 является одним из процессов мультизадачной операционной системы реального времени контроллера и состоит из множества потоков исполнения (задач), обеспечивающих выполнение кода пользовательского приложения, загруженного в контроллер, обмен данными с периферийными модулями, подключенными к межмодульным шинам контроллера, взаимодействие по сети и других операций. Более подробная информация о задачах приложения и их приоритетах приведена в п. 4.4 настоящего руководства.

Загрузкой процессора далее называется относительная степень занятости процессора выполнением кода некоторой задачи в течение заданного интервала времени. Например, если задача выполнялась 5 мс в течение 1 с, а остальное время находилась в состоянии ожидания, то загрузка процессора для данной задачи составляет:

$$\frac{100 \% \cdot 5 \text{ (мс)}}{1000 \text{ (мс)}} = 0,5 \%$$

Для оценки загрузки процессора выполнением кода разных сервисов и отдельных задач предназначена команда *topservice* оболочки ПЛК (начиная с версии 3.3.x.x системного программного обеспечения контроллеров Fastwel).

При выполнении команды *topservice* без параметров производится оценка загрузки процессора всеми сервисами системного программного обеспечения контроллера на интервале времени 1 с, например:

**topservice**

```

=>CPU count      : 1
CPU0             : 61.321 %
Total            : 61.321 %
System           : 1.830 %
Kernel           : 0.000 %
Runtime          : 59.490 %
  Core           : 1.830 %
  FBUS           : 3.661 %
  Visualization  : 0.915 %
  Communication  : 3.661 %
  MODBUS         : 6.407 %
  OPCUA          : 1.830 %
  IEC-Tasks      : 41.186 %
  Other          : 0.000 %

```

Информация о загрузке процессора выводится командой *topservice* в следующих полях:

**CPU count** – количество процессоров, на которых функционирует системное программное обеспечение контроллера;

**CPU<sub>n</sub>** – полная загрузка процессора с номером *n* в процентах;

**Total** – полная загрузка всех процессоров, количество которых отображено в поле **CPU count**;

**System** – загрузка всех процессоров задачами сервисов операционной системы вне процесса системы исполнения приложений МЭК 61131-3 контроллера;

**Kernel** – загрузка всех процессоров задачами, обработчиками аппаратных прерываний и другими исполняемыми объектами ядра операционной системы контроллера;

**Runtime** – загрузка всех процессоров задачами процесса системы исполнения приложений CODESYS V3;

**Runtime.Core** – загрузка всех процессоров задачами, относящимися к ядру системы исполнения приложений МЭК 61131-3;

**Runtime.FBUS** – загрузка всех процессоров задачами, обслуживающими обмен данными с периферийными модулями, подключенными к локальным и удаленным портам межмодульной шины FBUS контроллера;

**Runtime.Visualization** – загрузка всех процессоров задачами подсистемы целевой визуализации;

**Runtime.Communication** – загрузка всех процессоров задачами сервиса сетевого взаимодействия по протоколу IDE Gateway со средой разработки и/или CODESYS OPC Server V3, а также сетевыми переменными без использования OPC UA;

**Runtime.MODBUS** – загрузка всех процессоров задачами сервисов мастера и подчиненного узлов протокола MODBUS;

**Runtime.OPCUA** – загрузка всех процессоров задачами сервера OPC UA;

**Runtime.Iec-Tasks** – загрузка всех процессоров задачами, выполняющими пользовательский код приложения, загруженного в контроллер;

**Runtime.Other** – загрузка всех процессоров задачами приложения, относящимся к библиотекам среды разработки, которые явно или неявно подключены к приложению.

При выполнении команды *topservice* с параметром *all* информация о загрузке процессора дополняется именами задач отдельных сервисов процесса системы исполнения, например:

**topservice all**

```

=>CPU count      : 1
CPU0             : 68.269 %
Total            : 68.269 %
System           : 3.846 %
Kernel           : 0.000 %
Runtime          : 64.423 %
  Core           : 0.962 %
    01179 [main          ]: 0.000 %
    01192 [TimerThread   ]: 0.000 %
    01193 [SchedProcessorLoad0]: 0.000 %

```



```

01194 [SchedException]           ]: 0.000 %
01195 [TaskGapTask]              ]: 0.000 %
01196 [PlcShellLocalTask]        ]: 0.962 %
01197 [CouplerException]         ]: 0.000 %
01198 [ServiceTask]              ]: 0.000 %
01797 [AsyncTask128]             ]: 0.000 %
FBUS : 2.885 %
01199 [FBUS0 PORT]               ]: 0.000 %
01200 [FBUS1 PORT]               ]: 0.000 %
01201 [FBUS2 PORT]               ]: 0.000 %
01202 [FBUS]                     ]: 2.885 %
Visualization : 4.808 %
01205 [SysWindowSingleTaskingTask] ]: 4.808 %
Communication : 1.923 %
01251 [IPC_SHM_Ipc__VisuClientControlle] ]: 0.000 %
01262 [BlkDrvUdp]                ]: 0.962 %
01263 [BlkDrvTcp]                ]: 0.000 %
01264 [BlkDrvCom]                ]: 0.962 %
MODBUS : 7.692 %
01253 [MBM_MASTER_TASK]          ]: 3.846 %
01957 [MBTCP_SERVER_TASK]        ]: 0.962 %
01958 [MBTCP_CLIENT_TASK]        ]: 2.885 %
OPCUA : 1.923 %
01265 [OPCUAServer]              ]: 1.923 %
IEC-Tasks : 44.231 %
01959 [CpuLoadTask]              ]: 0.000 %
01960 [MainTask]                 ]: 0.000 %
01961 [Task]                     ]: 40.385 %
01962 [VISU_TASK]                ]: 3.846 %
Other : 0.000 %
01204 [CAAEvtTask]               ]: 0.000 %

```

В данном примере в приложении, загруженном в контроллер, имеется несколько задач приложения с именами *CpuLoadTask*, *MainTask*, *Task* и *VISU\_TASK*, а также одна скрытая задача *CAAEvtTask*., загрузка процессора которых на интервале 1 с составляет 0 %, 0 %, 40,385 %, 3,846 % и 0 % соответственно.

Кроме того, выполнение команды оболочки ПЛК из веб-конфигуратора в контексте задачи *PlcShellLocalTask* ядра системы исполнения требует 3,252 % процессорного времени на интервале 1 с. Если команда оболочки ПЛК запущена в среде разработки, то ее выполнение осуществляется в контексте задачи *AsyncTask128*.



При разных запусках команды *topservice* оболочки ПЛК полученные значения загрузки процессора одними и теми же сервисами могут отличаться в виду различного распределения процессорного времени между разными задачами в зависимости от алгоритма приложения, объема входящего и исходящего сетевого трафика и влияния других факторов в окружении системы исполнения.




## 6. Системные библиотеки

### 6.1. Общие сведения

Библиотеки являются мощным средством повышения продуктивности при разработке приложений МЭК 61131-3. Библиотеки IDE МЭК 61131-3 могут одновременно содержать программные единицы, разработанные на языках ГОСТ Р МЭК 61131-3, и функции, реализация которых находится в системе исполнения приложений контроллера.

В настоящем документе *системной* называется библиотека IDE МЭК 61131-3, содержащая хотя бы одну функцию, реализованную в системе исполнения контроллера.

IDE МЭК 61131-3 включает в себя множество системных библиотек, имена которых начинаются с префиксов *Cmp* и *Sys*. Префикс *Cmp* используется при именовании библиотек, обеспечивающих доступ к функциям соответствующих компонентов ядра системы исполнения, а префикс *Sys* используется в именах библиотек, предоставляющих доступ к окружению системы исполнения: файлам, сокетам, последовательным портам и т.п.

	<p>В контроллерах с системой исполнения приложений МЭК 61131-3 реализован фиксированный набор системных библиотек определенных версий, поэтому в процессе инсталляции Fastwel PLC Application Toolkit для каждой целевой платформы Fastwel устанавливается отдельный набор системных библиотек тех версий, которые поддерживаются системой исполнения соответствующего контроллера.</p>
---	---

Все контроллеры Fastwel поддерживают следующие системные библиотеки IDE МЭК 61131-3, которые допускается использовать при разработке приложений:

1. *CmpApp* – служит для получения информации о загруженном приложении, а также для управления состоянием приложения. Примеры ограниченного использования данной библиотеки для автоматического запуска приложения сразу после загрузки из среды разработки имеются в проектах PlatformControl.project, SystemEvents.project, TasksExchange.project (см. таблицу 4), а также в п. 4.6.2 настоящего документа.
2. *CmpErrors* – содержит коды ошибок, возвращаемые функциями системных библиотек.
3. *CmpIecTask* – предназначена для получения информации о задачах исполняющегося приложения.
4. *CmpLog* – позволяет добавлять записи в системный журнал контроллера.
5. *SysCom* – предназначена для обмена данными через последовательные порты. Примеры использования библиотеки имеются в проектах ModbusRtuClient.project и SerialConsole.project (см. таблицу 4).
6. *SysFile* – библиотека доступа к файлам. Примеры использования библиотеки имеются в проектах SimpleDataLogger.project, FileRetainer.project, SystemEvents.project (см. таблицу 4).
7. *SysDir* – библиотека доступа к каталогам (папкам) файловой системы. Пример использования библиотеки имеется в проекте SimpleDataLogger.project (см. таблицу 4).
8. *SysCpuHandling* – содержит функции атомарного доступа к переменным и битовым полям переменных приложения и позволяет реализовывать относительно легковесные примитивы синхронизации доступа к общим данным из нескольких задач приложения.
9. *SysMem* – содержит функции копирования и сравнения содержимого областей памяти.
10. *SysSocket* – предназначена для реализации собственных протоколов обмена по сети Ethernet с использованием протоколов UDP и TCP. Примеры использования библиотеки имеются в проектах TCP\_UDP\_Sockets.project и 2xPLCs\_Sockets.project (см. таблицу 4).
11. *SysTarget* – содержит функции доступа к статическим свойствам целевой платформы контроллера: идентификатору платформы, имени устройства и узла, серийному номеру и т.п. Пример чтения и преобразования серийного номера контроллера имеется в проекте PlatformControl.project (см. таблицу 4).
12. *SysTime* – содержит типы данных и наборы функций для работы с системным временем и измерения временных интервалов. Примеры использования функций библиотеки

имеются в проектах DateTimeUtilities.project, Networkvariables.project, SystemEvents.project и SimpleDataLogger.project (см. таблицу 4).

13. ISysTypes – содержит базовые системные типы RTS\_IEC\_HANDLE, RTS\_IEC\_RESULT и константу RTS\_INVALID\_HANDLE.

Документация на перечисленные выше системные библиотеки отображается в соответствующей вкладке **Менеджера библиотек**, показанной на рисунке 149.

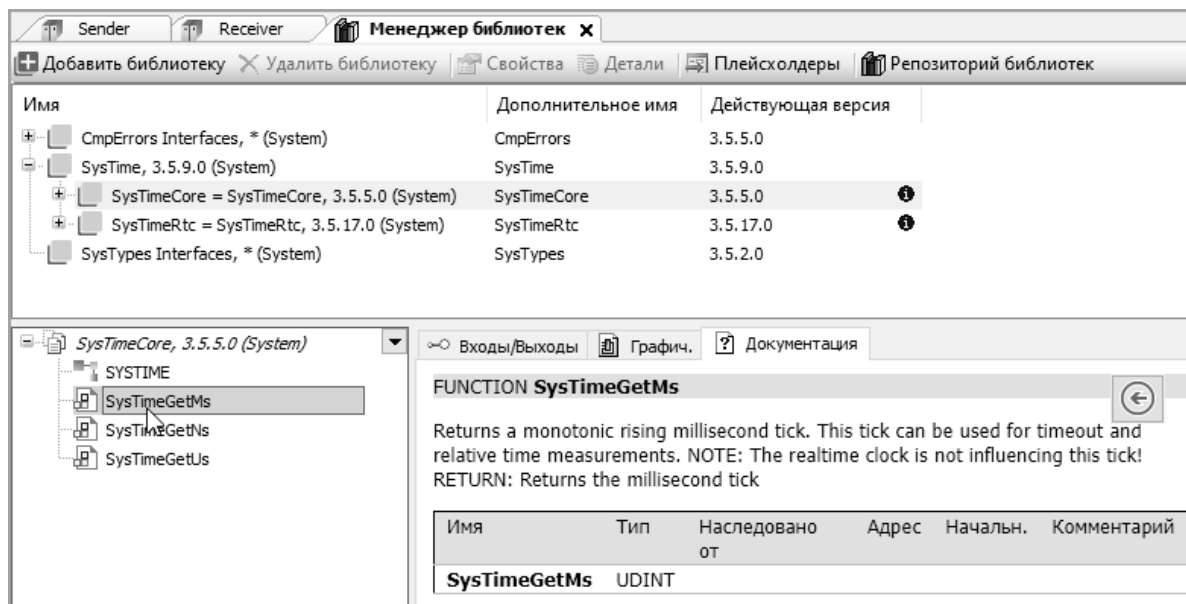


Рисунок 149 – Представление документации на функции системных библиотек

В категории *Application* набора библиотек, входящих в IDE МЭК 61131-3, имеются две библиотеки Standard и Util, содержащие минимальный стандартизованный набор функциональных блоков и функций формирования и обработки сигналов, преобразования строк и манипуляции битовыми полями. Данные библиотеки используются в примерах программирования, устанавливаемых при инсталляции Fastwel PLC Application Toolkit. Кроме того, интерактивная справочная система IDE МЭК 61131-3 содержит достаточно подобное описание программных единиц данных библиотек.

В состав Fastwel PLC Application Toolkit в настоящее время входят следующие библиотеки поддержки платформ Fastwel, имена которых начинаются с соответствующего префикса:

1. FastwelBoard – системная библиотека, содержащая функциональные блоки и функции доступа к специфическим органам управления и индикации контроллеров Fastwel. Пример применения данной библиотеки содержится в проекте FastwelBoardExample.project.
2. FastwelCore – данная системная библиотека содержит функции, функциональные блоки и системные события, специфические для платформ Fastwel, в том числе предназначенные для облегчения миграции приложений, разработанных в среде разработки CoDeSys 2.3 для контроллеров Fastwel I/O. Примеры применения данной библиотеки имеются в проектах FileRetainer.project, PlatformControl.project, SystemEvents.project и TasksExchange.project.
3. FastwelFbusIO – данная библиотека разработана на языке Structured Text ГОСТ Р МЭК 61131-3 и содержит функциональные блоки и функции формирования и обработки данных периферийных модулей Fastwel I/O и Fastwel I/O-2.
4. FastwelCpuLoad – данная системная библиотека содержит функции, предназначенные для оценки загрузки процессора в коде приложения. Пример применения данной библиотеки имеется в проекте FastwelCpuLoadExample.project.
5. FastwelModbusRTUClientSerial – данная библиотека является развитым примером использования библиотеки SysCom, разработана на языке Structured Text ГОСТ Р МЭК 61131-3 и содержит функциональные блоки и дополнительную инфраструктуру для реализации в приложении мастера протокола MODBUS RTU. Пример применения библиотеки имеется в проекте ModbusRtuClient.project.
6. FastwelModbus – данная системная библиотека содержит функциональные блоки, функции и типы данных для управления сервисами протоколов MODBUS и MODBUS

TCP в приложении контроллера. Описание функций и функциональных блоков библиотеки приведено в документе ИМЕС.00300-03 33 03 (см. таблицу 1). Пример применения библиотеки имеется в проекте ModbusControl.project.

7. FastwelIec60870 – данная системная библиотека содержит типы данных и функциональные блоки для работы в приложении с сервисом протокола ГОСТ Р МЭК 60870-5-104. Описание типов данных и функциональных блоков библиотеки приведено в документе ИМЕС.00300-03 33 04 (см. таблицу 1).
8. FastwelRemovableMedia – данная системная библиотека предназначена для программного управления съемными дисковыми накопителями, входящими в состав контроллера.

Проекты с примерами применения библиотек перечислены в таблице 4.

Библиотеки FastwelBoard, FastwelCore, FastwelFbusIO и FastwelModbus всегда автоматически подключаются к проекту для контроллеров Fastwel, а библиотека FastwelIec60870 подключается автоматически при добавлении устройства IEC60870 в конфигурацию приложения для контроллера Fastwel.

Настоящий раздел содержит информацию по применению некоторых общих библиотек с префиксом Fastwel, а также описание особенностей использования системных библиотек SysFile, SysDir и SysCom в приложениях, разрабатываемых для контроллеров Fastwel.

Исходный текст функциональных блоков и функций библиотек Fastwel открыт для изучения и для реализации собственных функций и функциональных блоков.

Для просмотра исходного текста следует дважды щелкнуть на имени блока в окне **Менеджера библиотек** или скомпилировать проект, содержащий экземпляры функциональных блоков или функций из библиотек Fastwel, щелкнуть правой кнопкой мыши над именем функции или типа блока в области деклараций и выполнить команду **Обзор – Перейти к определению** в контекстном меню, как показано на рисунке 150.

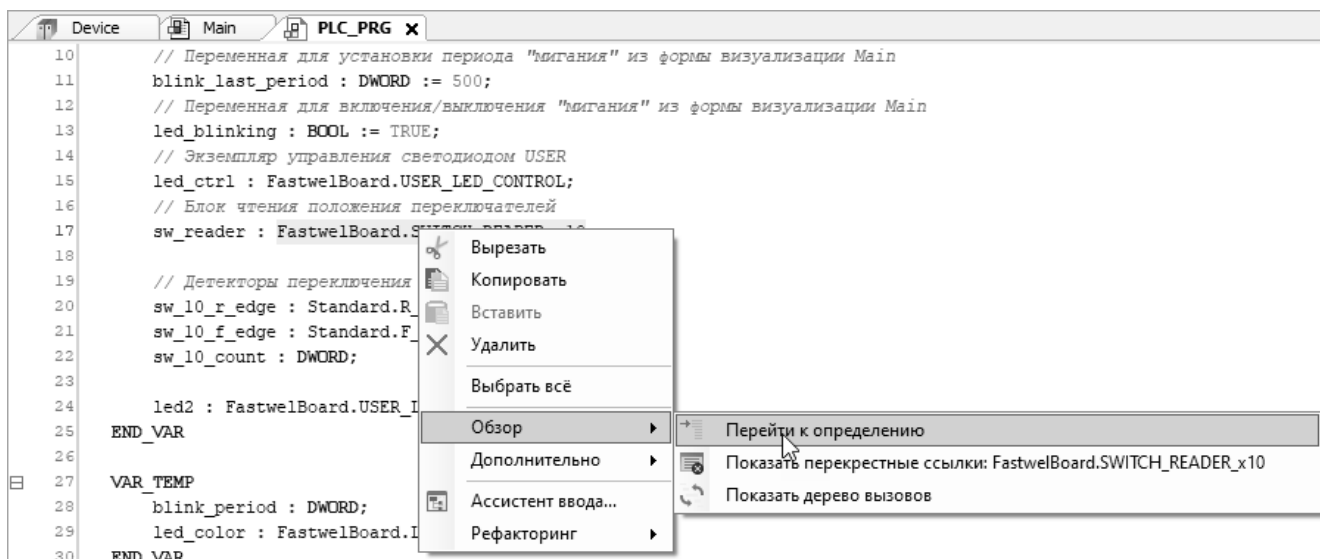


Рисунок 150 – Переход к просмотру исходного текста функционального блока из библиотеки FastwelBoard


## 6.2. Библиотеки системных типов CmpErrors и ISysTypes

Большинство функций системных библиотек IDE МЭК 61131-3, предоставляющих доступ к ресурсам окружения исполняющегося приложения, принимают в качестве параметров системные идентификаторы (хэндлы) ресурсов, представленные типом RTS\_IEC\_HANDLE и возвращают результат в виде кодов ошибок, представленных типом RTS\_IEC\_RESULT.

