

НАСТРОЙКА И РАБОТА REGUL OPC DA SERVER

Руководство пользователя

DPA-302.5

Версия документа 2.0

Версия ПО 2.0.1

Декабрь 2020

История изменений руководства пользователя

Версия руководства пользователя	Описание изменения
2.0	<p>Добавлена история изменений руководства пользователя.</p> <p>Добавлены знаки с предупреждающей и поясняющей информацией.</p> <p>Добавлены новые разделы:</p> <ul style="list-style-type: none">– «Управление экземплярами OPC сервера»;– «Поддержка целостности данных»;– «Обращение в службу технической поддержки». <p>Дополнительно по тексту внесены небольшие изменения с уточняющей информацией</p>

АННОТАЦИЯ

Настоящий документ содержит сведения об установке и работе в приложении Regul OPC DA Server. OPC DA Server предназначен для доступа к экспортируемым переменным (символьной конфигурации проекта) приложения Epsilon LD через интерфейс OPC DA.

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

- иметь, как минимум, среднее техническое образование;
- приступить к работе только после изучения данного руководства.


Обновление информации в Руководстве

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


Для своевременного отслеживания выхода новой версии Руководства рекомендуется оформить подписку на обновление документа. Для этого необходимо на сайте Производителя: <https://www.prosoftsystems.ru/> во вкладке «Документация» под иконками документов кликнуть на кнопку «Подписаться на обновления» и оставить свои контактные данные.

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

ПРЕДУПРЕЖДАЮЩИЕ ЗНАКИ

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

ИНФОРМАЦИОННЫЕ ЗНАКИ

	<p>ИНФОРМАЦИЯ</p> <p>Здесь следует обратить внимание на <u>важную</u> информацию</p>
---	---

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
УСТАНОВКА ПРОГРАММЫ	7
НАСТРОЙКА КОНФИГУРАЦИИ.....	9
Основные параметры	9
Резервирование	10
ПЛК	11
ОРС.....	13
Безопасность	13
Интерфейс приложения	13
Журнал событий.....	14
Управление экземплярами ОРС сервера	15
Добавление экземпляра	16
Удаление экземпляра	16
Запуск экземпляра.....	17
ОПИСАНИЕ РАБОТЫ СЕРВЕРА	18
НАСТРОЙКА СРЕДЫ EPSILON LD.....	21
ПОДДЕРЖКА РЕЗЕРВИРОВАНИЯ.....	23
Аппаратное резервирование	23
Программное резервирование	23
Вспомогательные механизмы.....	23
ПОДДЕРЖКА ЦЕЛОСТНОСТИ ДАННЫХ	24
ОБРАЩЕНИЕ В СЛУЖБУ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ.....	27
ПРИЛОЖЕНИЕ А. АЛГОРИТМ ВЫБОРА АКТИВНОГО КОНТРОЛЛЕРА	28
ПРИЛОЖЕНИЕ Б. ОГРАНИЧЕНИЕ ПРОПУСКНОЙ СПОСОБНОСТИ	29
ПРИЛОЖЕНИЕ В. ОПИСАНИЕ ФАЙЛА КОНФИГУРАЦИИ.....	31

ВВЕДЕНИЕ

Основная цель настоящего документа – дать пользователю базовые знания о том, как осуществить доступ к экспортируемым переменным приложения Epsilon LD через интерфейс OPC DA Server (далее – OPC сервер). Приложение служит для работы с контроллерами серии Regul RX00 как в одиночном исполнении, так и в составе резервированной системы.

При работе с контроллером OPC сервер обеспечивает:

- подключение по заданным IP-адресам к одному или двум (схема с резервированием и без) контроллерам Regul; по одному или двум сетевым интерфейсам к каждому контроллеру;
- доступ к экспортируемым переменным Epsilon LD проекта (символьной конфигурации) для чтения/записи произвольных наборов данных как с активного контроллера (в схеме с резервированием), так и с каждого контроллера отдельно;
- синхронизацию операций чтения/записи переменных с ИЕС-задачами (начиная с версии ПО 2.0.1).