Оба типа декларированы в системной библиотеке SysTypes, которую следует добавлять в проект посредством вспомогательной интерфейсной библиотеки ISysTypes или ISysTypes2.

Тип RTS\_IEC\_HANDLE является алиасом (псевдонимом) для типа POINTER TO BYTE (указатель на байт), и предназначен для хранения идентификатора системного ресурса (файла, устройства, сокета

и т.п.), внешнего по отношению к исполняющемуся приложению и, как правило, получаемого у операционной системы контроллера.

	<p>Система исполнения приложений МЭК 61131-3 не содержит подсистемы автоматического учета и освобождения внешних ресурсов, запрошенных приложением, в связи с чем необходимо самостоятельно освобождать полученные системные идентификаторы в обработчиках соответствующих системных событий и/или в специальных методах функциональных блоков FB_Exit и FB_Reinit (см. п. 4.6.2 и п. 4.6.3).</p>
---	---

Начальное значение переменной типа RTS\_IEC\_HANDLE, соответствующее отсутствующему или недействительному внешнему ресурсу, представлено константой RTS\_INVALID\_HANDLE. Рекомендуется присваивать это значение при декларации переменных типа RTS\_IEC\_HANDLE. Кроме того, сразу после освобождения (закрытия) внешнего ресурса, представленного некоторой переменной типа RTS\_IEC\_HANDLE, необходимо присвоить этой переменной значение RTS\_INVALID\_HANDLE:

```
// Метод FB_Exit блока SERIAL_COMM_FB из ModbusRtuClient.project
METHOD FB_Exit
VAR_INPUT
    bInCopyCode : BOOL;
END_VAR

IF hSerialPort <> RTS_INVALID_HANDLE THEN
    // освобождаем внешний ресурс
    SysCom.SysComClose(hSerialPort);
    // присваиваем начальное недействительное значение
    hSerialPort := RTS_INVALID_HANDLE;
END_IF
```

Тип RTS\_IEC\_RESULT является алиасом типа UDINT и предназначен для передачи приложению кода ошибки вызова системной функции. Перечень кодов ошибок находится в библиотеке CmpErrors в списке констант Errors.

Во избежание дефектов в разрабатываемых приложениях всегда рекомендуется анализировать коды ошибок, возвращаемые функциями системных библиотек:

```
// Фрагмент исходного текста блока SERIAL_COMM_FB из ModbusRtuClient.project
IF hSerialPort = RTS_INVALID_HANDLE THEN
    (* Попытка открыть последовательный порт *)
    hSerialPort := SysCom.SysComOpen(serialPortParameters.sPort, ADR(iec_res));
END_IF

IF hSerialPort <> RTS_INVALID_HANDLE AND CmpErrors.Errors.ERR_OK = iec_res THEN
    (* Порт успешно открыт, выполняем настройку параметров *)
    iec_res := SysCom.SysComSetSettings(hSerialPort, ADR(serialPortParameters), 0);

    IF CmpErrors.Errors.ERR_OK = iec_res THEN
        (* Настройка успешна, готовы к работе *)
        state := MBCS_READY;
        status := MBCS_OK_IDLE;
    ELSE
        (* Не удалось настроить параметра порта, закрываем порт *)
        SysCom.SysComClose(hSerialPort);
        hSerialPort := RTS_INVALID_HANDLE;
        state := MBCS_UNCERTAIN;
        status := MBCS_FAIL_BAD_PARAMETER;
    END_IF
END_IF
```

### 6.3. Библиотека FastwelBoard

#### 6.3.1. Общие сведения

Библиотека FastwelBoard содержит функции чтения состояния 10-позиционного блока переключателей, управления одним (USER) или двумя (USR1, USR2) двухцветными светодиодными индикаторами, а также для выполнения программного перезапуска контроллера.

Кроме того, в библиотеку включены функциональные блоки управления индикаторами и получения состояния переключателей.

### 6.3.2. Типы данных

#### 6.3.2.1. BoardLeds\_t

Данный тип содержит перечисление идентификаторов программно-управляемых светодиодных индикаторов контроллера: RUN/ERR, APP, I/O и зарезервированных для управления из кода приложения.

```
TYPE BoardLeds_t :
(
  (* резерв, не использовать! *)
  LED_RUNERR := 0,
  (* резерв, не использовать! *)
  LED_APP := 1,
  (* резерв, не использовать! *)
  LED_IO := 2,
  (* Пользовательский светодиод USER *)
  LED_USER := 3,
  (* Пользовательский светодиод USR1 *)
  LED_USR1 := LED_USER,
  (* Пользовательский светодиод USR2 *)
  LED_USR2 := 4
);
END_TYPE
```



Управление индикаторами RUN/ERR, APP и I/O осуществляется сервисами системы исполнения контроллера, в связи с чем доступ к ним из кода приложения, загруженного в контроллер, приведет к неправильной работе индикации.

Управление индикаторами COMM1 и COMM2 выполняется аппаратными средствами сетевых интерфейсов.

#### 6.3.2.2. BootReason\_t

Данный тип содержит перечисление кодов причин последнего запуска контроллера:

```
TYPE BootReason_t :
(
  (* нет возможности определить причину *)
  BOOT_REASON_UNKNOWN := 0,
  (* включение питания *)
  BOOT_REASON_POWERUP := 1,
  (* программный перезапуск *)
  BOOT_REASON_SOFTWARE := 2,
  (* перезапуск по сторожевому таймеру *)
  BOOT_REASON_WATCHDOG := 3
);
END_TYPE
```

BOOT\_REASON\_POWERUP соответствует запуску по включению питания контроллера, BOOT\_REASON\_SOFTWARE – любому программному перезапуску, а BOOT\_REASON\_WATCHDOG – перезапуску по системному сторожевому таймеру.

#### 6.3.2.3. LedsColor\_t

Данный тип содержит перечисление цветов индикаторов:

```
TYPE LedsColor_t :
(
  (* Выключен *)
  LED_COLOR_NONE := 0,
  (* Зеленый *)
  LED_COLOR_GREEN := 1,
  (* Красный *)
  LED_COLOR_RED := 2
);
END_TYPE
```

### 6.3.3. Функции

#### 6.3.3.1. SysBoardReadSwitches

Функция предназначена для чтения состояния переключателей. При успешном вызове маска состояний записывается в битовые поля переменной, адрес которой передан в качестве первого параметра. Битовому полю с номером 0 соответствует состояние первого переключателя (1 – включен, 0 – включен), а битовому полю с номером 9 – состояние десятого переключателя.



Функция SysBoardReadSwitches требует значительного количества вычислительных ресурсов, и время ее выполнения составляет порядка 350 мкс.

Рекомендуется использовать функциональный блок SWITCH\_READER\_x10 с периодом опроса блока переключателей от 100 мс и более, и вызывать экземпляр блок в задаче со значением приоритета от 15 и более.

#### Прототип

```
FUNCTION SysBoardReadSwitches : RTS_IEC_RESULT
VAR_INPUT
  (* Указатель на массив битовых полей, в который будет помещен результат *)
  pSwitchMask : POINTER TO BYTE;
  (* Размер массива pSwitchMask в байтах *)
  MaskSize : DWORD;
END_VAR
```

#### Входные параметры

**pSwitchMask : POINTER TO BYTE**

Адрес переменной (указатель на переменную), в битовые поля с номерами от 0 до 9 которой будут записаны логические состояния переключателей с 1 по 10 контроллера. Рекомендуется использовать тип DWORD для объявления переменной.

**MaskSize : DWORD**

Размер переменной (в байтах), адрес которой передан в качестве первого параметра.

#### Возвращаемый результат

**CmpErrors.Errors.ERR\_OK**

Состояние переключателей прочитано успешно.

**CmpErrors.Errors.ERR\_FAILED**

Неправильные параметры или не удалось прочесть состояние переключателей.


#### Пример

```
PROGRAM PLC_PRG
VAR
  // Пользовательский переключатель 9
  sw9_state : BOOL;
  // Пользовательский переключатель 10
  sw10_state : BOOL;
  // Целевая переменная
  sw_states : DWORD;
  // Результат вызова
  iec_res : RTS_IEC_RESULT;
END_VAR

iec_res := FastwelBoard.SysBoardReadSwitches(ADR(sw_states), sizeof(sw_states));
IF CmpErrors.Errors.ERR_OK = iec_res THEN
  sw9_state := sw_states.8;
  sw10_state := sw_states.9;
END_IF
```

### 6.3.3.2. SysBoardReadSwitch

Функция реализована на языке Structured Text ГОСТ Р МЭК 61131-3 и предназначена для чтения состояния одного переключателя путем вызова системной функции SysBoardReadSwitches.

	<p>Не рекомендуется использовать данную функцию для последовательного чтения состояния более одного переключателя, поскольку при каждом вызове этой функции происходит вызов SysBoardReadSwitches со временем выполнения порядка 350 мкс. Для чтения состояния переключателей рекомендуется использовать функциональный блок SWITCH_READER_x10 с периодом опроса блока переключателей от 100 мс и более, и вызывать экземпляр блок в задаче со значением приоритета от 15 и более.</p>
---	--

#### Прототип

```
FUNCTION SysBoardReadSwitch : BOOL
VAR_INPUT
  (* Индекс переключателя от 0 до 9 *)
  idx : WORD;
  (* Результат выполнения операции чтения.
    ERR_OK - в случае успеха,
    <> ERR_OK - в противном случае *)
  pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

#### Входные параметры

**idx : WORD**

Номер переключателя от 0 до 9.

**pResult : POINTER TO RTS\_IEC\_RESULT**

Адрес переменной для возврата результата выполнения функции. Допускается передавать 0.

#### Возвращаемый результат

**TRUE**

Переключатель с заданным номером включен.

**FALSE**

Переключатель с заданным номером выключен, либо вызов функции завершился неудачей. Для проверки успешности вызова следует убедиться в равенстве ERR\_OK переменной, адрес которой передан функции в качестве второго параметра.

#### Пример

```
PROGRAM PLC_PRG
VAR
  // Пользовательский переключатель 9
  sw9_state : BOOL;
  // Признак ошибки доступа к переключателю 9
  sw9_error : BOOL;
  // Результат вызова
  iec_res : RTS_IEC_RESULT;
END_VAR

sw9_state := FastwelBoard.SysBoardReadSwitch(8, ADR(iec_res));
sw9_error := CmpErrors.Errors.ERR_OK <> iec_res;
```

### 6.3.3.3. SysBoardSetUserLed

Функция реализована на языке Structured Text ГОСТ Р МЭК 61131-3 и предназначена для управления светодиодным индикатором USER и USR1.



	<p>Приложение, управляющее индикатором USER или USR1, для отключения индикатора перед загрузкой нового приложения, в котором не предусмотрено управление индикатором, должно вызвать функцию SysBoardSetUserLed с параметром FastwelBoard.LED_COLOR_NONE в обработчике системного события <i>PrepareExit</i>. Информация об обработчиках системных событий приведена в п. 4.5. Информация о методе FB_Exit функциональных блоков приведена в п. 4.6.2.</p> <p>Если управление индикатором USER осуществляется в пользовательском функциональном блоке, следует выключить индикатор в методе функционального блока FB_Exit.</p> <p>В системном программном обеспечении контроллеров Fastwel версии 2.x.x.x и выше индикатор USER выключается автоматически перед загрузкой нового приложения или по команде сброса текущего приложения.</p>
---	--

#### Прототип

```
FUNCTION SysBoardSetUserLed : BOOL
VAR_INPUT
  (* Цвет светодиода *)
  color : LedsColor_t;
END_VAR
```

#### Входные параметры

color : LedsColor\_t

Цвет светодиода USER:

FastwelBoard.LED\_COLOR\_NONE – светодиод выключен,

FastwelBoard.LED\_COLOR\_GREEN – светодиод светится зеленым цветом;

FastwelBoard.LED\_COLOR\_RED – светодиод светится красным цветом;

#### Возвращаемый результат

TRUE

Вызов функции завершился успешно.

FALSE

При вызове функции зафиксирована аппаратная ошибка.

#### Пример


```
PROGRAM PLC_PRG
VAR
  // Цвет
  led_color : FastwelBoard.LedsColor_t;
  // Блок формирования прямоугольных импульсов
  led_blinker : Util.BLINK;
END_VAR

// Формируем меандр с частотой 10 Гц
led_blinker(ENABLE:= TRUE, TIMELOW:= T#50MS, TIMEHIGH:= T#50MS);
// USER мигает с частотой меандра
led_color := SEL(led_blinker.OUT, FastwelBoard.LED_COLOR_NONE, FastwelBoard.LED_COLOR_GREEN);
FastwelBoard.SysBoardSetUserLed(led_color);
```

#### **6.3.3.4. SysBoardReboot**

Функция предназначена для программного перезапуска контроллера.

При вызове данной функции система исполнения останавливает и завершает работу текущего загруженного приложения, что позволяет приложению перед перезапуском контроллера закрыть файлы, сокеты, освободить запрошенные ресурсы и т.д. в обработчике системного события *PrepareExit*. Кроме того, при завершении работы текущего приложения будут вызваны методы FB\_Exit всех экземпляров функциональных блоков. Информация об обработчиках системных событий приведена в п. 4.5. Информация о методе FB\_Exit функциональных блоков приведена в п. 4.6.2.

	<p>Последовательность завершения работы приложения и перезапуска контроллера выполняется полностью асинхронно по отношению к задаче, в которой вызывается <i>SysBoardReboot</i>.</p> <p>В связи с этим необходимо обеспечивать однократный вызов данной функции вводом дополнительных проверок перед передачей ей управления. В проекте <i>FastwelBoardExample.project</i> (см. таблицу 4), поставляемом в <i>Fastwel PLC Application Toolkit</i>, демонстрируется вариант реализации дополнительной проверки путем анализа наличия двух соседних фронтов изменения состояния переключателя – заднего и переднего.</p>
---	--

Обратите внимание, что.

#### Прототип

**FUNCTION SysBoardReboot : BOOL**

#### Входные параметры

нет

#### Возвращаемый результат

**TRUE**

Вызов функции завершился успешно.

**FALSE**

При вызове функции зафиксирована ошибка.

#### Пример

```
PROGRAM PLC_PRG
VAR
    // блок чтения переключателей
    sw_reader : FastwelBoard.SWITCH_READER_x10;
    // детектор заднего фронта переключателя 9
    sw9_fedge : Standard.F_TRIG;
    // флаг успешного запуска последовательности перезапуска
    do_reboot : BOOL;
END_VAR

// с периодом 10 Гц читаем положение переключателей
sw_reader(Period := T#100MS);
// вызываем детектор заднего фронта переключателя 9
sw9_fedge(CLK := sw_reader.SwitchStates[9]);
IF sw9_fedge.Q AND NOT do_reboot THEN
    // если есть фронт и еще не начат перезапуск, начинаем
    do_reboot := FastwelBoard.SysBoardReboot();
    // больше делать нечего, выходим
    RETURN;
END_IF;
```

### 6.3.3.5. SysBoardGetBootReason

Функция предназначена для определения причины последнего запуска или перезапуска контроллера.

#### Прототип

```
FUNCTION SysBoardGetBootReason : BootReason_t
VAR_INPUT
    pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

#### Входные параметры

**pResult : POINTER TO RTS\_IEC\_RESULT;**

Необязательный параметр для получения результата вызова функции:

ERR\_OK – успешный вызов, причина определена;

ERR\_NOT\_SUPPORTED – функция не поддерживается;

другие значения, не равные ERR\_OK, соответствуют неудачному вызову функции.

#### Возвращаемый результат

**BOOT\_REASON\_UNKNOWN**

Неизвестная причина или функция не поддерживается на данном устройстве.

**BOOT\_REASON\_POWERUP**

Последний запуск контроллера произошел по причине включения питания.

**BOOT\_REASON\_SOFTWARE**

Последний запуск контроллера произошел после программного перезапуска из приложения, командой *reboot* оболочки ПЛК или из веб-конфигуратора.

**BOOT\_REASON\_WATCHDOG**

Последний запуск контроллера произошел по системному сторожевому таймеру. Обратите внимание, что в ряде случаев программный перезапуск контроллера завершается перезапуском по аппаратному сторожевому таймеру. Это характерно для ситуации, когда на контроллере функционирует приложение, потребляющее слишком много вычислительных ресурсов.

#### Пример

**PROGRAM PLC\_PRG**

**VAR**

*// Результат системного вызова*

*iec\_res : RTS\_IEC\_RESULT;*

*// Причина последнего запуска*

*boot\_reason : FastwelBoard.BootReason\_t;*

**END\_VAR**

*// Определяем причину, если это не было сделано ранее*

**IF** *boot\_reason* = *FastwelBoard.BOOT\_REASON\_UNKNOWN* **THEN**

*boot\_reason := FastwelBoard.SysBoardGetBootReason(ADR(iec\_res));*

**END\_IF**

### 6.3.4. Функциональные блоки

#### 6.3.4.1. SWITCH\_READER\_x10

Данный блок реализован на Structured Text и предназначен для периодического чтения текущего состояния десяти переключателей контроллера.

#### Входы

**Period : TIME := T#0MS;**

Определяет период чтения состояния переключателей. При равенстве T#0ms чтение производится в каждом цикле программной единицы, из которой вызывается экземпляр блока.



Вызов функции SysBoardReadSwitches, на базе которой реализован блок, выполняется около 350 мкс, поэтому не следует использовать период чтения менее 100 мс.

#### Выходы

**SwitchStates : ARRAY [1..SWITCHES\_NUMBER] OF BOOL;**

Массив логических состояний блока переключателей (FALSE – выключен, TRUE – включен). Константа SWITCHES\_NUMBER определена в данном функциональном блоке и равна 10.

**Failure : BOOL := TRUE;**

Признак ошибки доступа к блоку переключателей. Устанавливается в TRUE при аппаратном отказе блока переключателей.

Пример:

```
PROGRAM Example_SysBoard
VAR
  // Экземпляр блока чтения переключателей
  sw_reader : FastwelBoard.SWITCH_READER_x10;
  // Детектор переднего фронта переключателя 10
  sw10_redge : Standard.R_TRIG;

  // Результат системного вызова
  iec_res : RTS_IEC_RESULT;
  // Причина последнего запуска
  boot_reason : FastwelBoard.BootReason_t;
END_VAR

// с периодом 10 Гц читаем положение переключателей
sw_reader(Period := T#100MS);

// вызываем детектор переднего фронта переключателя 10
sw10_redge(CLK := sw_reader.SwitchStates[10]);
IF sw10_redge.Q THEN
  // обнаружен фронт, определяем причину последнего запуска
  boot_reason := FastwelBoard.SysBoardGetBootReason(ADR(iec_res));
END_IF
```

#### 6.3.4.2. USER\_LED\_CONTROL

Данный блок реализован на Structured Text и предназначен для управления светодиодным индикатором USER или USR1.

Входы

Color : LedsColor\_t;

Определяет цвет индикатора:

LED\_COLOR\_NONE – выключен;

LED\_COLOR\_RED – красный;

LED\_COLOR\_GREEN – зеленый.

BlinkPeriod : TIME := T#0MS;

Определяет период прерывистого свечения в мс. При равенстве T#0ms индикатор непрерывно светится выбранным цветом или выключен, если Color равен LED\_COLOR\_NONE.

Выходы

Нет.

Пример:

```
PROGRAM PLC_PRG
VAR
  // Переменная для установки цвета из формы визуализации Main
  led_color_selected : INT := 2;
  // Переменная для установки периода "мигания" из формы визуализации Main
  blink_last_period : DWORD := 500;
  // Переменная для включения/выключения "мигания" из формы визуализации Main
  led_blinking : BOOL := TRUE;
  // Экземпляр управления светодиодом USER
  led_ctrl : FastwelBoard.USER_LED_CONTROL;
END_VAR

VAR_TEMP
  blink_period : DWORD;
  led_color : FastwelBoard.LedsColor_t;
END_VAR
```

```

// Если включили мигание, используем последнее установленное значение периода
blink_period := SEL(led_blinking, 0, blink_last_period);

// Выбираем цвет
CASE led_color_selected OF
  1 : led_color := FastwelBoard.LED_COLOR_RED;
  2 : led_color := FastwelBoard.LED_COLOR_GREEN;
ELSE
  led_color := FastwelBoard.LED_COLOR_NONE;
END_CASE

// Вызываем блок управления светодиодом USER
led_ctrl(Color:= led_color, BlinkPeriod:= DWORD_TO_TIME(blink_period));

```

#### 6.3.4.3. USER\_LEDS\_CONTROL

Данный блок реализован на Structured Text и предназначен для управления светодиодными индикаторами USR1 (или USER) и USR2.

##### Входы

**SelectedLed : BoardLeds\_t := BoardLeds\_t.LED\_USER1;**

Определяет управляемый светодиодный индикатор:

LED\_USER – USER или USR1;

LED\_USER – USER или USR1;

LED\_USER2 – USR2

**Color : LedsColor\_t;**

Определяет цвет индикатора:

LED\_COLOR\_NONE – выключен;

LED\_COLOR\_RED – красный;

LED\_COLOR\_GREEN – зеленый.

**BlinkPeriod : TIME := T#0MS;**

Определяет период прерывистого свечения в мс. При равенстве T#0ms индикатор непрерывно светится выбранным цветом или выключен, если Color равен LED\_COLOR\_NONE.

##### Выходы

Нет.

### 6.4. Библиотека FastwelCore

#### 6.4.1. Общие сведения

Библиотека FastwelCore содержит функции, функциональные блоки и системные события с префиксом *Legacy*, специфические для платформ Fastwel, в том числе предназначенные для облегчения миграции приложений, разработанных в среде разработки CoDeSys 2.3 для контроллеров Fastwel I/O с системой исполнения приложений CoDeSys 2.3.

#### 6.4.2. Типы данных

##### 6.4.2.1. F\_PLCTL\_RESULT

Данный тип содержит перечисление результатов вызова функций библиотеки с префиксом *FwPlatform*: FwPlatformGetSerialNumber, FwPlatformSetSerialNumber, FwPlatformReset:

```

TYPE F_PLCTL_RESULT :
(
  (* успешное выполнение операции *)
  F_PLCTL_OK,
  (* операция не поддерживается *)
  F_PLCTL_NOT_SUPPORTED,
  (* недопустимое значение параметра *)
  F_PLCTL_INCORRECT_PARAM,
  (* системная ошибка *)
  F_PLCTL_GENERIC
);
END_TYPE

```

#### 6.4.2.2. F\_RESET\_MODE

Данный тип содержит перечисление кодов режима сброса, передаваемых функции FwPlatformReset:

```

TYPE F_RESET_MODE :
(
  (* Горячий сброс аналогично команде Online/Reset Warm
   * кроме RETAIN сегмента *)
  F_RESET_WARM:= 16#1ACF,
  (* Холодный сброс аналогично команде Online/Reset Cold
   * включая RETAIN сегмент *)
  F_RESET_COLD:= 16#2BFC,
  (* Перезапуск системы исполнения *)
  F_RESET_HARD:= 16#5EEE,
  (* Презапуск контроллера с переходом в безопасный режим *)
  F_RESET_SAFE:= 16#3C1D,
  (* Сброс в заводской режим аналогично команде Online/Reset Original*)
  F_RESET_ORIGINAL:= 16#4D55
);
END_TYPE

```

#### 6.4.2.3. F\_NETWORK\_IFADDR

Данный тип представляет расширенную информацию об адресе сетевого интерфейса:

```

TYPE F_NETWORK_IFADDR :
STRUCT
  name : STRING(32); (* Имя интерфейса *)
  addr : SOCKADDRESS; (* Основной IP-адрес *)
  mask : SOCKADDRESS; (* Маска подсети *)
  broadcast : SOCKADDRESS; (* Адрес направленного широковещательного запроса подсети *)
END_STRUCT
END_TYPE

```

Адрес переменной данного типа должен использоваться в качестве параметра при вызове функции F\_Net\_getInfo.

#### 6.4.2.4. F\_NETWORK\_MODE

Данный тип представляет режим работы сетевого адаптера. Сетевой адаптер содержит один или более сетевых интерфейсов, которые могут быть подключены к IP сети.

```

TYPE F_NETWORK_MODE :
(
  (* Режим не определен *)
  F_NETWORK_MODE_UNDEFINED := 0,
  (* Режим одной подсети (одна сетевая маска) на оба сетевых интерфейса *)
  F_NETWORK_MODE_ONESUBNET,
  (* Режим различных подсетей на интерфейсах *)
  F_NETWORK_MODE_DSA,
  (* Режим коммутатора *)
  F_NETWORK_MODE_SWITCH,
  (* Режим межконтроллерного кольца *)
  F_NETWORK_MODE_RING
);
END_TYPE

```

Значение данного типа возвращается при вызове функции F\_Net\_GetIpInfo для адаптера, номер которого (начиная с 0), передан в качестве первого параметра функции. Если адаптер с данным номером

не сконфигурирован в момент вызова функции `F_Net_GetIpInfo` или отсутствует на данном контроллере, функция `F_Net_GetIpInfo` возвращает значение `F_NETWORK_MODE_UNDEFINED`.

Информация о режимах работы сетевых интерфейсов, поддерживаемых контроллером, приведена в руководстве программиста на соответствующий контроллер.

#### 6.4.2.5. F\_LINK\_DESCRIPTOR



Данный тип не совместим с системой исполнения приложений контроллеров Fastwel на базе 64-разрядных процессоров.

На контроллерах на базе 64-разрядного процессора следует использовать тип данных `F_LINK_DESCRIPTOR_EX` из библиотеки `FastwelTaskExchange`.

Данный тип используется для представления информации о связи между задачами приложения при вызове функции `F_IecTasks_linkVariables` (см. п. 6.4.3.8) на контроллерах Fastwel на базе 32-разрядных процессоров:

TYPE `F_LINK_DESCRIPTOR` :

STRUCT

(\* Адрес переменной, которая должна быть связана с другой переменной.

Пример: `descr.variableAddress := ADR(SomeProgram.SomeVariable);` \*)

`variableAddress` : `DWORD`;

(\* Размер переменной, чей адрес указан в `variableAddress`.

Пример: `descr.variableSize := SIZEOF(SomeProgram.SomeVariable)` \*)

`variableSize` : `INT`;

(\* Ссылка на адрес программной единицы, которой принадлежит связываемая переменная, чей адрес указан в `variableAddress`.

Пример: `descr.pouRef := ADR(SomeProgram);` \*)

`pouRef` : `DWORD`;

END\_STRUCT

END\_TYPE

#### 6.4.2.6. F\_LINK\_RESULT

Данный тип содержит перечисление возможных результатов вызова функции `F_IecTasks_linkVariables`, устанавливающей связь между задачами (см. п. 6.4.3.8):

TYPE `F_LINK_RESULT` :

(

(\* Исходное, действий не выполнялось. \*)

`F_LINK_UNCERTAIN` := 0,

(\* Успех, связь установлена \*)

`F_LINK_OK`,

(\* В качестве `pSourceDescriptor` передана неправильная ссылка на переменную-источник данных \*)

`F_LINK_INVALID_SOURCE`,

(\* В качестве `pDestinationDescriptor` передана неправильная ссылка на переменную-получатель данных \*)

`F_LINK_INVALID_DESTINATION`,

(\* Неправильный размер переменной-источника данных \*)

`F_LINK_INVALID_SOURCE_LEN`,

(\* Неправильный размер переменной-получателя данных \*)

`F_LINK_INVALID_DESTINATION_LEN`,

(\* Несовпадение размеров переменных источника и получателя данных. \*)

`F_LINK_SOURCE_DESTINATION_LEN_NOT_EQUAL`,

(\* Попытка связать переменные, принадлежащие программным единицам, вызываемым из одной и той же задачи. \*)

`F_LINK_ONE_TASK_CONNECTION_NOT_ALLOWED`,

(\* Вызов функции произошел вне обработчика системного события `LegacyOnInit` или в режиме `Simulation`. \*)

`F_LINK_CONNECTION_ALLOWED_ON_INIT_INTARGET_ONLY`,

(\* Недостаточно системных ресурсов для связывания переменных \*)

`F_LINK_NOT_ENOUGH_RESOURCES`,

(\* `pDestinationDescriptor` ссылается на переменную, которая уже связана с другим источником данных. \*)

`F_LINK_INPORT_EXIST_FOR_DESTINATION`

);

END\_TYPE



#### 6.4.2.7. F\_TASK\_INFO

Адрес переменной данного типа должен быть передан в качестве параметра функции F\_IecTasks\_getInfo (см. п. 6.4.3.7):

```

TYPE F_TASK_INFO :
STRUCT
  (* Период циклической задачи в мкс или 16#FFFFFF для ациклической задачи. *)
  period_us : DWORD;
  (* Количество выполненных циклов задачи. *)
  cyclesCount : DWORD;
  (* Количество запаздываний циклической задачи. *)
  overrunsCount : DWORD;
  (* Минимальное время ввода данных и выполнения пользовательского кода в микросекундах. *)
  minExecutionTime_us : DWORD;
  (* Максимальное время ввода данных и выполнения пользовательского кода в микросекундах. *)
  maxExecutionTime_us : DWORD;
  (* Имя задачи. *)
  name : STRING(23);
  (* Счетчик микросекунд в момент последнего запуска задачи перед вызовом
    F_IecTasks_getInfo. *)
  startCycleTickCount_us : DWORD;
  (* Время ввода данных и выполнения пользовательского кода в микросекундах в цикле,
    предшествующем вызову F_IecTasks_getInfo. *)
  lastExecutionTime_us : DWORD;
END_STRUCT
END_TYPE

```

#### 6.4.3. Функции

##### 6.4.3.1. FwPlatformGetSerialNumber

Функция реализована на Structured Text и предназначена для получения серийного (заводского) номера контроллера путем вызова функции SysTargetGetSerialNumber.

##### Прототип

```

FUNCTION FwPlatformGetSerialNumber : F_PLCTL_RESULT
VAR_INPUT
  pValue : POINTER TO DWORD;
END_VAR

```

##### Входные параметры

pValue : POINTER TO DWORD

Указатель на переменную типа DWORD, в которую при успешном вызове функции будет помещено значение серийного номера.

Числовая часть серийного номера будет помещена в первые три байта переменной, адресуемой pValue. Старший байт содержит код временного периода изготовления. Для преобразования бинарного значения серийного номера в строку воспользуйтесь функцией FastwelCore.FwSerialNumberToString.

##### Возвращаемый результат

F\_PLCTL\_OK

Вызов функции завершился успешно.

F\_PLCTL\_INCORRECT\_PARAM

В качестве параметра pValue передано нулевое значение.

F\_PLCTL\_GENERIC

Ошибка при вызове SysTargetGetSerialNumber.

Пример

```

PROGRAM PLC_PRG
VAR
  // Сюда будет помещен серийный номер
  serial_number : DWORD;
  // Серийный номер, преобразованный в строку
  serial_number_string : STRING(32);
END_VAR

IF serial_number = 0 THEN
  // читаем серийный номер один раз (в случае успеха)
  IF FastwelCore.F_PLCTL_OK = FastwelCore.FwPlatformGetSerialNumber(ADR(serial_number)) THEN
    // преобразуем значение в строку
    FastwelCore.FwSerialNumberToString(serial_number, serial_number_string);
  END_IF
END_IF

```

**6.4.3.2. FwPlatformReset**

Функция реализована на Structured Text и предназначена для программного перезапуска приложения или системы исполнения приложения. Данная функция выполняется синхронно в контексте вызывающей задачи при всех типах перезапуска, кроме F\_RESET\_HARD.

Прототип

```

FUNCTION FwPlatformReset : F_PLCTL_RESULT
VAR_INPUT
  (* Тип сброса *)
  ResetMode : F_RESET_MODE;
  (* Сообщение при сбросе F_RESET_SAFE сохраняется в normdump.txt *)
  pMsg : POINTER TO STRING;
END_VAR

```

Входные параметры

**ResetMode : F\_RESET\_MODE**

Содержит тип сброса приложения или системы исполнения:

**F\_RESET\_WARM** – перезапуск приложения, аналогичный команде **Онлайн – Сброс**, выполняемой из среды разработки. После сброса все переменные, кроме энергонезависимых, принимают начальные значения;

**F\_RESET\_COLD** – перезапуск приложения, аналогичный команде **Онлайн – Сброс холодный**, выполняемой из среды разработки. После сброса все переменные, кроме сохраняемых энергонезависимых (RETAIN PERSISTENT), принимают начальные значения;

**F\_RESET\_HARD** – перезапуск системы исполнения, с точки зрения состояния переменных приложения идентичный **F\_RESET\_WARM**;

**F\_RESET\_SAFE** – перезапуск контроллера в принудительном (форсированном) безопасном режиме с сохранением информации в системном журнале. Информация о безопасном режиме приведена в руководстве по эксплуатации и руководстве программиста на контроллер;

**F\_RESET\_ORIGINAL** – перезапуск приложения, аналогичный команде **Онлайн – Сброс заводской**, выполняемой из среды разработки.

**pMsg : POINTER TO STRING**

Указатель на строку сообщения, которая будет отображена в системном журнале после перезапуска контроллера в форсированном безопасном режиме **F\_RESET\_SAFE**.

Возвращаемый результат

**F\_PLCTL\_OK**

Вызов функции завершился успешно.

**F\_PLCTL\_INCORRECT\_PARAM**

В качестве параметра ResetMode передано неправильное значение.

**F\_PLCTL\_GENERIC**

Ошибка при выполнении сброса текущего приложения контроллера.

Пример

```

PROGRAM PLC_PRG
VAR
  // Переменная для выбора типа перезапуска
  selected_reset_type : INT := 0;
  // Переменная, передний фронт которой инициирует перезапуск
  do_reset : BOOL;
  // Режим перезапуска
  reset_mode : FastwelCore.F_RESET_MODE;
  // Детектор фронта для выполнения сброса
  sw_reset_edge : Standard.R_TRIG;

  // Переменная, передний фронт которой инициирует переход в безопасный режим
  do_safe : BOOL;
  // Сообщение для отображения в журнале
  safe_mode_message : STRING(80) := 'SAFE mode forced from the user application!';
  // Детектор фронта для падения в безопасный режим
  sw_safe_edge : Standard.R_TRIG;
END_VAR

// определяем передний фронт сигналов сброса и перехода в безопасный режим
sw_reset_edge(CLK := do_reset);
sw_safe_edge(CLK := do_safe);

IF sw_safe_edge.Q THEN
  // нужно перейти в безопасный режим
  // на всякий случай блокируем программный перезапуск
  do_safe := TRUE;
  // падаем
  FastwelCore.FwPlatformReset(FastwelCore.F_RESET_SAFE, ADR(safe_mode_message));
END_IF

IF NOT do_safe AND sw_reset_edge.Q THEN
  // если не идем в SAFE-режим и был сигнал перезапуска, выбираем режим перезапуска
  CASE selected_reset_type OF
    0: reset_mode := FastwelCore.F_RESET_WARM;
    1: reset_mode := FastwelCore.F_RESET_COLD;
    2: reset_mode := FastwelCore.F_RESET_ORIGINAL;
  ELSE
    reset_mode := FastwelCore.F_RESET_WARM;
  END_CASE
  // и выполняем перезапуск
  FastwelCore.FwPlatformReset(reset_mode, 0);
END_IF

```

**6.4.3.3. F\_Net\_getAdaptersCount**

Данная функция возвращает количество сетевых адаптеров на данном контроллере. Понятие сетевого адаптера определено в руководстве по эксплуатации и руководстве программиста на контроллер и означает устройство, содержащее один или более интерфейсов, которые могут быть подключены к одной или более IP сетям.



Время выполнения функции F\_Net\_getAdaptersCount может составлять единицы миллисекунд, в связи с чем рекомендуется вызывать ее однократно при запуске приложения.

Прототип

```
FUNCTION F_Net_getAdaptersCount : UINT
```

Входные параметры

Нет.

Возвращаемый результат

Количество сетевых адаптеров на данном контроллере.

Пример

см. п. 6.4.3.5.

**6.4.3.4. F\_Net\_getInterfacesCount**

Данная функция возвращает количество интерфейсов, принадлежащих заданному сетевому адаптеру.



Время выполнения функции F\_Net\_getInterfacesCount может составлять единицы миллисекунд, в связи с чем рекомендуется вызывать ее однократно при запуске приложения.

Прототип

```
FUNCTION F_Net_getInterfacesCount : UINT
VAR_INPUT
  adapterNum : UINT;
  pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

Входные параметры

**adapterNum : UINT**

Номер сетевого адаптера, начиная с 0.

**pResult : POINTER TO RTS\_IEC\_RESULT**

Указатель на переменную для получения результата вызова функции:

ERR\_OK – вызов завершился успешно;

ERR\_PARAMETER – функции передан номер адаптера, отсутствующего на данном контроллере.

Возвращаемый результат

Количество сетевых интерфейсов, принадлежащих данному контроллеру. В коммутируемых режимах функция возвращает 1, а в некоммутируемых – 2. Перед использованием возвращаемого результата следует проверить результат вызова pResult.

Пример

см. п. 6.4.3.5.

**6.4.3.5. F\_Net\_getIpInfo**

Функция используется для получения информации о режиме работы сетевого адаптера с номером adapterNum и IP параметров сетевого интерфейса с номером interfaceNum, принадлежащего данному адаптеру. Возвращает F\_NETWORK\_MODE\_UNDEFINED в случае ошибки аргументов или при отсутствии сконфигурированного интерфейса с номером interfaceNum у адаптера adapterNum. Причина ошибки может быть выяснена путем анализа результата, передаваемого через pResult.

Время выполнения функции составляет около 2,5 мс, в связи с чем рекомендуется вызывать ее однократно при запуске приложения

Прототип

```
FUNCTION F_Net_getIpInfo : F_NETWORK_MODE
VAR_INPUT
  adapterNum : UINT;
  interfaceNum : UINT;
  pIfAddr : POINTER TO F_NETWORK_IFADDR;
  pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

Входные параметры**adapterNum** : UINT

Номер сетевого адаптера, начиная с 0.