Со стороны OPC-интерфейса:

- поддержка интерфейсов OPC DA 2.05a (со стороны OPC-клиентов рекомендуется использовать синхронный *IOPCSyncIO* и асинхронный *IOPCAsyncIO2* интерфейсы, асинхронный интерфейс *IOPCAsyncIO* не поддерживается);
- возможность назначения OPC-групп для работы как с тегами активного контроллера, так и для работы отдельно с каждым контроллером напрямую;
- поддержка переменных всех простых типов ИЕС, включая одномерные массивы простых типов; доступ к конечным элементам сложных типов данных (структуры, вложенные структуры, многомерные массивы, массивы массивов, массивы структур и т.д.). Также поддерживается: чтение одно/многомерных массивов простых типов и структур, массива массивов. Не поддерживается чтение: корневых элементов структур; элементов структур (вложенных структур), объявленных по ссылке; отдельных подмассивов для массива массивов;
- поддержка VBA-клиентов (только до версии 2.0.1) с предоставлением библиотеки типов;
- возможность дополнительной регистрации в системе независимых экземпляров OPC сервера (начиная с версии ПО 2.0.1).

В системе резервирования OPC сервер обеспечивает:

- четкий и однозначный алгоритм выбора активного контроллера (см. Приложение А);
- задержку переключения на резервный канал (интерфейс) при наличии резервного подключения (интерфейса) с момента обнаружения обрыва связи по активному каналу – примерно один период опроса данных с контроллера.



ВНИМАНИЕ!

Новый функционал, включающий возможность синхронизации чтения/записи переменных с IEC-задачами и регистрацию дополнительных независимых экземпляров OPC сервера, не доступен для более ранних версий ПО (до 1.0.36 включительно)

Перечень рекомендуемых документов

Для получения дополнительной подробной информации по настройке контроллеров серии Regul RX00 рекомендуется ознакомиться со следующими документами (доступны на сайте <http://www.prosoftsystems.ru>):

- Программное обеспечение Epsilon LD. Руководство пользователя;
- Regul R600. Системное руководство;
- Regul R500. Системное руководство;
- Regul R200. Системное руководство;
- Конфигурирование резервированной системы на контроллерах серии REGUL RX00. Руководство пользователя.

УСТАНОВКА ПРОГРАММЫ

Для корректной работы программы (до версии ПО 1.0.36 включительно), перед установкой, необходимо удалить все ранее установленные версии с компьютера. Начиная с версии ПО 2.0.1, программа может быть установлена параллельно с одной из предыдущих версий (ПО 1.0.36 и более ранней). Деинсталляция производится стандартными средствами операционной системы.

Получите от предприятия-изготовителя файл установки с именем **ps_regul_opcda_setup_vX.X.X.exe**, где vX.X.X– номер версии программного обеспечения, (например, *ps_regul_opcda_setup_v2.0.1.exe*). Запустите файл от имени администратора. Откроется окно выбора языка установки (Рисунок 1). Выберите язык и нажмите кнопку **ОК**, и в открывшемся окне приветствия нажмите кнопку *Далее*.

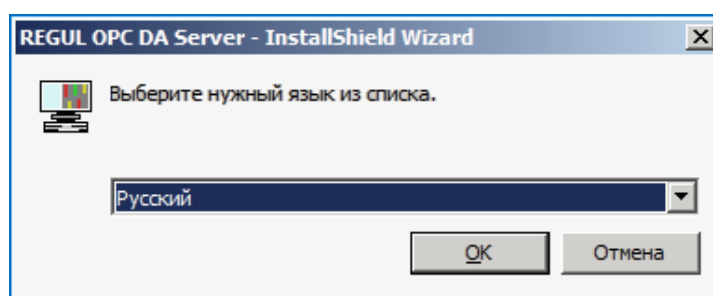


Рисунок 1 – Окно выбора языка установки

Откроется окно для выбора местоположения программы (Рисунок 2). По умолчанию программа установки создает папку **Prosoft-Systems** в каталоге **Program Files** (если операционная система Windows 32-bit) или в каталоге **Program Files (x86)** (если операционная система Windows 64 bit). Рекомендуется не изменять предложенный по умолчанию путь к установке программы.