**interfaceNum** : UINT

Номер сетевого интерфейса, начиная с 0, принадлежащего адаптеру adapterNum.

**pIfAddr** : POINTER TO F\_NETWORK\_IFADDR

Указатель на структуру, в которой будут возвращены IP параметры интерфейса.

**pResult** : POINTER TO RTS\_IEC\_RESULT

Указатель на переменную для получения результата вызова функции:

ERR\_OK – вызов завершился успешно;

ERR\_PARAMETER – передан неправильный параметр;

ERR\_PENDING – IP параметры интерфейса не определены. Если интерфейс с заданным номером точно присутствует в конфигурации контроллера, то данное значение свидетельствует об установленном режиме динамического назначения IP-адреса (DHCP) для выбранного интерфейса, при этом процесс назначения адреса выбранному интерфейсу пока не завершен.

Возвращаемый результат

Функция возвращает текущий режим сетевого интерфейса:

F\_NETWORK\_MODE\_UNDEFINED – вызов функции завершился неудачей или IP-адрес в момент вызова еще не распределен;

F\_NETWORK\_MODE\_ONESUBNET – IP-адреса сетевых интерфейсов находятся в одной подсети;

F\_NETWORK\_MODE\_DSA – IP-адреса сетевых интерфейсов находятся в разных подсетях;

F\_NETWORK\_MODE\_SWITCH – режим *Коммутатор*, поддерживается на контроллерах со встроенным коммутатором Ethernet;

F\_NETWORK\_MODE\_RING – режим *Кольцо*, поддерживается на контроллерах со встроенным коммутатором Ethernet.

Пример

PROGRAM PLC\_PRG

VAR

// Флаг однократного вызова

run\_once : BOOL;

// Режимы интерфейсов

net\_mode : ARRAY [0..1] OF FastwelCore.F\_NETWORK\_MODE;

// IP-параметры интерфейсов

net\_iface\_addr : ARRAY [0..1] OF FastwelCore.F\_NETWORK\_IFADDR;

// Количество адаптеров

adapters\_count : UINT;

// Количество интерфейсов адаптера с номером -

ifaces\_count : UINT;

// Результат системного вызова

iec\_res : RTS\_IEC\_RESULT;

END\_VAR

VAR\_TEMP

idx : UINT;

END\_VAR

IF NOT run\_once THEN

// вызываем однократно

run\_once := TRUE;

// получаем количество адаптеров

adapters\_count := FastwelCore.F\_Net\_getAdaptersCount();

IF adapters\_count = 1 THEN

// если адаптер 1, получаем количество интерфейсов для адаптера 0

ifaces\_count := FastwelCore.F\_Net\_getInterfacesCount(0, ADR(iec\_res));

IF CmpErrors.Errors.ERR\_OK = iec\_res THEN

```

// получаем IP-параметры интерфейсов адаптера 0
FOR idx := 0 TO SIZEOF(net_iface_addr)/SIZEOF(net_iface_addr[0]) - 1 DO
  net_mode[idx] := FastwelCore.F_Net_getIpInfo(0, idx, ADR(net_iface_addr[idx]), 0);
END_FOR
END_IF
END_IF
END_IF

```

#### 6.4.3.6. F\_IecTasks\_getCount

Данная функция реализована для облегчения миграции проектов, разработанных для контроллеров Fastwel с системой исполнения приложений CoDeSys 2.3, и возвращает суммарное количество циклических и ациклических задач приложения.

##### Прототип

```
FUNCTION F_IecTasks_getCount : WORD
```

##### Входные параметры

Нет.

##### Возвращаемый результат

Общее количество циклических и ациклических задач текущего приложения контроллера.

#### 6.4.3.7. F\_IecTasks\_getInfo

Данная функция возвращает статистическую информацию о циклической или ациклической задаче, номер которой передан в качестве второго параметра. Если задача с заданным номером отсутствует в системе, функция возвращает 0.

При анализе статистики циклических задач следует вызывать данную функцию в одной из программных единиц, вызываемых из анализируемой задачи, передавая в taskNumber значение 16#FFFF (или 65535).

При анализе статистики ациклических задач следует вызывать данную функцию в одной из программных единиц, вызываемых из анализируемой задачи, передавая в качестве taskNumber номер задачи в списке **Конфигурация задач** приложения.

##### Прототип

```

FUNCTION F_IecTasks_getInfo : WORD
VAR_INPUT
  pTaskInfo : POINTER TO F_TASK_INFO;
  taskNumber : WORD;
END_VAR

```

##### Входные параметры

pTaskInfo : POINTER TO F\_TASK\_INFO

Адрес переменной типа F\_TASK\_INFO (см. п. 6.4.2.7).

taskNumber : WORD;

Номер задачи, начиная с 0.

##### Возвращаемый результат

0 – в системе отсутствует задача с заданным номером; 1 – в pTaskInfo записана статистическая информация о выполнении задачи.

##### Пример

```

PROGRAM PLC_PRG
VAR
  task_info : FastwelCore.F_TASK_INFO;
END_VAR

// Получаем статистическую информацию о текущей циклической задаче
FastwelCore.F_IecTasks_getInfo(ADR(task_info), 16#FFFF);

```

#### 6.4.3.8. F\_IecTasks\_linkVariables

Традиционно при разработке приложений для программируемых логических контроллеров применяется подход, при котором все входные переменные (их значения получаются приложением от входных каналов модулей ввода-вывода и входящих коммуникационных объектов внешней сети) и большинство внутренних переменных алгоритма, реализуемого приложением, делаются глобальными, т.е. доступными любой программной единице приложения.

Если система исполнения контроллера выполняет все программные единицы последовательно, то указанный подход вполне приемлем, особенно для небольших приложений.

При разработке приложения для контроллера с многозадачной системой исполнения использование глобальных переменных, чтение и запись которых может осуществляться в разных, параллельно исполняющихся задачах, может привести к серьезным и, в ряде случаев, трудновоспроизводимым ошибкам.

Пусть, например, некоторая переменная *gMyVariable* объявлена в списке *GVL* (*VAR\_GLOBAL*), и ее значение вычисляется в программе *PRG1*, исполняющейся под управлением циклической задачи *Task1*, для которой заданы период исполнения 10 мс и приоритет 1. Пусть значение *gMyVariable* используется в алгоритме, выполняемом другой программой *PRG2*, функционирующей под управлением другой циклической задачи *Task2*, имеющей период исполнения 30 мс и приоритет 15, т.е. более низкий, чем *Task1*. Циклограмма приложения представлена на рисунке 151.

В цикле *Task1*, начавшемся в момент (5 мс), *PRG1* вычисляет значение *gMyVariable*, которое, к примеру, становится равным 1. Далее в момент, близкий к (10 мс), начинается исполнение задачи *Task2*, которая запускает *PRG2*. *PRG2* использует *gMyVariable* в реализуемом алгоритме в течение промежутка времени между моментами (10 мс) и (30 мс), полагая, что ее значение равно 1.

При наступлении момента (15 мс) более приоритетная *Task1* вытесняет *Task2* (приостанавливает *Task2*) и вызывает *PRG1*, которая вновь вычисляет *gMyVariable*, после чего управление возвращается в ту точку кода *PRG2*, где была вытеснена *Task2*. Как видно, значение *gMyVariable* теперь не равно 1, каковым оно было при вызове *PRG2* в начале цикла *Task2*, а это значит, что алгоритм *PRG2* скорее всего будет работать неправильно.

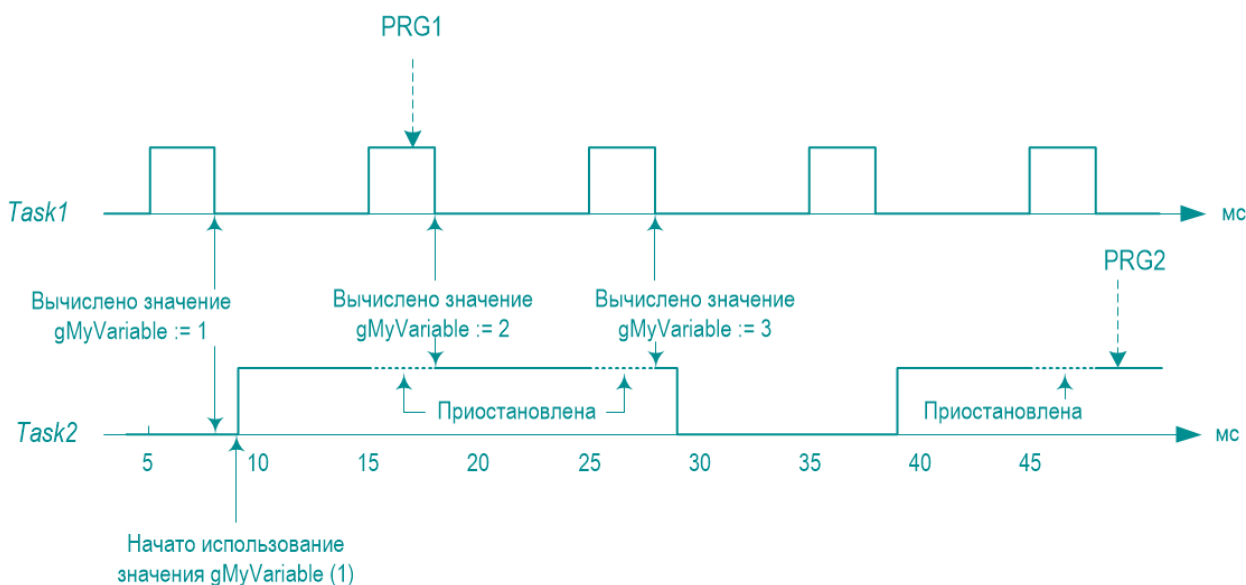


Рисунок 151 – Циклограмма выполнения двух задач с доступом к одной глобальной переменной

Если же *PRG2* в процессе работы изменяет *gMyVariable*, то ее изменения "отменяются" всякий раз, когда задача *Task1* вытесняет *Task2*, и *PRG1* изменяет *gMyVariable*.

Решение данной проблемы путем копирования *gMyVariable* в какую-нибудь внутреннюю или входную переменную *PRG2* в начале каждого цикла *Task2* является корректным только для переменных длиной не более разрядности процессора, которая в контроллере CPM723-01 составляет 32 бита (4 байта). Переменные большей длины, включая *LREAL*, массивы, строки и структуры, длина которых превышает 4 байта, невозможно скопировать атомарно – т.е. исключив возможность вытеснения во время копирования другой, более приоритетной, задачей, которая изменит копируемое значение, в



результате чего во внутреннюю переменную PRG2 будет скопирована часть старого значения и часть нового, вычисленного "вклинившейся" более приоритетной задачей.

Для того, чтобы пользователь был способен избежать указанных явлений при обмене данными между программами, вызываемыми из разных задач, в библиотеку FastwelCore включена функция F\_IecTasks\_linkVariables, которая предназначена для связи переменных одинакового размера, принадлежащих программам, вызываемым из разных задач. При этом механизм межзадачного обмена данными аналогичен описанному в п. 4.3.2, за исключением того, что при связывании двух задач создается отдельный канал межзадачного обмена с отдельным буфером, размер которого равен размеру связываемых переменных:

1. Для переменной связываемой задачи, которая будет выступать в качестве источника данных, создается выходной порт.
2. Для переменной другой связываемой задачи, которая будет выступать в качестве получателя данных, создается входной порт.
3. Создается канал обмена с отдельным буфером, размер которого равен размеру связываемых переменных.
4. Выходной порт связывается с каналом в качестве источника, а входной порт – в качестве приемника данных.

В процессе работы задача, чья переменная связана с переменной другой задачи в качестве приемника данных, перед вызовом корневой программной единицы последовательно читает все свои входные порты, среди которых также оказываются и созданные при вызове F\_IecTasks\_linkVariables. При чтении порта, когда доступ по записи к связанному с ним каналу блокирован, значение, находящееся в буфере канала, копируется в переменную-приемник данных. Задача, чья переменная связана с переменной другой задачи в качестве источника данных, после вызова корневой программной единицы последовательно пишет во все свои выходные порты, среди которых оказываются и созданные при вызове F\_IecTasks\_linkVariables. При записи в выходной порт, когда доступ по чтению к связанному с ним каналу блокирован, значение переменной-источника данных копируется в буфер канала.



Функция F\_IecTasks\_linkVariables и F\_IecTasks\_linkVariablesEx из библиотеки FastwelTaskExchange должна вызываться только из обработчика системного события *LegacyOnInit*.

#### Прототип

```
FUNCTION F_IecTasks_linkVariables : F_LINK_RESULT
VAR_INPUT
    pSourceDescriptor : POINTER TO F_LINK_DESCRIPTOR;
    pDestinationDescriptor : POINTER TO F_LINK_DESCRIPTOR;
END_VAR
```

#### Входные параметры

**pSourceDescriptor : POINTER TO F\_LINK\_DESCRIPTOR**

Указатель на описатель переменной, которая будет использоваться в межзадачном обмене в качестве источника данных.

**pDestinationDescriptor : POINTER TO F\_LINK\_DESCRIPTOR**

Указатель на описатель переменной, которая будет использоваться в межзадачном обмене в качестве получателя данных.

#### Возвращаемый результат

**F\_LINK\_UNCERTAIN**

Зарезервированное значение, которое присваивается переменной типа F\_LINK\_RESULT по умолчанию пока еще не выполнено никаких действий по связыванию.

**F\_LINK\_OK**

Связывание выполнено успешно.

**F\_LINK\_INVALID\_SOURCE**

Неправильный описатель переменной-источника данных. Ситуации:

- pSourceDescriptor равен 0;
- адрес переменной (pSourceDescriptor^.variableAddress) равен 0;
- переменная находится во входном (%I\*) или выходном (%Q\*), %M\* или RETAIN-сегментах;
- заданная ссылка (pSourceDescriptor^.pouRef) определяет программную единицу, которая не относится к множеству программных единиц, вызываемых из каких-либо циклических или ациклических задач приложения.

#### **F\_LINK\_INVALID\_DESTINATION**

Неправильный описатель переменной-получателя данных. Ситуации:

- pDestinationDescriptor равен 0;
- адрес переменной (pDestinationDescriptor^.variableAddress) равен 0;
- переменная находится во входном (%I\*) или выходном (%Q\*), %M\* или RETAIN-сегментах;
- заданный ссылка (pDestinationDescriptor^.pouRef) определяет программную единицу, которая не относится к множеству программных единиц, вызываемых из каких-либо циклических или ациклических задач приложения.

#### **F\_LINK\_INVALID\_SOURCE\_LEN**

Неправильная длина переменной-источника данных (0 или более размера глобальной области данных)

#### **F\_LINK\_INVALID\_DESTINATION\_LEN**

Неправильная длина переменной-приемника данных (0 или более размера глобальной области данных)

#### **F\_LINK\_SOURCE\_DESTINATION\_LEN\_NOT\_EQUAL**

Отличие длин переменных-источника и приемника данных. Они должны быть равными друг другу и не равными нулю.

#### **F\_LINK\_ONE\_TASK\_CONNECTION\_NOT\_ALLOWED**

Попытка связать переменные, принадлежащие POU, которые вызываются из одной и той же задачи.

#### **F\_LINK\_CONNECTION\_ALLOWED\_ON\_INIT\_INTARGET\_ONLY**

Попытка вызова данной функции в месте, отличном от обработчика события *LegacyOnInit*.

#### **F\_LINK\_NOT\_ENOUGH\_RESOURCES**

Не хватило системных ресурсов для связывания.

#### **F\_LINK\_INPORT\_EXIST\_FOR\_DESTINATION**

Переменная, заданная в качестве получателя данных вторым параметром, уже связана с другой (или этой же) переменной-источником данных.

#### Пример

Применение функции F\_IecTasks\_linkVariables иллюстрируется примером в проекте TasksExchange.project, поставляемом в Fastwel PLC Application Toolkit (см. таблицу 4).



На контроллерах Fastwel на базе 64-разрядных процессоров следует использовать функцию F\_IecTasks\_linkVariablesEx и тип данных F\_LINK\_DESCRIPTOR\_EX из библиотеки FastwelTaskExchange.

#### **6.4.3.9. FwCheckSum16**

Функция предназначена для вычисления 16-разрядной контрольной суммы содержимого участка памяти, начальный адрес которого и размер переданы в качестве первого и второго параметров соответственно.



Размер участка памяти, обрабатываемый функцией FwChecksum16, ограничен максимальным значением типа WORD: 16#FFFF (65535).

#### Прототип

```
FUNCTION FwChecksum16 : WORD
VAR_INPUT
  pBuffer : POINTER TO BYTE;
  bufferLength : WORD;
  startChecksum : WORD;
END_VAR
```

#### Входные параметры

**pBuffer : POINTER TO BYTE**

Указатель на участок памяти, для содержимого которого требуется вычислить контрольную сумму.

**bufferLength : WORD**

Размер участка.

**startChecksum : WORD**

Начальное значение контрольной суммы. Данный параметр может использоваться при вычислении общей контрольной суммы нескольких несмежных участков памяти.

#### Возвращаемый результат:

16-разрядная сумма всех байт заданного участка памяти плюс startChecksum.

#### **6.4.3.10. FwChecksum32**

Функция предназначена для вычисления 32-разрядной контрольной суммы содержимого участка памяти, начальный адрес которого и размер переданы в качестве первого и второго параметров соответственно.

```
FUNCTION FwChecksum32 : DWORD
VAR_INPUT
  pBuffer : POINTER TO BYTE;
  bufferLength : DWORD;
  startChecksum : DWORD;
END_VAR
```

#### Входные параметры

**pBuffer : POINTER TO BYTE**

Указатель на участок памяти, для содержимого которого требуется вычислить контрольную сумму.

**bufferLength : DWORD**

Размер участка.

**startChecksum : DWORD**

Начальное значение контрольной суммы. Данный параметр может использоваться при вычислении общей контрольной суммы нескольких несмежных участков памяти.

#### Возвращаемый результат

32-разрядная сумма всех байт заданного участка памяти плюс startChecksum.

#### **6.4.3.11. FwCRC16**

Функция предназначена для вычисления 16-разрядной циклической контрольной суммы содержимого участка памяти, начальный адрес которого и размер переданы в качестве первого и второго параметров соответственно.



Размер участка памяти, обрабатываемый функцией FwCRC16, ограничен максимальным значением типа WORD: 16#FFFF (65535).

Циклическая контрольная сумма вычисляется в соответствии с указаниями п. 6.2.2 спецификации *MODBUS over Serial Line. Specification and Implementation Guide. V1.02.*

```
FUNCTION FwCRC16 : WORD
VAR_INPUT
    pBuffer : POINTER TO BYTE;
    bufferLength : WORD;
END_VAR
```

#### Входные параметры

**pBuffer : POINTER TO BYTE**

Указатель на участок памяти, для содержимого которого требуется вычислить CRC16.

**bufferLength : WORD**

Размер участка.

#### Возвращаемый результат

16-разрядная CRC всех байт заданного участка памяти.

### **6.4.3.12. FwCRC32**

Функция предназначена для вычисления 32-разрядной циклической контрольной суммы содержимого участка памяти, начальный адрес и размер которого переданы в качестве первого и второго параметров соответственно.

Циклическая контрольная сумма вычисляется с использованием полинома:  

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

```
FUNCTION FwCRC32 : DWORD
VAR_INPUT
    pBuffer : POINTER TO BYTE;
    bufferLength : DWORD;
    startCRC : DWORD;
END_VAR
```

#### Входные параметры

**pBuffer : POINTER TO BYTE**

Указатель на участок памяти, для содержимого которого требуется вычислить CRC16.

**bufferLength : DWORD**

Размер участка.

**startCRC : DWORD**

Начальное значение CRC32.

#### Возвращаемый результат

32-разрядная CRC всех байт заданного участка памяти, вычисленная с использованием полинома:  

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

### **6.4.3.13. FwSerialNumberToString**

Данная функция предназначена для преобразования в строку 32-разрядного значения серийного номера, полученного путем вызова функции FwPlatformGetSerialNumber (см. п. 6.4.3.1).

Прототип

```

FUNCTION FwSerialNumberToString : INT
VAR_INPUT
    // Серийный номер в исходном формате
    SerialNumber : DWORD;
END_VAR

VAR_IN_OUT
    // Результат будет помещен сюда
    ResultString : STRING(32);
END_VAR

```

Входные параметры

**SerialNumber : DWORD;**

Серийный номер, полученный вызовом функции FwPlatformGetSerialNumber.

**ResultString : STRING(32);**

Ссылка на строковую переменную для размещения результата преобразования.

Возвращаемый результат

Длина результирующей строки или 0, если не удалось выполнить преобразование.

Пример

см. п. 6.4.3.1.

**6.4.4. Функциональные блоки****6.4.4.1. CALL\_PERIOD\_GETTER**

Функциональный блок предназначен для определения периода вызова программной единицы, из которой вызывается экземпляр данного блока. Период вызова, как правило, требуется определять при реализации алгоритмов фильтрации и управления.

Выходная переменная Ready устанавливается в TRUE после успешного определения периода вызова, значение которого помещается в выходную переменную Period.

Если блок вызывается из циклической задачи, период будет определен при первом же вызове.

Входы

**CurrentTime : TIME := T#0ms;**

Счетчик миллисекунд от запуска контроллера, полученный вызовом функции TIME() в вызывающей программной единице. При равенстве T#0ms функция TIME() вызывается внутри блока.

Выходы

**Ready : BOOL;**

Признак готовности результата определения периода вызова.

**Period : TIME;**

Период вызова (в миллисекундах) программной единицы, из которой вызывается экземпляр данного блока.

Пример:

```

PROGRAM PLC_PRG
VAR
    dwCallPeriod : TIME;
    call_period_blk : FastwelCore.CALL_PERIOD_GETTER;
END_VAR

```

```
// В циклических задачах не нужно передавать TIME()
// Этот код будет работать в любых задачах.
call_period_blk( CurrentTime := TIME());

IF call_period_blk.Ready THEN
  // В циклической задаче будет готовность при первом же вызове
  dwCallPeriod := call_period_blk.Period;
END_IF
```

#### 6.4.4.2. RETAIN\_VAR\_ENGINE

Блок предназначен для сохранения значений переменной в файле во время работы приложения и последующего восстановления последнего сохраненного значения переменной при включении питания.

В качестве переменной может использоваться внутренняя переменная программной единицы типа PROGRAM, кроме VAR\_TEMP, VAR\_INPUT, VAR\_OUTPUT и VAR\_IN\_OUT, или глобальная переменная любого типа.

При сохранении глобальной переменной (VAR\_GLOBAL) самый первый вызов блока должен быть произведен до запуска циклических и ациклических задач в обработчике системного события *LegacyOnInit*. При использовании внутренней переменной программы экземпляр блока должен быть вызван до остальных инструкций данной программы.

При первом вызове значения входных параметров запоминаются во внутренних переменных блока, после чего выполняется поиск файла InputFileName.

Если файл обнаружен, его размер сравнивается с InputUserSize + 4 (размер поля контрольной суммы). При совпадении размера проверяется целостность содержимого файла путем проверки контрольной суммы, после чего сохраненное значение копируется в переменную, указатель на которую передан параметром InputUserPointer.

Если файл не обнаружен или его размер не соответствует InputUserSize, либо при нарушении целостности содержимого выполняется повторное создание данного файла, после чего в него записывается текущее значение переменной, указатель на которую передан в качестве параметра InputUserPointer.

При последующих вызовах входные параметры игнорируются и выполняются следующие действия:

1. Проверяется, истек ли минимальный период сохранения, заданный параметром InputMinimumPeriod.
2. При истечении минимального периода сохранения сравниваются текущее значение переменной, указатель на которую передан параметром InputUserPointer, с ранее сохраненным значением.
3. При любом изменении значения переменной, новое значение сохраняется в блоке и в файле InputFileName.

#### Входы

**InputUserPointer : POINTER TO BYTE**

Указатель на переменную, подлежащую сохранению и восстановлению при включении питания контроллера. В качестве переменной может использоваться внутренняя переменная программной единицы типа PROGRAM, кроме VAR\_TEMP, VAR\_INPUT, VAR\_OUTPUT и VAR\_IN\_OUT, или глобальная переменная любого типа. При сохранении глобальной переменной (VAR\_GLOBAL) самый первый вызов блока должен быть произведен до запуска циклических и ациклических задач в обработчике системного события *LegacyOnInit*. При использовании внутренней переменной программы экземпляр блока должен быть вызван до остальных инструкций внутри данной программы.

**InputUserSize : INT**

Размер переменной, указатель на которую передается параметром InputUserPointer. Не должен превышать константу RETAIN\_MAX\_LEN (4096 байт)

**InputMinimumPeriod : TIME**

Минимальный период сохранения значения переменной, указатель на которую передается параметром InputUserPointer. Таким образом, сохранение значения переменной произойдет в случае, если оно изменилось и с момента последней записи прошло не менее InputMinimumPeriod.

**InputFileName : STRING(80)**

Имя файла, в котором будет сохраняться значение переменной. Должно быть задано символами латинского алфавита в виде пути и имени файла. Все подкаталоги в пути к файлу должны быть созданы заранее.

#### Выходы

**Valid : BOOL**

Признак корректного функционирования блока.

**RetainCacheCount : DWORD**

Количество проверок сохраняемой переменной за время работы, при которых ее значение не менялось.

**RetainSuccessCount : DWORD**

Количество успешных сохранений переменной.

**RetainErrorCount : DWORD**

Количество ошибок.

#### Пример

Применение блока **RETAIN\_VAR\_ENGINE** иллюстрируется примером в проекте **FileRetainer.project**, поставляемом в **Fastwel PLC Application Toolkit** (см. таблицу 4).

## **6.5. Библиотеки доступа к файлам и каталогам SysFile и SysDir**

### **6.5.1. Общие сведения**

Информация о дисковых накопителях, входящих в состав контроллера, приведена в документации на контроллер. Настоящий подраздел содержит информацию об особенностях реализации доступа к дисковым накопителям контроллера из приложений МЭК 61131-3.

Файлы системы исполнения расположены в т.н. *корневом каталоге системы исполнения*, доступ к которому обеспечивается при подключении к контроллеру по протоколу FTP, FTPS, SFTP или из IDE МЭК 61131-3 с правами учетной записи *Administrator*. Кроме того, в корневой каталог системы исполнения загружаются системные файлы со страницы **Система** веб-конфигуратора контроллера.

В корневом каталоге системы исполнения имеется подкаталог *user*, далее называемый *каталогом пользователя*, который предназначен для создания, удаления, записи и чтения файлов и каталогов из приложения IDE МЭК 61131-3 средствами системных библиотек SysFile и SysDir.

При подключении к контроллеру по протоколу FTP, FTPS, SFTP с правами учетной записи *Everyone* обеспечивается доступ только к каталогу пользователя.

Информация об учетных записях и правах доступа приведена в п. 5.3 настоящего документа.

Для доступа к файлам и каталогам в каталоге пользователя при помощи функций библиотек SysFile и SysDir должны использоваться *относительные* пути. Относительным является путь, первый символ которого не является разделителем пути '/' (прямой слэш). Если первым символом пути является разделитель '/', то такой путь называется *абсолютным*.

Если к контроллеру подключен один или более USB-накопителей, то доступ к ним осуществляется через подкаталоги с именами *usbNpM* в каталоге пользователя, где N – номер гнезда USB, а M – номер раздела на накопителе. Например, если накопитель установлен в нижнее гнездо, имеет один раздел и содержит на нем файл *my\_app\_settings.txt*, то для доступа к файлу из приложения должен использоваться путь:

**szFilePath : STRING := 'usb3p1/my\_app\_settings.txt';**

Если в контроллер установлена карта MicroSD, то доступ к ней осуществляется через подкаталог с именем *microsd1*, появляющийся в каталоге пользователя *user*.




Пусть, например, на карте microSD, установленной в гнездо контроллера, имеется каталог *data*, в котором расположен файл *my\_app\_settings.txt*. Тогда для доступа к этому файлу из приложения должен использоваться относительный путь:



```
szFilePath : STRING := 'microsd1/data/my_app_settings.txt';
```

Если, требуется создать или прочитать файл в каталоге пользователя, то в пути достаточно указать имя файла:

```
szFileInUserFolder : STRING := 'my_file.dat';
```

	При разработке приложений, в которых требуется создавать каталоги и файлы, рекомендуется использовать только относительные пути, поскольку для них гарантируется неизменность местоположения в течение всего жизненного цикла изделия, а также возможность переноса приложений на другие контроллеры Fastwel с системой исполнения приложений МЭК 61131-3.
	Если требуется вызывать функции библиотек SysDir и SysFile из циклических или ациклических задач, абсолютное значение приоритета задачи должно быть от 15 и выше. При этом в случае необходимости частого или периодического доступа к файлам или каталогам или при больших объемах сохраняемых данных в качестве целевого накопителя следует использовать съемные дисковые накопители.
	Съемные дисковые накопители (карты microSD и USB-накопители) при поставке обычно отформатированы с использованием файловой системы FAT или FAT32, поэтому отключение питания контроллера в момент выполнения операции записи в файл, требующей увеличения размера файла, в момент создания или удаления файла и в момент закрытия файла, может привести к повреждению файловой системы на карте. В связи с этим при записи бинарных данных рекомендуется использовать файлы фиксированного размера, заранее установленного при помощи функции SysFileSetPos.

Пример работы с файлом фиксированного размера:

```
VAR
```

```

// флаг необходимости записать данные из binary_data
bFileWritten : BOOL;
// путь к файлу на карте
szBinFileName : STRING := 'microsd1/my_data.bin';
// хэндл файла
h_wr_file : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
// режим открытия
wr_file_access_mode : SysFile.ACCESS_MODE;
// требуемый фиксированный размер файла
required_file_size : __XWORD := 65536;
// маркер в последних 4 байтах
marker_data : DWORD := 16#ACEDFADE;
// буфер для записи в файл
binary_data : ARRAY [1..100] OF BYTE := [100(16#55)];
// размер записи
data_size : DWORD;

```

```
END_VAR
```

```
VAR_TEMP
```

```

// размер записанных данных
written_size : __XWORD;
// текущий размер файла
current_size : __XWORD;
// результат операции
iec_write_result : RTS_IEC_RESULT;

```

```
END_VAR
```

```
// Если нужно записать данные, и карта вставлена
```

```
IF NOT bFileWritten AND MMCExists() THEN
```

```

// получаем текущий размер файла

```

```
current_size := SysFile.SysFileGetSize(szBinFileName, ADR(iec_write_result));
```

```
IF CmpErrors.Errors.ERR_OK <> iec_write_result THEN
```

```

// файл не найден, будет создавать для чтения и записи

```

```
wr_file_access_mode := SysFile.AM_WRITE_PLUS;
```

```
ELSE
```

```

// файл имеется, будем открывать для чтения и записи

```

```
wr_file_access_mode := SysFile.AM_READ_PLUS;
```

```
END_IF
```

```

// открываем
h_wr_file := SysFile.SysFileOpen(szBinFileName, wr_file_access_mode, ADR(iec_write_result));

IF RTS_INVALID_HANDLE <> h_wr_file THEN
    // открыли, смотрим, не требуется ли сделать правильный размер
    IF current_size <> required_file_size THEN
        // встаем в требуемый конец минус 4 байта на маркер
        iec_write_result := SysFile.SysFileSetPos(h_wr_file,
                                                    required_file_size - SIZEOF(marker_data));
        IF CmpErrors.Errors.ERR_OK = iec_write_result THEN
            // записываем маркер
            written_size := SysFile.SysFileWrite(h_wr_file, ADR(marker_data),
                                                  SIZEOF(marker_data), ADR(iec_write_result));
            IF written_size = SIZEOF(marker_data) AND
               iec_write_result = CmpErrors.Errors.ERR_OK THEN
                // успешно, возвращаемся в начало
                iec_write_result := SysFile.SysFileSetPos(h_wr_file, 0);
            END_IF
        END_IF
    END_IF

    IF iec_write_result <> CmpErrors.Errors.ERR_OK THEN
        // ничего не вышло, закрываем
        SysFile.SysFileClose(h_wr_file);
        h_wr_file := CmpErrors.HandleConstants.RTS_INVALID_HANDLE;
    END_IF

    IF RTS_INVALID_HANDLE <> h_wr_file THEN
        // теперь записываем размер и данные

        // записываем размер данных
        data_size := SIZEOF(binary_data);
        written_size := SysFile.SysFileWrite(h_wr_file, ADR(data_size),
                                              SIZEOF(data_size), ADR(iec_write_result));

        IF written_size = SIZEOF(data_size) AND iec_write_result = CmpErrors.Errors.ERR_OK THEN
            // записываем данные
            written_size := SysFile.SysFileWrite(h_wr_file, ADR(binary_data),
                                                  data_size, ADR(iec_write_result));

            // ставим флаг
            bFileWritten := iec_write_result = CmpErrors.Errors.ERR_OK AND
                           written_size = data_size;
        END_IF

        // Принудительно переносим содержимое системного буфера файловой операции на диск.
        // В данном случае это делать необязательно, поскольку собираемся закрыть файл.
        // При закрытии файла происходит перенос содержимого буферов записи на диск.
        SysFile.SysFileFlush(h_wr_file);
        // Закрываем файл
        SysFile.SysFileClose(h_wr_file);
        // Делаем недействительным хэндл файла
        h_wr_file := CmpErrors.HandleConstants.RTS_INVALID_HANDLE;
    END_IF
END_IF
END_IF

```

Для определения факта наличия установленной карты microSD следует открыть соответствующий каталог и, в случае успеха, закрыть полученный системный идентификатор:

```

FUNCTION MicroSD_Exists : BOOL
VAR
  mmc_handle : RTS_IEC_HANDLE;
  mmc_result : RTS_IEC_RESULT;
END_VAR

MicroSD_Exists := FALSE;

mmc_handle := SysFile.SysFileOpen('microsd1', SysFile.AM_READ, ADR(mmc_result));

IF mmc_handle <> RTS_INVALID_HANDLE THEN
  MicroSD_Exists := TRUE;
  SysFile.SysFileClose(mmc_handle);
END_IF

```



Если во время извлечения съемного накопителя на него выполняется запись, это может привести к потере записываемых данных или к повреждению данных на карте. В связи с этим в приложении рекомендуется реализовать алгоритм взаимодействия с пользователем, гарантирующий отсутствие доступа к карте при извлечении. Данный алгоритм может, к примеру, при включенном переключателе 9 или 10 закрывать все файлы и каталоги на карте, после чего индикатором USER или USR1 извещать пользователя о возможности извлечения карты.

Для безопасного извлечения карты следует пользоваться функциями библиотеки FastwelRemovableMedia (см. п. 6.7)

Пример реализации алгоритма взаимодействия с пользователем при извлечении карты microSD:

```

FUNCTION_BLOCK MMCEjector

VAR_INPUT
  // Флаг от приложения, что доступа к карте больше нет
  bFilesReleased : BOOL;
  // Номер переключателя для запроса извлечения карты
  uiSwitchNumber : UINT := 10;
END_VAR

VAR_OUTPUT
  // Флаг наличия установленной карты
  bInserted : BOOL;
  // Флаг запроса извлечения карты
  bEjectRequest : BOOL;
  // Флаг возможности извлечения карты (необязательный)
  bEjectAllowed : BOOL;
END_VAR

VAR
  sw_reader : FastwelBoard.SWITCH_READER_x10;
  released_edge : Standard.R_TRIG;
  led_blinker : FastwelBoard.USER_LED_CONTROL;
  led_color : FastwelBoard.LedsColor_t := FastwelBoard.LED_COLOR_NONE;
  led_period : TIME := T#0MS;
END_VAR

VAR_TEMP
  sw_number : UINT;
  mmc_handle : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
  mmc_result : RTS_IEC_RESULT;
END_VAR

// читаем состояние переключателей
sw_reader(Period := T#100MS);
// выбираем номер переключателя для регистрации запроса извлечения карты
sw_number := SEL(uiSwitchNumber <> 10 AND uiSwitchNumber <> 9, uiSwitchNumber, 10);
// проверяем и устанавливаем признак запроса извлечения
bEjectRequest := sw_reader.SwitchStates[sw_number];

bInserted := FALSE;

```

```

bEjectAllowed := FALSE;
led_color := FastwelBoard.LED_COLOR_NONE;
led_period := T#0MS;

// проверяем, установлена ли карта
mmc_handle := SysFile.SysFileOpen('microsd1', SysFile.AM_READ, ADR(mmc_result));

IF mmc_handle <> RTS_INVALID_HANDLE THEN
  // да, установлена
  bInserted := TRUE;
  SysFile.SysFileClose(mmc_handle);
  mmc_handle := RTS_INVALID_HANDLE;
END_IF

IF bInserted AND bEjectRequest THEN
  // карта установлена и есть запрос на извлечение
  IF bFilesReleased THEN
    // Есть сигнал от приложения, что файлы закрыты.
    bEjectAllowed := TRUE;
    // Можно извлекать - включаем зеленый.
    led_color := FastwelBoard.LED_COLOR_GREEN;
    led_period := T#0MS;
  ELSE
    // Пока нельзя извлекать - мигаем красным
    led_color := FastwelBoard.LED_COLOR_RED;
    led_period := T#100MS;
  END_IF
END_IF

// вызываем блок управления индикатором USER
led_blinker(Color := led_color, BlinkPeriod := led_period);

```

Функциональный блок MMCEjector может использоваться следующим образом:

```

PROGRAM PLC_PRG
VAR
  // Эту переменную нужно установить в TRUE, как только закрыты все
  // файлы и каталоги на карте MicroSD.
  mmc_data_released : BOOL;

  // Экземпляр блока взаимодействия с картой, индикатором USER и переключателем
  ejector : MMCEjector;
END_VAR

// вызываем экземпляр блока
ejector( bFilesReleased:= mmc_data_released, uiSwitchNumber:= 10,
         bInserted=>, bEjectRequest=>, bEjectAllowed=>);

IF ejector.bInserted THEN
  // если карта установлена
  IF NOT ejector.bEjectRequest THEN
    // и нет запроса на извлечение карты, читаем/пишем данные на карту
    mmc_data_released := FALSE;
    // ...
  ELSE
    // и если пользователь решил извлечь карту, закрываем все файлы
    // и каталоги на карте.
    // ...
    // затем устанавливаем признак возможности извлечения
    mmc_data_released := TRUE;
  END_IF
END_IF

```

## 6.5.2. Рекомендации по использованию библиотеки SysFile

### 6.5.2.1. Общие сведения

Для работы с файлами предназначены следующие основные функции библиотеки:

1. SysFileGetSize – возвращает размер файла, путь к которому задан в качестве параметра.
2. SysFileOpen – создает новый или открывает существующий файл, путь к которому задан в качестве параметра, возвращает системный идентификатор (хэндл) для доступа к файлу.