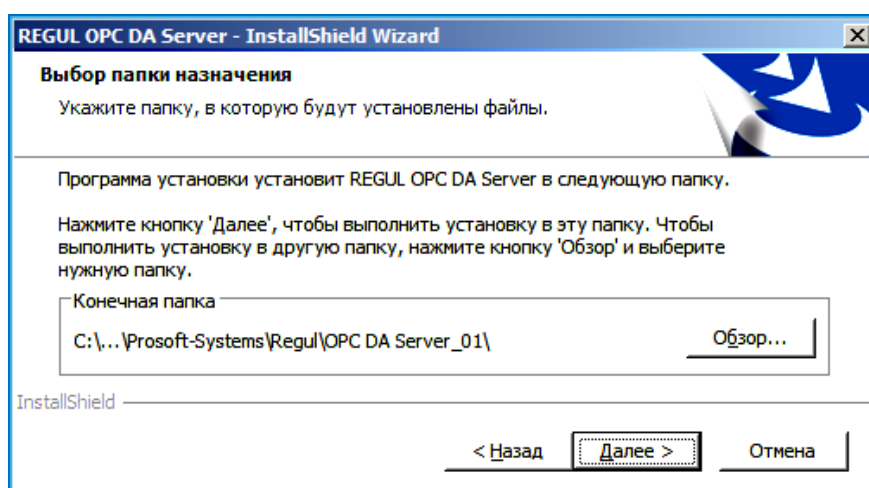


Рисунок 2 – Диалоговое окно выбора папки, куда будет установлена программа

В окне **Выбор папки назначения** нажмите кнопку *Далее*. Откроется окно выбора компонентов программы для установки (Рисунок 3).

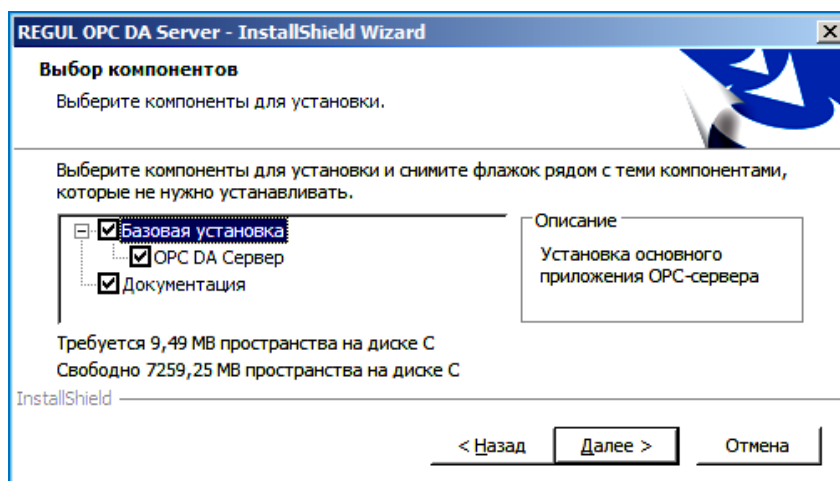


Рисунок 3 – Диалоговое окно выбора компонентов для установки


Поставьте флажок напротив нужных компонентов. Под перечнем компонентов в строке: **Требуется __ МВ пространства на диске C** отображается объем свободного пространства жесткого диска, требуемый для корректной установки всех выбранных компонентов.

Представлен следующий перечень компонентов:

- **Базовая установка** – установка основного экземпляра **OPC DA Сервер**;
- **Документация** – руководство пользователя по установке и настройке **OPC DA Сервера Regul**.