3. SysFileSetPos – устанавливает позицию для чтения или записи данных в открытом файле.
4. SysFileGetPos – возвращает текущую позицию для чтения или записи в открытом файле.
5. SysFileRead – выполняет чтение в буфер приложения заданного количества байт из открытого файла, начиная с текущей позиции для чтения или записи.
6. SysFileWrite – выполняет запись данных в открытый файл из буфера приложения, начиная с текущей позиции для чтения или записи в файле.
7. SysFileFlush – переносит содержимое системного буфера (кэша), в котором накоплены данные последних операций записи.
8. SysFileClose – закрывает ранее открытый файл.

#### 6.5.2.2. Проверка наличия или отсутствия файла по заданному пути

Функция SysFileGetSize может использоваться для выяснения, присутствует ли некоторый файл по заданному пути. Проверка наличия ранее созданного файла обычно требуется для правильного выбора значения параметра `am` типа `ACCESS_MODE`, передаваемого функции `SysFileOpen` при открытии файла. Неправильный выбор режима открытия файла может привести к потере данных, ранее записанных в файл.

Для проверки наличия файла по заданному пути можно использовать следующие операции:

```
VAR
    // признак наличия файла
    bFileExists : BOOL;
    // путь к файлу
    szFilePath : STRING := 'my_data.bin';
    // размер файла
    xw_file_size : __XWORD;
    // результат операции
    get_size_res : RTS_IEC_RESULT;
END_VAR

// нужно обязательно проверить результат операции, т.к. может присутствовать файл
// нулевой длины.
xw_file_size := SysFile.SysFileGetSize(szFilePath, ADR(get_size_res));
// файл имеется 100%
bFileExists := CmpErrors.Errors.ERR_OK = get_size_res;
```

Для проверки отсутствия файла по заданному пути можно использовать следующий код:

```
VAR
    // признак отсутствия файла
    bFileNotExist : BOOL;
    // путь к файлу
    szFilePath : STRING := 'my_data.bin';
    // размер файла
    xw_file_size : __XWORD;
    // результат операции
    get_size_res : RTS_IEC_RESULT;
END_VAR

// нужно обязательно проверить результат операции, т.к. может присутствовать файл
// нулевой длины.
xw_file_size := SysFile.SysFileGetSize(szFilePath, ADR(get_size_res));
// файла нет 100%
bFileNotExist:= CmpErrors.Errors.ERR_NO_OBJECT = get_size_res;
```

#### 6.5.2.3. Режимы доступа к файлу

Функция открытия файла имеет следующий прототип:

```
FUNCTION SysFileOpen : RTS_IEC_HANDLE
VAR_INPUT
    // Путь к файлу
    szFile : REFERENCE TO STRING;
    // Режим доступа
    am : ACCESS_MODE;
    // Результат операции
    pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

Параметр `am`, имеющий тип `ACCESS_MODE`, определяет режим доступа к открываемому файлу. Тип `SysFile.ACCESS_MODE` перечисляет следующие режимы доступа к файлу:

#### **AM\_READ**

Если файл присутствует по заданному пути `szFile`, то он будет открыт только для чтения, при этом позиция чтения данных будет установлена в начало файла. Если файла нет по заданному пути, то `SysFileOpen` вернет `RTS_INVALID_HANDLE` и код ошибки `ERR_NO_OBJECT` в переменной, адрес которой передан функции в качестве третьего параметра.

#### **AM\_WRITE**

Если файл присутствует по заданному пути `szFile`, то он будет открыт только для записи, текущее содержимое файла будет утрачено, а позиция записи данных будет установлена в начало файла. Если файла нет по заданному пути, и все каталоги в пути были ранее успешно созданы, то `SysFileOpen` создаст файл с заданным именем, установит позицию записи данных в начало файла и вернет системный идентификатор (хэндл) созданного файла.

#### **AM\_APPEND**

Если файл присутствует по заданному пути `szFile`, то он будет открыт только для записи, при этом позиция записи данных будет установлена в конец файла. Если файла нет по заданному пути, то `SysFileOpen` вернет `RTS_INVALID_HANDLE` и код ошибки `ERR_NO_OBJECT` в переменной, адрес которой передан функции в качестве третьего параметра.

#### **AM\_READ\_PLUS**

Если файл присутствует по заданному пути `szFile`, то он будет открыт для чтения и записи, при этом позиция чтения и записи данных установлена в начало файла. Если файла нет по заданному пути, то `SysFileOpen` вернет `RTS_INVALID_HANDLE` и код ошибки `ERR_NO_OBJECT` в переменной, адрес которой передан функции в качестве третьего параметра.

#### **AM\_WRITE\_PLUS**

Если файл присутствует по заданному пути `szFile`, то он будет открыт для чтения и записи, текущее содержимое файла будет утрачено, а позиция чтения и записи данных будет установлена в начало файла. Если файла нет по заданному пути, и все каталоги в пути были ранее успешно созданы, то `SysFileOpen` создаст файл с заданным именем, установит позицию чтения и записи данных в начало файла и вернет системный идентификатор (хэндл) созданного файла.

#### **AM\_APPEND\_PLUS**

Если файл присутствует по заданному пути `szFile`, то он будет открыт для чтения и записи, при этом позиция чтения и записи данных установлена в конец файла. Если файла нет по заданному пути, и все каталоги в пути были ранее успешно созданы, то `SysFileOpen` создаст файл с заданным именем, установит позицию чтения и записи данных в начало файла и вернет системный идентификатор (хэндл) созданного файла.

Если приложению, загруженному в контроллер, требуется при первом запуске создать текстовый файл, а затем записывать в него строки друг за другом, то наиболее подходящим режимом открытия является `AM_APPEND_PLUS`:

```
VAR RETAIN
  // номер записи
  entry_num : DWORD := 0;
END_VAR

VAR
  // файл будет создан в каталоге пользователя user
  szFilePath : STRING := 'my_log.txt';
  // строка для записи в файл
  szLogEntry : STRING;
  // флаг добавления очередной записи
  bTryOpen : BOOL;
  open_file_res : RTS_IEC_RESULT;
  write_file_res : RTS_IEC_RESULT;
  h_existing_file : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
END_VAR
```

```

IF NOT bTryOpen THEN
  // открываем в режиме дописывания
  h_existing_file := SysFile.SysFileOpen(szFilePath, AM_APPEND_PLUS, ADR(open_file_res));

  IF CmpErrors.HandleConstants.RTS_INVALID_HANDLE <> h_existing_file AND
    CmpErrors.Errors.ERR_OK = open_file_res THEN
    // успех, формируем строку для записи в файл
    szLogEntry := Standard.CONCAT('Text entry ', DWORD_TO_STRING(entry_num));
    szLogEntry := Standard.CONCAT(szLogEntry, '$R$N');
    // записываем все символы строки, кроме закрывающего 0
    SysFile.SysFileWrite(h_existing_file,
                        ADR(szLogEntry),
                        Standard.LEN(szLogEntry), ADR(write_file_res));

    IF CmpErrors.Errors.ERR_OK = write_file_res THEN
      // записали успешно, увеличиваем на 1 номер записи
      entry_num := entry_num + 1;
    END_IF

    SysFileClose(h_existing_file);
    h_existing_file := CmpErrors.HandleConstants.RTS_INVALID_HANDLE;
  END_IF
  bTryOpen := TRUE;
END_IF

```

Как указывалось в п. 6.5.1, при записи данных в файл, сопровождающейся увеличением размера файла, в случае внезапного пропадания питания все предыдущие данные в файле могут оказаться утраченными, а размер файла станет равным 0. Более того, если питание пропало в момент обновления записи о файле в служебных областях файловой системы FAT или FAT32 на съемном дисковом накопителе, может произойти повреждение файловой системы с отказом последующего доступа к накопителю.

Для снижения вероятности таких отказов при реализации алгоритмов сохранения данных реального времени в файлах рекомендуется избегать использования режимов AM\_APPEND, AM\_APPEND\_PLUS, использовать файлы заранее заданного фиксированного размера (см. п. 6.5.1) и сохранять данные только в виде бинарных записей со служебными полями контроля целостности и определения местоположения.

#### 6.5.2.4. Чтение и запись данных

Чтение данных из открытого файла функцией SysFileRead выполняется с текущей позиции, которая может быть получена вызовом функции SysFileGetPos или установлена вызовом SysFileSetPos. Третий параметр функции SysFileRead (ulSize) должен иметь значение, меньшее либо равное размеру буфера, адрес которого передан функции в качестве второго параметра (pbyBuffer). Функция возвращает количество прочитанных байт, равное либо запрошенному количеству ulSize, либо разности значений позиции конца файла и текущей позиции чтения. После успешного чтения текущая позиция чтения (и записи) увеличивается на количество прочитанных байт.

Запись данных в открытый файл выполняется с текущей позиции, которая может быть получена вызовом функции SysFileGetPos или установлена вызовом SysFileSetPos. Обратите внимание, что после установки позиции вызовом SysFileSetPos, для реального увеличения размера файла нужно выполнить запись в установленную позицию. После успешной записи текущая позиция записи (и чтения) увеличивается на количество записанных байт.

Доступ к файлам на встроенном NAND флэш-накопителю контроллера CPM723-01 осуществляется синхронно без кэширования, т.е. записываемые данные немедленно передаются на флэш-диск. Скорость записи на NAND флэш-накопитель контроллера CPM723-01 составляет единицы Мбайт/с и зависит от размера записываемых данных.

Доступ к файлам на встроенном NAND флэш-накопителе контроллера CPM810-03 осуществляется полусинхронно с кэшированием чтения, т.е. записываемые данные немедленно передаются на флэш-диск. Скорость записи на NAND флэш-накопитель контроллера CPM810-03 составляет десятки Мбайт и зависит от размера передаваемых данных.





Абсолютное значение приоритета задачи, из которой выполняются файловые операции, не должно быть менее 15.

Доступ к карте microSD и USB-накопителям выполняется асинхронно с использованием системного кэширования. В связи с этим при необходимости держать некоторый файл открытым большую часть времени работы приложения, до закрытия файла требуется периодически вызывать функцию SysFileFlush. Скорость записи на карту microSD и USB-накопители составляет десятки Мбайт/с и в меньшей степени, чем для NAND флэш-накопителя, зависит от размера записываемых данных.

Дисковые накопители в составе контроллера снабжены системой выравнивания износа. В связи с этим при ожидаемой заполненности встроенного NAND флэш-накопителя на половину доступного размера и записи в файл с частотой 10 Гц время до износа накопителя составит около 3000000 часов. Для карты microSD, поставляемой в составе контроллера, этот показатель при тех же условиях примерно в 3 раза больше, чем у встроенного NAND флэш-накопителя.

#### 6.5.2.5. Закрытие файлов

Функция SysFileClose переносит данные, относящиеся к недавним операциям записи в файл из системного кэша на диск, освобождает системные ресурсы, связанные с ранее открытым файлом, и обновляет служебные записи о файле в используемой файловой системе. Данная функция должна быть вызвана после завершения всех операций с ранее открытым файлом, а также при завершении работы или при сбросе приложения.

Если системный идентификатор (хэндл) файла принадлежит функциональному блоку, то закрытие файла может быть выполнено в методе FB\_Exit. В программных единицах типа PROGRAM можно предусмотреть действие (action), вызываемое при обработке системного события *PrepareExit*, закрывающее все ранее открытые файлы. Информация об обработке системных событий приведена в п. 4.5 настоящего документа.

#### 6.5.3. Рекомендации по использованию библиотеки SysDir

Библиотека SysDir предназначена для работы с каталогами (папками) и содержит следующие основные функции:

1. SysDirCreate – создает каталог с заданным именем.
2. SysDirDelete – удаляет каталог с заданным именем.
3. SysDirRename – изменяет имя выбранного каталога.
4. SysDirOpen – открывает доступ к каталогу для последующего перечисления находящихся в нем файлов и подкаталогов.
5. SysDirRead – возвращает информацию о файле или подкаталоге в открытом каталоге.
6. SysDirClose – завершает доступ к каталогу и освобождает требуемые для доступа системные ресурсы.

Все функции библиотеки SysDir являются весьма тяжеловесными с точки зрения потребления системных ресурсов.

В библиотеке имеются функции получения и установки т.н. текущего каталога: SysDirGetCurrent и SysDirSetCurrent. Не рекомендуется использовать данные функции в приложениях, разрабатываемых для контроллеров любых производителей с системой исполнения CODESYS V3.

Для создания каталога используется функция SysDirCreate, которая в качестве параметра принимает ссылку на путь создаваемому каталогу, последний элемент которого является именем создаваемого каталога. Например, в вызове:

```
SysDir.SysDirCreate('microsd1/subdir1/subdir2/subdir3');
```

передаваемая функции строка содержит путь с последним элементом subdir3, который и является именем каталога, подлежащего созданию в каталоге subdir1/subdir2 на карте microSD, при этом все элементы пути, кроме последнего, должны быть именами уже существующих каталогов и подкаталогов.

Функция SysDirOpen позволяет убедиться, что каталог с заданным именем присутствует на диске, а также для последующего перечисления файлов и каталогов, принадлежащий выбранному каталогу, при помощи функции SysDirRead.

Для проверки наличия каталога на диске можно использовать следующий код:

```
VAR
  DirName : STRING := 'rootdata/data1';
  DirEntry : STRING(256);

  hDir : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
  bDirExists : BOOL;
  iec_res : RTS_IEC_RESULT;
END_VAR

// Проверяем доступность каталога.
// ВНИМАНИЕ! Это очень тяжеловесная операция!
hDir := SysDir.SysDirOpen(DirName, DirEntry, 0, 0, ADR(iec_res));
IF hDir <> RTS_INVALID_HANDLE THEN
  // Успех, каталог найден
  bDirExists := TRUE;
  SysDir.SysDirClose(hDir);
  hDir := RTS_INVALID_HANDLE;
END_IF
```

Функция SysDirRead может использоваться для перечисления файлов и каталогов, находящихся в открытом каталоге:

```
VAR
  // флаг запуска операции
  bDirOpen : BOOL;
  // Путь к каталогу пользователя
  szDirOpen : STRING := '';
  szDirContents : STRING(512);
  // Записи об именах файлов и подкаталогов
  szDirRead : ARRAY [1..10] OF STRING(32);
  // Служебная информация о файлах и подкаталогах
  dirInfoRead : ARRAY [1..10] OF SysDir.DirInfo;
  // индекс
  dir_idx : UINT;

  sDirInfo : SysDir.DirInfo;
  dir_open_res : RTS_IEC_RESULT;
  h_dir : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
END_VAR

IF NOT bDirOpen THEN
  // открываем каталог
  h_dir := SysDir.SysDirOpen(  szDirOpen, szDirContents,
                              SIZEOF(szDirContents), ADR(sDirInfo), ADR(dir_open_res));

  IF CmpErrors.HandleErrors.RTS_INVALID_HANDLE <> h_dir THEN
    // успех
    bDirOpen := dir_open_res = CmpErrors.Errors.ERR_OK;

    IF bDirOpen THEN
      // читаем до конца массива или до завершения перечисления
      FOR dir_idx := 1 TO 10 DO
        IF CmpErrors.Errors.ERR_OK <> SysDir.SysDirRead(h_dir,
                                                         szDirRead[dir_idx],
                                                         SIZEOF(szDirRead[dir_idx]),
                                                         ADR(dirInfoRead[dir_idx])) THEN
          // выходим, файлы и подкаталоги закончились
          EXIT;
        END_IF
      END_FOR
    END_IF
    // закрываем каталог
    SysDir.SysDirClose(h_dir);
    h_dir := CmpErrors.HandleErrors.RTS_INVALID_HANDLE;
  END_IF
END_IF
```

Функция SysDirClose должна быть вызвана после завершения всех операций с ранее открытым каталогом, а также при завершении работы или при сбросе приложения.

Если системный идентификатор (хэндл) каталога принадлежит функциональному блоку, то закрытие каталога может быть выполнено в методе FB\_Exit. В программных единицах типа PROGRAM можно предусмотреть действие (action), вызываемое при обработке системного события *PrepareExit* и закрывающее все ранее открытые каталоги. Информация об обработке системных событий приведена в п. 4.5 настоящего документа.

## 6.6. Библиотека SysCom

### 6.6.1. Общие сведения

Библиотека SysCom предназначена для реализации взаимодействия приложения, загруженного в контроллер, с внешними устройствами по последовательным каналам связи RS-485 или RS-232C через встроенные последовательные порты контроллера и/или модули NIM741 или NIM742, подключенные к локальной межмодульной шине контроллера.

Библиотека SysCom содержит следующие основные функции:

1. SysComOpen – открывает последовательный порт с заданным числовым идентификатором (номером) и возвращает системный идентификатор (хэндл) порта для последующего использования в приложении.
2. SysComOpen2 – открывает и настраивает параметры заданного последовательного порта, а затем возвращает хэндл открытого порта.
3. SysComSetSettings – выполняет настройку параметров ранее открытого последовательного порта.
4. SysComRead – читает данные из открытого порта с заданным системным идентификатором.
5. SysComWrite – записывает данные в открытый порт с заданным системным идентификатором.
6. SysComClose – закрывает ранее открытый последовательный порт по системному идентификатору, переданному в качестве параметру.

Остальные функции библиотеки в подавляющем большинстве случаев не нужны при разработке приложений.

Проекты ModbusRtuClient.project и SerialConsole.project, входящие в пакет адаптации CODESYS V3 для контроллеров Fastwel, иллюстрируют основные приемы работы с функциями библиотеки.

### 6.6.2. Начало и завершение работы с последовательным портом

Перед началом обмена данными через последовательный порт необходимо получить его системный идентификатор и настроить параметры обмена: скорость обмена, количество бит данных и стоп-бит и режим контроля четности.

Для получения системного идентификатора могут использоваться функции SysComOpen или SysComOpen2. Функция SysComOpen2, в отличие от SysComOpen, позволяет открыть последовательный порт и настроить параметры обмена за один вызов.


Обе функции могут быть вызваны из задач приложения или в обработчиках системных событий, предшествующих запуску приложения.

Функция SysComOpen имеет следующий прототип:

```
FUNCTION SysComOpen : RTS_IEC_HANDLE
VAR_INPUT
    // Номер порта
    sPort : COM_Ports;
    // Результат операции
    pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```

Первый параметр типа COM\_Ports предназначен для указания номера порта, который требуется открыть.

Для портов на базе модулей NIM741 или NIM742, подключенных к локальной шине контроллера, sPort должен иметь значение  $(100 + n)$ , где  $n$  принимает значение от 1 до 64 включительно и представляет номер модуля на шине среди всех активных модулей, присутствующих в конфигурации загруженного приложения.

	<p>В соответствующую позицию списка модулей в конфигурации локальной межмодульной шины приложения должен быть добавлен элемент <i>NIM741 RS-485 1xUART Stream Module</i> или <i>NIM742 RS-232C 1xUART Stream Module</i>. Более подробная информация об идентификации последовательных портов на базе модулей NIM741/NIM841 или NIM742/NIM842 приведена в документе ИМЕС.00300-03 33 01 (см. таблицу 1).</p>
---	---

Второй параметр служит для передачи адреса переменной, в которую будет помещен результат вызова функции. При попытке повторно открыть ранее открытый порт результат вызова будет равен ERR\_FAILED, и функция вернет RTS\_INVALID\_HANDLE.

При успешном открытии порта функция вернет системный идентификатор порта, а переменная типа RTS\_IEC\_RESULT, адрес которой передан в качестве второго параметра, будет содержать ERR\_OK.

После успешного вызова SysComOpen для настройки параметров обмена следует вызвать функцию SysComSetSettings, передав в качестве первого параметра системный идентификатор, полученный вызовом SysComOpen, а в качестве второго – адрес переменной типа COM\_Settings, содержащей требуемые значения параметров обмена. Третий параметр функции SysComSetSettings типа COM\_SettingsEx должен быть равен 0.

Например, пусть в программе COM\_READER объявлены следующие переменные для работы портом на базе NIM741, установленным в третью позицию на локальной шине контроллера:

```
PROGRAM COM_READER
VAR
  (* Параметры порта *)
  comSettings : SysCom.COM_Settings :=( sPort := 103,
                                         byStopBits :=SysCom.COM_StopBits.SYS_ONESTOPBIT,
                                         byParity :=SysCom.COM_Parity.SYS_NOPARITY,
                                         ulBaudrate := SysCom.COM_Baudrate.SYS_BR_115200,
                                         ulTimeout := 0,
                                         ulBufferSize := 512);

  (* Если равна TRUE, то порт открыт и настроен согласно comSettings *)
  portInitialized : BOOL;
  (* Описатель порта, значение по умолчанию – не определено *)
  portHandle : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
END_VAR
```

Тогда действие (action) InitCOM по открытию и настройке порта может иметь следующий вид:

```
// InitCOM action
IF portHandle = RTS_INVALID_HANDLE THEN
  // если ранее не был открыт, пробуем открыть
  portHandle := SysCom.SysComOpen( comSettings.sPort, ADR(iec_res));
END_IF

IF portHandle <> RTS_INVALID_HANDLE THEN
  // системный идентификатор получен
  IF iec_res = CmpErrors.Errors.ERR_OK THEN
    // если еще не было настройки, пробуем выполнить настройку
    IF NOT portInitialized THEN
      iec_res := SysCom.SysComSetSettings( portHandle, ADR(comSettings), 0);
      portInitialized := (iec_res = CmpErrors.Errors.ERR_OK);
      IF NOT portInitialized THEN
        // вызов действия (action), закрывающего порт
        CloseCOM();
      END_IF
    END_IF
  ELSE
    // вызов действия (action), закрывающего порт
    CloseCOM();
  END_IF
END_IF
```

Тогда в теле основной программы можно повторно вызывать действие InitCOM, не заботясь о текущем состоянии приложения:

```
IF portInitialized THEN
  // Выполняем обмен данными. При необходимости изменить параметры обмена меняем
  // соответствующие поля в структуре comSettings и записываем FALSE в portInitialized.
ELSE
  //
  InitCOM();
END_IF
```

В данном примере используется действие CloseCOM, которое служит для завершения работы с портом и освобождения связанных с ним системных ресурсов:

```
// CloseCOM action
IF portHandle <> RTS_INVALID_HANDLE THEN
  SysCom.SysComClose(portHandle);
  portHandle := RTS_INVALID_HANDLE;
  portInitialized := FALSE;
END_IF
```

Данное действие необходимо вызывать в обработчике системного события *PrepareExit* (см. п. 4.5) для корректного освобождения системных ресурсов, связанных с ранее открытым портом, в случае модификации приложения методом полной загрузки (см. п. 4.6.2), при перезапуске текущего приложения и в случае загрузки другого приложения.

Если для обмена данными через последовательный порт предполагается разработка соответствующего функционального блока, то получение доступа к порту и завершение работы с портом могут быть выполнены в методах функционального блока FB\_Init и FB\_Exit соответственно. В таком случае для корректного освобождения системных ресурсов не потребуется устанавливать обработчики системных событий.

При настройке параметров порта функцией SysComSetSettings используется структура COM\_Settings:

```
TYPE COM_Settings :
STRUCT
  // Номер порта: 101, 102, ..., 164
  sPort      : COM_Ports;
  // Кол-во стоповых бит: SYS_ONESTOPBIT, SYS_ONESTOPBITS, SYS_TWOSTOPBITS
  byStopBits : COM_StopBits;
  // Режим контроля четности: SYS_NOPARITY, SYS_ODDPARITY, SYS_EVENPARITY
  byParity   : COM_Parity;
  // Скорость обмена: SYS_BR_9600, SYS_BR_19200, SYS_BR_38400, SYS_BR_57600, SYS_BR_115200
  ulBaudrate : COM_Baudrate;
  // Не используется с портами на базе NIM741/NIM742!
  ulTimeout  : COM_Timeout;
  // Размер системного буфера
  ulBufferSize : UDINT;
END_STRUCT
END_TYPE
```

Параметр ulTimeout не используется при настройке портов на базе модулей NIM741 и NIM742, и должен быть равен 0.

При настройке встроенных последовательных портов контроллера параметр ulTimeout данной структуры определяет длительность (в миллисекундах) межсимвольного таймаута чтения данных порта следующим образом:

- Если в буфере приема встроенного последовательного порта перед вызовом SysComRead нет символов, то выход из функции SysComRead произойдет по истечении таймаута, переданного ей в качестве четвертого параметра (ulTimeout).
- Если в буфере приема встроенного последовательного порта перед вызовом SysComRead есть хотя бы один символ или с момента входа в SysComRead в течение таймаута, установленного параметром ulTimeout в структуре COM\_Settings, в буфере приема появился хотя бы один символ, то далее выход из функции SysComRead произойдет не ранее завершения еще одного межсимвольного таймаута.

Параметр ulTimeout не влияет на работу функции SysComWrite.

Параметр `ulBufferSize` позволяет определить размеры внутренних системных буферов приема и передачи, ассоциированных с данным портом. Если этот параметр сделан равным 0, то по умолчанию система зарезервирует максимально возможные 1024 байта для буфера приема и для буфера передачи. Значение данного параметра не должно быть менее максимального размера данных, передаваемых через последовательный порт.

Настройка параметров порта может быть выполнена одновременно с получением системного идентификатора при помощи функции `SysComOpen2`:

```
FUNCTION SysComOpen2 : RTS_IEC_HANDLE
VAR_INPUT
    // Номер и параметры порта
    pSettings : POINTER TO COM_Settings;
    // Не используется с портами на базе NIM741/NIM742, должен быть равен 0
    pSettingsEx : POINTER TO COM_SettingsEx;
    // Результат операции
    pResult : POINTER TO RTS_IEC_RESULT;
END_VAR
```



Параметр `pSettingsEx` не используется при работе с портами на базе модулей NIM741/NIM841 и NIM742/NIM842, и должен быть равен 0.

Функция `SysComOpen2` позволяет значительно сократить количество кода, если инициализация и настройка параметров порта в приложении должны выполняться однократно:

```
portHandle := SysCom.SysComOpen2(ADR(comSettings), 0, ADR(iec_res));
```

### 6.6.3. Обмен данными через последовательный порт

Функции `SysComRead` и `SysComWrite` предназначены для чтения (приема) и записи (передачи) данных через последовательный порт.

Основной особенностью указанных функций в системе исполнения контроллеров Fastwel при работе с последовательными портами, реализованными на базе модулей NIM741/NIM742/NIM841/NIM842, является полная асинхронность функций `SysComWrite` и `SysComRead` относительно задачи (потока исполнения), в контексте которой вызывается каждая из этих функций.

При работе со встроенными последовательными портами функция `SysComRead` выполняется асинхронно, если параметр `ulTimeout` равен 0, и синхронно – в противном случае. Функция `SysComWrite` при работе со встроенными портами всегда асинхронна.

Полная асинхронность означает, что при вызове функция взаимодействует только с внутренним системным буфером чтения (`SysComRead`) или записи (`SysComWrite`) соответствующего последовательного порта и никогда не останавливается в ожидании завершения приема (`SysComRead`) или передачи (`SysComWrite`) данных по каналу связи.

Асинхронность функций чтения и записи обуславливает следующие особенности их поведения:

1. При отсутствии в приложении более приоритетных задач, чем задача, из которой вызывается `SysComRead/SysComWrite`, время выполнения данных функций составляет от десятков до сотен микросекунд. Таким образом, функции чтения и записи в последовательный порт могут вызываться из задач любого приоритета.
2. Четвертый параметр `ulTimeout` для `SysComWrite` не имеет значения, и всегда может быть равен 0. Для `SysComRead` данный параметр имеет значение только при работе со встроенными последовательными портами контроллера.
3. Функция `SysComRead` завершается без какого-либо ожидания, скопировав в буфер приложения (пользовательский буфер), адрес которого передан в качестве второго параметра, ровно столько байт, сколько было принято сервисом последовательного порта во внутренний системный буфер (буфер приема) перед вызовом `SysComRead`. Функция возвращает количество байт, скопированных в пользовательский буфер из буфера приема.



При работе со встроенным последовательным портом функция SysComRead приостанавливается на время, определенное параметром ulTimeout (в мс), с момента вызова до приема очередного символа.

4. Функция SysComWrite завершается без какого-либо ожидания, скопировав данные из пользовательского буфера, адрес которого передан в качестве второго параметра, во внутренний системный буфер передачи. При успешном завершении функция возвращает количество байт, равное значению третьего параметра (ulSize).
5. Максимальный размер одновременно передаваемых или принимаемых данных не может быть более 1024 байт.

Для передачи данных через последовательный порт с некоторым системным идентификатором должна использоваться функция SysComWrite:

```
FUNCTION SysComWrite : UDINT
VAR_INPUT
  // Системный идентификатор порта
  hCom      : RTS_IEC_HANDLE;
  // Пользовательский буфер, содержащий данные для передачи
  pbyBuffer  : POINTER TO BYTE;
  // Количество байт данных в буфере pbyBuffer, подлежащее передаче
  ulSize     : UDINT;
  // Не используется с портами на базе NIM741/NIM742!
  ulTimeout  : COM_Timeout;
  // Результат операции
  pResult    : POINTER TO RTS_IEC_RESULT;
END_VAR
```

Функция возвращает количество байт, записанных во внутренний системный буфер передачи из пользовательского буфера.

Функция SysComWrite должна вызываться не чаще, чем требуется для передачи заданного количества байт по каналу связи при выбранной скорости обмена. Например, для передачи 256 байт с параметрами обмена 115200-8-N-1 требуется около 22,2 мс, т.е. минимальный период вызова SysComWrite в таком случае (с небольшим запасом) составляет 25 мс.

Пример передачи данных:

```
PROGRAM COM_WRITER

VAR CONSTANT
  (* Размер передаваемого пакета *)
  FRAME_SIZE : UDINT := 256;
END_VAR

VAR
  (* Параметры порта *)
  comSettings : SysCom.COM_Settings :=( sPort := 102,
                                         byStopBits :=SysCom.COM_StopBits.SYS_ONESTOPBIT,
                                         byParity :=SysCom.COM_Parity.SYS_NOPARITY,
                                         ulBaudrate := SysCom.COM_Baudrate.SYS_BR_115200,
                                         ulTimeout := 0,
                                         ulBufferSize := FRAME_SIZE);

  (* Пользовательский буфер передачи *)
  data_buffer : ARRAY [1..FRAME_SIZE] OF BYTE;
  (* Если равна TRUE, то порт открыт и настроен согласно comSettings *)
  portInitialized : BOOL;
  (* Описатель порта, значение по умолчанию – не определено *)
  portHandle : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;
  (* Результат вызова *)
  iec_res : RTS_IEC_RESULT;
  (* Период вызова, может быть константой *)
  send_period : TIME := T#50MS;
  (* Момент времени для передачи *)
  time_to_send : TIME;
  (* Текущий момент времени в мс *)
  current_time : TIME;
END_VAR

VAR_TEMP
  idx : UDINT;
  wr_size : UDINT;
```



```

END_VAR

current_time := TIME();

IF NOT portInitialized THEN
  InitCOM();
END_IF

IF portInitialized THEN
  IF TIME_TO_DINT(current_time - time_to_send) >= 0 THEN
    // В первый байт пакета записываем его размер минус размер поля длины
    data_buffer[1] := UINT_TO_BYTE(SIZEOF(data_buffer) - SIZEOF(data_buffer[1]));
    // Записываем данные для передачи
    FOR idx := 2 TO FRAME_SIZE DO
      data_buffer[idx] := UDINT_TO_BYTE(idx);
    END_FOR
    // Записываем во внутренний системный буфер передачи
    wr_size := SysCom.SysComWrite(
      portHandle,
      ADR(data_buffer[1]),
      SIZEOF(data_buffer),
      0,
      ADR(iec_res));

    // Определяем момент времени для передачи следующего пакета
    time_to_send := current_time + send_period;
  END_IF
END_IF

```

В данном примере используется действие InitCOM, рассмотренное ранее в п. 6.6.2.

Для приема данных через последовательный порт должна использоваться функция SysComRead:

```

FUNCTION SysComRead : UDINT
VAR_INPUT
  // Системный идентификатор порта
  hCom      : RTS_IEC_HANDLE;
  // Пользовательский буфер, в который будут помещены принятые данные
  pbyBuffer : POINTER TO BYTE;
  // Требуемое количество считываемых байт или размер пользовательского буфера приема
  ulSize    : UDINT;
  // Не используется с портами на базе NIM741/NIM742!
  ulTimeout : COM_Timeout;
  // Результат операции
  pResult   : POINTER TO RTS_IEC_RESULT;
END_VAR

```

Функция возвращает количество байт, скопированных из внутреннего системного буфера приема в пользовательский буфер pbyBuffer.

Алгоритм селекции пакетов, поступающих от удаленного передатчика данных, зависит от протокола обмена. Например, при использовании потоковых протоколов, подобных MODBUS RTU, перед обработкой принятых данных после чтения из порта очередной порции данных следует "засекать" интервальный таймаут, длительность которого составляет не менее длительности передачи по каналу связи 3...3,5 символов на текущей установленной скорости обмена. Практически при скоростях обмена более 19200 бит/с в качестве значения интервального таймаута можно использовать период цикла задачи, в контексте которой вызывается SysComRead.

Пример приема пакетов для приведенного выше примера передачи:

```

PROGRAM COM_READER
VAR
  // Параметры порта
  comSettings : SysCom.COM_Settings :=( sPort := 103,
    byStopBits :=SysCom.COM_StopBits.SYS_ONESTOPBIT,
    byParity :=SysCom.COM_Parity.SYS_NOPARITY,
    ulBaudrate := SysCom.COM_Baudrate.SYS_BR_115200,
    ulTimeout := 0,
    ulBufferSize := 512);

  // Если равна TRUE, то порт открыт и настроен согласно comSettings
  portInitialized : BOOL;

  // Описатель порта, значение по умолчанию – не определено
  portHandle : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;

```

```

// Пользовательский буфер приема
data_buffer : ARRAY [1..512] OF BYTE;
// Указатель на место в пользовательском буфере приема
read_ptr : POINTER TO BYTE := 0;
// Общее количество принятых байт в очередном пакете
bytes_read : UDINT;
// Доступный размер в пользовательском буфере приема
avail_size : UDINT;
// Момент наступления интервального таймаута
wait_end : TIME := T#0MS;
// Значение интервального таймаута
end_tout : TIME := T#10MS;
// Количество принятых фрагментов, похожих на пакеты
chunks_received : DWORD;
// Количество принятых пакетов
frames_received : DWORD;
// Состояние приемника: 0 - ждем новый пакет, 1 - ждем окончания очередного пакета
state : INT := 0;
// Кол-во байт, прочитанных из системного буфера приема
rd_size : UDINT;
// Результат операции
iec_res : RTS_IEC_RESULT;
// Текущий момент времени
current_time : TIME;
END_VAR

VAR_TEMP
  idx : UDINT;
END_VAR

current_time := TIME();

IF NOT portInitialized THEN
  InitCOM();
END_IF

IF portInitialized THEN
  IF avail_size = 0 THEN
    // пользовательский буфер заполнен до конца, будем ждать новый пакет
    state := 0;
  END_IF
  IF read_ptr = 0 OR state = 0 THEN
    // ожидаем очередной пакет
    read_ptr := ADR(data_buffer[1]);
    bytes_read := 0;
    avail_size := SIZEOF(data_buffer);
    wait_end := T#0MS;
  END_IF
  // читаем данные
  rd_size := SysCom.SysComRead( portHandle, read_ptr, avail_size, 0, ADR(iec_res));
  IF rd_size <> 0 THEN
    // Удалось что-то прочитать, увеличиваем общее кол-во принятых байт
    bytes_read := bytes_read + rd_size;
    // Уменьшаем доступный размер в пользовательском буфере приема
    avail_size := avail_size - rd_size;
    // Сдвигаем указатель в пользовательском буфере приема
    read_ptr := read_ptr + rd_size;
    IF state = 0 THEN
      // Ждали начало очередного пакета, теперь ждем его конец
      state := 1;
    END_IF
  ELSE
    // прочитали 0 байт
    IF bytes_read <> 0 THEN
      // ранее что-то удалось принять
      IF wait_end = T#0MS THEN
        // засекаем интервальный таймаут
        wait_end := current_time + end_tout;
      ELSIF TIME_TO_DINT(current_time - end_tout) >= 0 THEN
        // таймаут наступил
        chunks_received := chunks_received + 1;
        // достаем и проверяем поле длины из первого байта

```

```

IF bytes_read = (*FRAME_SIZE*) (data_buffer[1] + 1) THEN
  // оно совпадает с количеством принятых байт, значит успех
  frames_received := frames_received + 1;
END_IF
// возвращаемся в состояние ожидания очередного пакета
state := 0;
END_IF
END_IF
END_IF
END_IF

```

В данном примере используется действие InitCOM, рассмотренное ранее в п. 6.6.2.



При большом количестве модулей ввода-вывода на локальной шине и высокой частоте опроса модулей циклограмма обмена данными через последовательные порты на базе NIM741/NIM841 и NIM742/NIM842 может иметь аperiodический характер.