При выборе компонента **OPC DA Сервер** будет установлен один экземпляр OPC сервера, позволяющий работать с одиночным ПЛК либо с двумя ПЛК в резерве и доступный для OPC-клиентов под именем **psregulopcda_01**. При необходимости, пользователь может динамически зарегистрировать дополнительные экземпляры OPC сервера, что даст возможность подключения с одного АРМа к требуемому количеству ПЛК. Подробное описание приведено в разделе «Управление экземплярами OPC сервера».

После выбора компонентов для установки нажмите кнопку **Далее**. Откроется диалоговое окно с сообщением, что мастер готов выполнить установку *Regul OPC DA Server*. Чтобы начать установку нажмите кнопку **Установить**.

Начнется копирование файлов и установка программы, что может занять некоторое время. Дождитесь окончания процесса и появления оповещения об успешном окончании установки. Нажмите кнопку **Готово**. По окончании установки программы автоматически будет создан ярлык в меню **Пуск** , которым можно воспользоваться для ручного запуска приложения.

В ходе установки исполняемый файл будет автоматически зарегистрирован как COM OPC DA сервер и станет доступен для подключения OPC-клиентам. При этом язык интерфейса установленного приложения, по умолчанию, будет соответствовать языку, выбранному в начале установки

НАСТРОЙКА КОНФИГУРАЦИИ

После установки приложения конфигурационный файл **config.ini** будет расположен в папке:
 c:\ProgramData\Prosoft-Systems\Regul\OPC DA Server_01\ (папка может быть скрыта)

Полное описание параметров, содержащихся в файле **config.ini**, приведено в приложении В. Основные параметры, связанные с параметрами конфигурационного файла, вынесены в диалог настроек приложения.

Основные параметры

Запустите приложение и перейдите в основное меню **Инструменты** ⇒ **Настройки...** (Рисунок 4)

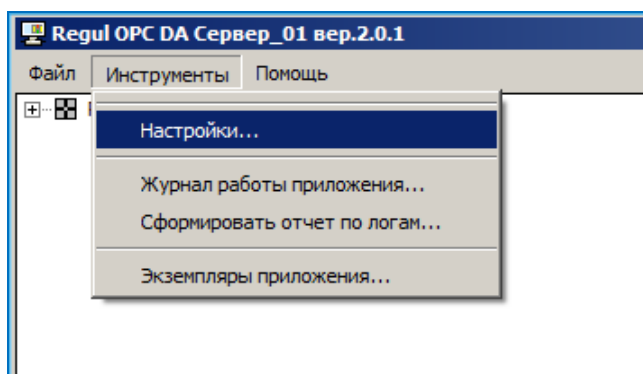


Рисунок 4 - Пункт основного меню Инструменты

В окне **Настройки** (Рисунок 5) доступны следующие подгруппы настроек: **Резервирование**, **ПЛК**, **OPC**, **Безопасность**, **Интерфейс приложения**.