Кроме того, при переходе межмодульной шины FBUS в частично исправное состояние возможны перемежающиеся сбои обмена данными по последовательному каналу через исправные модули NIM741/NIM841 и NIM742/NIM842, связанные с занятостью межмодульной шины FBUS во время выполнения процедуры обнаружения и повторной инициализации модулей, с которыми утрачена связь.

## 6.7. Библиотека FastwelRemovableMedia

### 6.7.1. Общие сведения

Библиотека FastwelRemovableMedia предназначена для получения информации о съемных дисковых накопителях, для контроля состояния съемных дисковых накопителей, а также для программного отключения накопителя в приложении IDE МЭК 61131-3 контроллера перед физическим извлечением из соответствующего гнезда.

Библиотека содержит следующие функции:

1. SysBoardDeviceEject – программное отключение накопителя по имени системного дискового устройства.
2. SysBoardDeviceGetAllDisks – получение информации о дисковых устройствах.
3. SysBoardDeviceGetAllMountPoints – получение информации о точках монтирования разделов дисковых устройств.
4. SysBoardDeviceGetDiskInfo – получение информации о дисковом устройстве, которому принадлежит точка монтирования раздела.

В библиотеке определено событие EVT\_BoardDeviceNotify, позволяющее приложению получать асинхронные уведомления о подключении или отключении съемных накопителей.

### 6.7.2. Типы данных

#### 6.7.2.1. MountDisk

Данный тип содержит описание дискового накопителя, а также перечень всех точек монтирования разделов, имеющихся на данном накопителе.

```

TYPE MountDisk :
STRUCT
  (* описание дискового накопителя *)
  Info : MountDiskInfo;
  (* точки монтирования, связанные с данным накопителем *)
  pMountPoints : POINTER TO MountPoint;
  (* количество точек монтирования, связанные с данным накопителем *)
  iMountPointsCount : DINT;
END_STRUCT
END_TYPE

```

#### 6.7.2.2. MountDiskInfo

Данный тип содержит информацию о дисковом накопителе, в том числе имя и основные свойства системного устройства.

```

TYPE MountDiskInfo :
STRUCT
  (* имя системного устройства, соответствующего диску *)
  szDiskDevice : STRING(DEVICE_NAME_MAX);
  (* флаги свойств дискового устройства *)
  ulFlags : DWORD;
END_STRUCT
END_TYPE

```

Поле szDiskDevice содержит полное имя системного дискового устройства, например */dev/mmcblk*.

Значение DEVICE\_NAME\_MAX определено в декларации глобальных констант в библиотеке, и имеет значение 63.

Поле ulFlags содержит битовую маску флагов свойств системного дискового устройства, которые определены в виде констант:

```

VAR_GLOBAL CONSTANT
  (* накопитель является блочным устройством *)
  DISK_FLAG_BLOCK : DWORD := 1;
  (* накопитель может быть безопасно отключен функцией SysBoardDeviceEject *)
  DISK_FLAG_REMOVABLE : DWORD := 2;
END_VAR

```

Наличие установленного бита DISK\_FLAG\_REMOVABLE в поле ulFlags свидетельствует о возможности отключить накопитель функцией SysBoardDeviceEject.

### 6.7.2.3. MountPoint

Данный тип описывает точку монтирования раздела съемного дискового накопителя.

Точкой монтирования является каталог szPath, через который осуществляется доступ к разделу szDevice некоторого дискового накопителя.

Для карты microSD, устанавливаемой в соответствующее гнездо контроллера, система выполнит монтирование разделов карты в каталог *./user/microsd<n>*, где n = 1, 2 и т.д. – номера разделов карты microSD.

Для USB-накопителя с одним разделом, подключенным к гнезду USB1 система выполнит монтирование раздела в каталог *./user/usb1p1*

Значение DEVICE\_NAME\_MAX определено в декларации глобальных констант библиотеки, и имеет значение 63.

Значение MOUNTPOINT\_PATH\_MAX определено в декларации глобальных констант библиотеки, и имеет значение 254.

```

TYPE MountPoint :
STRUCT
  (* системное имя устройства точки монтирования, соответствующего разделу накопителя *)
  szDevice : STRING(DEVICE_NAME_MAX);
  (* путь к каталогу, в который смонтирован раздел накопителя *)
  szPath : STRING(MOUNTPOINT_PATH_MAX);
END_STRUCT
END_TYPE

```

### 6.7.2.4. EVTPARAM\_BoardDeviceNotify

Данный тип содержит информацию о системном событии с идентификатором EVT\_BoardDeviceNotify (значение 16#10001), которое формируется компонентом CMPID\_SysBoardDevice (значение 16#2015) при подключении или отключении съемного дискового накопителя.

```

TYPE EVTPARAM_BoardDeviceNotify :
STRUCT
  (* имя устройства, соответствующего разделу съемного накопителя *)
  pszDeviceName : POINTER TO STRING;
  (* каталог, в который смонтирован раздел съемного накопителя *)
  pszPath : POINTER TO STRING;
  (* доступность: TRUE – есть доступ к каталогу монтирования раздела *)

```

```

bAccessible : BOOL;
END_STRUCT
END_TYPE

```

При подключении съемного дискового накопителя и завершении монтирования имеющегося на нем раздела компонент CMPID\_SysBoardDevice генерирует системное событие EVT\_BoardDeviceNotify, что приводит к вызову всех зарегистрированных функций-обработчиков с параметром типа POINTER TO EventParam, у которого в поле pParameter передается указатель на переменную типа EVTPARAM\_BoardDeviceNotify, содержащую следующие значения:

- поле pszDevice – имя системного устройства, соответствующего разделу подключенного накопителя, например: `/dev/mmcblk0p1` или `/dev/sdb5`;
- поле pszPath – путь к каталогу, в который смонтирован раздел подключенного накопителя, например: `./user/microsd1` или `./user/usb3p5`;
- поле bAccessible – значение TRUE, указывающее на возможность доступа к файлам и каталогам на разделе pszDevice подключенного диска с использованием пути, начинающегося с каталога pszPath, например: `./user/microsd1/my_folder/my_file.txt` или `./user/usb3p5/my_folder/my_file.txt`.

При отключении карты MicroSD командой *eject* оболочки ПЛИК или вызовом функции SysBoardDeviceEject компонент CMPID\_SysBoardDevice генерирует системное событие EVT\_BoardDeviceNotify, что приводит к вызову всех зарегистрированных функций-обработчиков с параметром типа POINTER TO EventParam, у которого в поле pParameter передается указатель на переменную типа EVTPARAM\_BoardDeviceNotify, содержащую следующие значения:

- поле pszDevice – имя системного устройства, соответствующего отключенному разделу накопителя, например: `/dev/mmcblk0p1` или `/dev/sdb5`;
- поле pszPath – путь к каталогу, в который ранее был смонтирован отключенный раздел накопителя, например: `./user/microsd1` или `./user/usb3p5`;
- поле bAccessible – значение FALSE, указывающее на невозможность иметь доступ к файлам и каталогам на отключенном разделе pszDevice через каталог pszPath.



Все файлы и каталоги, открытые на всех подключенных разделах съемных накопителей, должны быть закрыты вызовом SysFileClose до выполнения команды *eject* и до вызова SysBoardDeviceEject.

В качестве обработчика события EVTPARAM\_BoardDeviceNotify должна использоваться функция со следующим прототипом (имя может быть другим):

```

FUNCTION DeviceNotifyCallback : RTS_IEC_RESULT
VAR_INPUT
    pEvent : POINTER TO EventParam;
END_VAR

```

В декларации функции можно объявить переменную для присвоения переменной типа EVTPARAM\_BoardDeviceNotify:

```

VAR
    pParameter : POINTER TO EVTPARAM_BoardDeviceNotify;
END_VAR

```

Данной переменной следует присвоить значение в поле pParameter в параметре функции pEvent:

```

DeviceNotifyCallback := ERR_PARAMETER;
IF pEvent <> 0 THEN
    // получаем параметр
    pParameter := pEvent^.pParameter;
    // анализируем параметр
    ...
    DeviceNotifyCallback := ERR_OK;
END_IF

```

Для регистрации функции обработки события следует подключить к проекту библиотеку CmpEventManager и выполнить регистрацию обработчика в коде приложения следующим образом:

```

VAR
    // системный идентификатор события
    hDeviceNotify : RTS_IEC_HANDLE := RTS_INVALID_HANDLE;

```

```

Result : RTS_IEC_RESULT;
END_VAR

IF hDeviceNotify = RTS_INVALID_HANDLE THEN
  // открываем событие, получая корректный системный идентификатор
  hDeviceNotify := EventOpen(EVT_BoardDeviceNotify, CMPID_SysBoardDevice, Result);
  IF hDeviceNotify <> RTS_INVALID_HANDLE THEN
    // регистрируем функцию обработки события DeviceNotifyCallback
    Result := EventRegisterCallbackFunction (hDeviceNotify, ADR(DeviceNotifyCallback));
  END_IF
END_IF

```

Если функция обработки события зарегистрирована успешно, то впоследствии следует выполнить обратную операцию – "разрегистрировать" функцию обработки и закрыть системный идентификатор события по завершении работы приложения:

```

IF hDeviceNotify <> RTS_INVALID_HANDLE THEN
  EventUnregisterCallbackFunction(hDeviceNotify, ADR(DeviceNotifyCallback));
  EventClose(hDeviceNotify, Result);
  hDeviceNotify := RTS_INVALID_HANDLE;
END_IF

```



Завершение работы с обработчиком события указанным методом должно быть обязательно сделано в обработчике системного события *PrepareExit* или в методе *FB\_Exit* функционального блока, которому принадлежит системный идентификатор события. Если этого не сделать, то приложение, многократно загружаемое в контроллер в процессе отладки или модернизации, станет источником утечки системных ресурсов.

### 6.7.3. Функции

#### 6.7.3.1. SysBoardDeviceGetAllDisks

Функция предназначена для получения информации обо всех подключенных дисковых накопителях.

##### Прототип

```

FUNCTION SysBoardDeviceGetAllDisks : RTS_IEC_RESULT
VAR_INPUT
  pDisks : POINTER TO MountDisk;
  pdiDisksCount : POINTER TO DINT;
  pMountPoints : POINTER TO MountPoint;
  diMountPointsCount : DINT;
END_VAR

```

##### Входные параметры

**pDisks : POINTER TO MountDisk**

Указатель на массив типа MountDisk (см. п. 6.7.2.1), в который будут помещены описания дисковых накопителей при успешном вызове.

Если данный параметр равен 0, а второй параметр не равен 0, то при успешном вызове в переменную по адресу pdiDisksCount будет помещено количество обнаруженных дисковых накопителей.

**pdiDisksCount : POINTER TO DINT**

Указатель на переменную типа DINT, в которую перед вызовом следует поместить количество элементов в массиве pDisks. После успешного вызова функции в данную переменную будет помещено количество описаний обнаруженных дисковых накопителей.

**pMountPoints : POINTER TO MountPoint**

Указатель на массив типа MountPoint (см. п. 6.7.2.3), в который будут помещены описания всех точек монтирования разделов съемных дисковых накопителей при успешном вызове.

**diMountPointsCount : DINT**

Переменная типа DINT, в которую до вызова следует поместить количество элементов в массиве pMountPoints.

Возвращаемый результат**ERR\_OK**

Успех.

**ERR\_PARAMETER**

С учетом оговорки по сочетанию нулевых значений первых двух параметров, данный результат возвращается при равенстве 0 любого из указателей, переданных функции, либо при равенстве нулю значения по указателю pDiDisksCount, либо при равенстве нулю значения diMountPointsCount.

**ERR\_FAILED**

Количество точек монтирования в pMountPoints не соответствует количеству точек монтирования, полученному при анализе pDisks^.pMountPoints, либо количество дисковых устройств pDisks не совпадает с количеством дисковых устройств, полученном при анализе pMountPoints.

Пример**PROGRAM PLC\_PRG****VAR**

```
disks : ARRAY [1..10] OF MountDisk;
iDisks : DINT;
mountPoints : ARRAY [1..10] OF MountPoint;
iMountPoints : DINT;
Result : RTS_IEC_RESULT;
```

**END\_VAR**

```
iDisks := SIZEOF(disks)/SIZEOF(disks[1]);
iMountPoints := SIZEOF(mountPoints)/SIZEOF(mountPoints[1]);
```

```
Result := SysBoardDeviceGetAllDisks(ADR(disks), ADR(iDisks), ADR(mountPoints), iMountPoints);
```

**6.7.3.2. SysBoardDeviceGetDiskInfo**

Функция предназначена для получения информации о дисковом устройстве, которому принадлежит заданная точка монтирования раздела.

Прототип**FUNCTION SysBoardDeviceGetDiskInfo : RTS\_IEC\_RESULT****VAR\_INPUT**

```
pMountPoint : POINTER TO MountPoint;
pDiskInfo : POINTER TO MountDiskInfo;
```

**END\_VAR**Входные параметры**pMountPoint : POINTER TO MountPoint**

Указатель на переменную типа MountPoint (см. п. 6.7.2.3), содержащую описатель точки монтирования раздела съемного накопителя.

**pDiskInfo : POINTER TO MountDiskInfo**

Указатель на переменную типа MountDiskInfo (см. п. 6.7.2.2), в которую при успешном вызове будет помещено описание системного устройства, соответствующего дисковому накопителю, которому принадлежит точка монтирования раздела pMountPoint.

Возвращаемый результат**ERR\_OK**

Успех.

**ERR\_FAILED**

Любой из указателей, переданных функции, равен 0, либо описатель точки монтирования содержит имя системного устройства, не являющегося дисковым накопителем.

Пример**PROGRAM PLC\_PRG**



```

VAR
  disks : ARRAY [1..10] OF MountDisk;
  iDisks : DINT;
  mountPoints : ARRAY [1..10] OF MountPoint;
  iMountPoints : DINT;
  Result : RTS_IEC_RESULT;
  diskInfo : MountDiskInfo;
END_VAR

iDisks := SIZEOF(disks)/SIZEOF(disks[1]);
iMountPoints := SIZEOF(mountPoints)/SIZEOF(mountPoints[1]);
Result := SysBoardDeviceGetAllDisks(ADR(disks), ADR(iDisks), ADR(mountPoints), iMountPoints);
IF Result = ERR_OK THEN
  // получаем описатель дискового устройства, которому принадлежит первая точка
  // монтирования раздела
  Result := SysBoardDeviceGetDiskInfo(ADR(mountPoints[1]), ADR(diskInfo));
END_IF

```

### 6.7.3.3. SysBoardDeviceGetAllMountPoints

Функция предназначена для получения информации обо всех точках монтирования разделов подключенных съемных дисковых накопителей.

#### Прототип

```

FUNCTION SysBoardDeviceGetAllMountPoints : RTS_IEC_RESULT
VAR_INPUT
  pMountPoints : POINTER TO MountPoint;
  pMountPointsCount : POINTER TO DINT;
END_VAR

```

#### Входные параметры

**pMountPoints : POINTER TO MountPoint**

Указатель на массив типа MountPoint (см. п. 6.7.2.3), в который будут помещены описания всех точек монтирования разделов съемных дисковых накопителей при успешном вызове.

Если данный параметр равен 0 при неравенстве 0 второго параметра, то в переменной с адресом pMountPointsCount будет возвращено количество точек монтирования разделов съемных дисковых накопителей.

**pMountPointsCount : POINTER TO DINT**

Указатель на переменную типа DINT, в которую перед вызовом следует поместить количество элементов в массиве pMountPoints. После успешного вызова функции в данную переменную будет помещено количество описаний обнаруженных точек монтирования разделов съемных дисковых накопителей.

#### Возвращаемый результат

**ERR\_OK**

Успех.

**ERR\_FAILED**

pMountPointsCount равен 0, либо равно 0 значение по указателю pMountPointsCount.

#### Пример

```

PROGRAM PLC_PRG
VAR
  mountPoints : ARRAY [1..10] OF MountPoint;
  iMountPoints : DINT;
  Result : RTS_IEC_RESULT;
END_VAR

iMountPoints := SIZEOF(mountPoints)/SIZEOF(mountPoints[1]);
// получаем точки монтирования разделов съемных дисковых накопителей
Result := SysBoardDeviceGetAllMountPoints (ADR(mountPoints), ADR(iMountPoints));

```

### 6.7.3.4. SysBoardDeviceEject

Функция предназначена для программного отключения съемного дискового накопителя, после которого возможно безопасно извлечь дисковый накопитель из гнезда контроллера.

Обратите внимание, что перед вызовом данной функции следует закрыть все файлы и каталоги, ранее открытые на всех разделах извлекаемого дискового накопителя.

После успешного вызова данной функции все ранее открытые системные идентификаторы (хэндлы) файлов и каталогов становятся недействительными.

В качестве первого параметра функция принимает имя системного устройства, соответствующего либо всему дисковому накопителю, либо точке монтирования раздела на дисковом накопителе. Однако при выполнении функции отключение производится в отношении всего дискового накопителя.

#### Прототип

```
FUNCTION SysBoardDeviceEject : RTS_IEC_RESULT
VAR_INPUT
    szDevice : POINTER TO STRING;
    uiFlags : DINT;
END_VAR
```

#### Входные параметры

**szDevice : POINTER TO STRING**

Указатель на строку, которая содержит имя системного устройства, соответствующего съемному дисковому накопителю или разделу на съемном дисковом накопителе.

**uiFlags : DINT**

Должен быть равен EJECT\_FLAG\_PARENT для устройства, соответствующего разделу на съемном дисковом накопителе.

Должен быть равен 0 для устройства, соответствующего съемному дисковому накопителю.

#### Возвращаемый результат

**ERR\_OK**

Успех, можно извлечь накопитель из гнезда.

**ERR\_FAILED**

Функция вызвана в отношении устройства, которое не является дисковым накопителем, либо отсутствует в системе, либо устройство не может быть извлечено, либо на данном дисковом накопителе остались открытые файлы или каталоги.

#### Пример

```
PROGRAM PLC_PRG
VAR
    // Дисковое устройство с данным разделом требуется извлечь
    szPathToEject : STRING := './user/microsd1';
    mountPoints : ARRAY [1..10] OF MountPoint;
    iMountPoints : DINT;
    Result : RTS_IEC_RESULT;
    iIndex : DINT;
END_VAR

// получаем точки монтирования разделов съемных дисковых накопителей
iMountPoints := SIZEOF(mountPoints)/SIZEOF(mountPoints[1]);
Result := SysBoardDeviceGetAllMountPoints (ADR(mountPoints), ADR(iMountPoints));

IF ERR_OK = Result AND iMountPoints > 0 THEN
    // если успешно, и есть смонтированные разделы, ищем диск с нужным смонтированным разделом
    FOR iIndex := 1 TO iMountPoints DO
        IF mountPoints[iIndex].szPath = szPathToEject THEN
            // нашли, делаем отключение диска
            Result := SysBoardDeviceEject(ADR(mountPoints[iIndex].szDevice), EJECT_FLAG_PARENT);
            // выходим из цикла
            EXIT;
        END_IF
    END_FOR
END_IF
```

END\_IF

**ПРИЛОЖЕНИЕ 1. ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

Версия	Дата	Ссылка	Статус	Примечания
0.1	28.07.2023	Документ	создан	Предварительный выпуск для валидации
1.0	22.09.2023	Документ	изменен	Первый выпуск