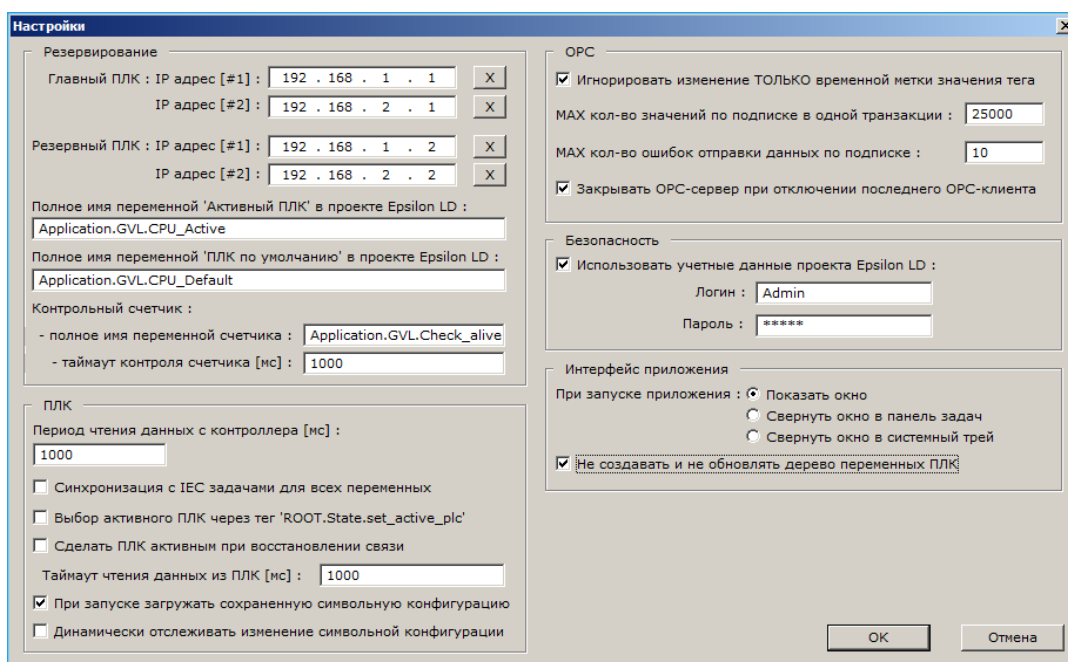



Рисунок 5 – Основные параметры настройки конфигурации

Резервирование

Таблица 1 – Описание параметров настройки резервирования

Параметр	Описание
Главный/Резервный ПЛК: IP адрес [#1/2]	<p>IP-адреса каналов подключения к двум контроллерам по двум интерфейсам для каждого ПЛК. Предполагается работа приложения либо с двумя ПЛК в резерве, либо с двумя ПЛК, на которых загружен один и тот же проект.</p> <p>По умолчанию, IP адреса #1 принадлежат к одной независимой подсети, а #2 – к другой. Например:</p> <p>Главный ПЛК IP адрес [#1] = 192.168.28.1/24</p> <p>Главный ПЛК IP адрес [#2]= 192.168.34.1/24</p> <p>Резервный ПЛК IP адрес [#1]= 192.168.28.2/24</p> <p>Резервный ПЛК IP адрес [#2]= 192.168.34.2/24</p> <p>Допускается настройка подключения к одиночному контроллеру REGUL по одному или двум интерфейсам.</p>
	<p>ВНИМАНИЕ!</p> <p>Настройка подключения к ПЛК с разными проектами недопустима!</p>
Переменные 'Активный ПЛК' и 'ПЛК по умолчанию' в проекте Epsilon LD	<p>При работе с двумя ПЛК в резерве OPC сервер определяет активный ПЛК на основании значений двух флагов – переменных, которые добавлены в символьную конфигурацию проекта и считываются с самих ПЛК</p>

При наличии в проекте компонента **Резервирование / Redundancy** (см. документацию «Конфигурирование резервированной системы на контроллерах серии Regul RX00») указанные переменные 'Активный ПЛК' и 'ПЛК по умолчанию', определяются, например, следующим способом:

Окно объявлений ROU

```
Stats : PsRedundancy.TStat2;
RedMode : SINT;
IsStateActive : BOOL; //флаг активности ПЛК
IsDefaultPlc : BOOL; //флаг ПЛК по умолчанию
```

Окно кода ROU

```
PsRedundancy.Synchronize3(FALSE, Stats);
RedMode := PsRedundancy.GetMode();
IsStateActive := (RedMode=PsRedundancy.ACTIVE) OR
(RedMode=PsRedundancy.ACTIVE_STANDALONE);
IsDefaultPlc := PsRedundancy.IsCpuA();
```

Соответственно, в этих полях настроек задается полный путь к этим переменным в виде *Имя_Приложения_Проекта / Имя_ROU / Имя_Переменной*, (Рисунок 6)

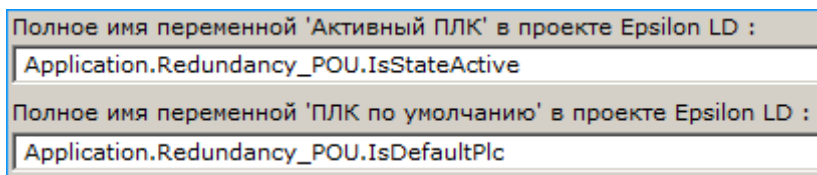


Рисунок 6 – Пример записи параметров



ВНИМАНИЕ!

В случае обрыва связи с ПЛК, когда OPC сервер не может получить актуальные значения указанных флагов, выбор активного ПЛК с точки зрения OPC сервера может не совпадать с фактическим состоянием подсистемы резервирования двух ПЛК

Продолжение таблицы 1

Параметр	Описание
Контрольный счетчик	Полное имя переменной счетчика проекта Epsilon LD, реализующей дополнительный механизм контроля состояния приложения/среды выполнения ПЛК. Переменная должна иметь тип UDINT и представляет собой простой счетчик, инкрементируемый в одной из задач приложения. Переменная должна быть добавлена в символьную конфигурацию проекта. При наличии подключения к контроллеру OPC сервер периодически вычитывает значение счетчика и, если его значение не меняется в течение периода, заданного в поле таймаута счетчика, то считается, что приложение на данном контроллере находится в состоянии ошибки (исключения/останова)

ПЛК

Таблица 2 – Описание параметров настройки ПЛК

Параметр	Описание
Период чтения данных с контроллера [мс]	Период запроса (частота обновления в понимании «не чаще чем») текущих значений всех переменных контроллера (точнее, подмножества переменных, на которые есть подписка от OPC-клиентов). При большом количестве считываемых в одном цикле данных и из-за ограниченной пропускной способности канала подключения, заданный период может не выдерживаться. В этом случае следующий цикл чтения будет начинаться сразу после окончания предыдущего
Синхронизация с ИЕС задачами для всех переменных	Включает режим синхронизации с ИЕС-задачами при чтении/записи всех переменных ПЛК

Параметр	Описание
<p>Выбор активного ПЛК через тег 'ROOT.State.set_active_plc'</p>	<p>Если этот флаг включен, то внешний OPC-клиент, через тег Root.State.set_active_plc, может задавать безусловный выбор активного контроллера вне зависимости от наличия с ним связи. То есть, если связь с ПЛК есть, то данные берутся из него, если связи нет, то все теги в общей группе PLC будут BAD и переключения на другой контроллер не произойдет. По умолчанию, этот флаг выключен (настройка относится к алгоритму выбора активного ПЛК в случае работы с двумя ПЛК без резервирования с одинаковыми проектами)</p> <p>При этом OPC сервер пробует сделать активным контроллер, заданный в Root.State.set_active_plc, если же с ним связи нет, то активным будет второй контроллер (если он на связи). Также поведение зависит от следующего параметра</p>
<p>Сделать ПЛК активным при восстановлении связи</p>	<p>Если этот флаг включен, то, после восстановления связи с контроллером, заданным в Root.State.set_active_plc, он снова станет активным. По умолчанию, этот флаг выключен (настройка относится к алгоритму выбора активного ПЛК в случае работы с двумя ПЛК без резервирования с одинаковыми проектами). При этом:</p> <ul style="list-style-type: none"> – при записи в Root.State.set_active_plc будет выполнен следующий алгоритм – активным будет либо заданный контроллер, либо второй (в зависимости от наличия связи); – если через Root.State.set_active_plc был выбран ПЛК1, затем с ПЛК1 связь пропала, активным стал ПЛК2, затем связь с ПЛК1 восстановилась – активным останется ПЛК2
<p>Таймаут чтения данных из ПЛК [мс]</p>	<p>Таймаут обмена данными между OPC сервером и ПЛК, в мс. Влияет на стабильность каналов подключения к ПЛК на чтение при запуске OPC сервера на нескольких АРМах и большом количестве считываемых переменных. В случае отображения в журнале сообщений периодических ошибок чтения с ПЛК рекомендуется увеличить данный период</p>
<p>При запуске загружать сохраненную символьную конфигурацию</p>	<p>В процессе работы приложения при подключении к ПЛК и считывании с него символьной конфигурации (списка открытых переменных) эта информация сохраняется в файл. При включении этой настройки OPC сервер в момент старта (до подключения к ПЛК!) может загружать данные из этого файла и сразу предоставлять OPC-клиентам информацию по доступности переменных ПЛК, их типам и правам доступа. Рекомендуется включить эту опцию по умолчанию</p>
<p>Динамически отслеживать изменение символьной конфигурации</p>	<p>В процессе отладки работы с ПЛК может меняться его символьная конфигурация, могут появиться новые переменные или быть удалены ранее доступные. Может поменяться тип переменных и пр. Для отслеживания подобных изменений без перезапуска OPC сервера требуется включить данную настройку. Однако, процесс сравнения предыдущей символьной конфигурации ПЛК с предположительно новой запускается при каждом подключении/восстановлении связи с ПЛК и является затратной по времени процедурой, особенно при большом количестве переменных. Поэтому включать эту опцию рекомендуется только на момент отладки проекта Epsilon LD</p>

OPC

Таблица 3 – Описание параметров настройки OPC

Параметр	Описание
Игнорировать изменение ТОЛЬКО временной метки значения тега	Если эта настройка включена, то значения тегов будут обновляться при изменении значения (value) или качества (quality) переменной контроллера; изменения только временной метки (timestamp) игнорируются. При выключении этой опции теги будут обновляться, даже если изменилась только временная метка (синхронный режим)
МАХ кол-во значений по подписке в одной транзакции	Определяет максимальное число новых значений тегов, отправляемых OPC-клиенту по подписке в одной транзакции (OnDataChange). Относится к оптимизации работы OPC-клиентов, блокирующих обратный вызов передачи данных по подписке для синхронной обработки новых значений
МАХ кол-во ошибок отправки данных по подписке	Задаёт максимальное количество ошибок отправки данных OPC-клиенту по подписке (OnDataChange), после которого данная OPC группа деактивируется в предположении, что OPC-клиент нештатно разорвал подключение
Закрывать OPC сервер при отключении последнего OPC-клиента	Если эта настройка включена, то приложение будет автоматически выгружено при отключении последнего OPC-клиента (кроме случаев, когда OPC сервер изначально был запущен вручную)

Безопасность

Таблица 4 – Описание параметра настройки безопасности

Параметр	Описание
Использовать учетные данные проекта Epsilon LD	Позволяет подключаться к контроллеру с учетными данными пользователя Online User проекта Epsilon LD

Интерфейс приложения

Таблица 5 – Описание параметров настройки интерфейса приложения

Параметр	Описание
При запуске приложения...	Определяет отображение главного окна приложения при запуске: <ul style="list-style-type: none"> – обычное окно, – свернутое в панель управления, – в системный трей
Не создавать и не обновлять дерево переменных ПЛК	Если эта настройка включена, то, после подключения к ПЛК и получения его символьной конфигурации, дерево переменных в окне приложения отображаться не будет. Иначе, при подключении к ПЛК в окне приложения отображается дерево всех (!!!) переменных из символьной конфигурации в 6-кратном объеме (ветки PLC1, PLC2, PLC_consistent, PLC1_consistent и PLC2_consistent являются дубликатами PLC). Если нет необходимости в наблюдении текущих значений переменных в окне

Параметр	Описание
	<p>приложения самого OPC сервера, то рекомендуется отключить построение дерева переменных с помощью этой настройки. Иначе, при большом количестве экспортируемых в символьной конфигурации переменных проекта (от 10000 и более), построение дерева занимает продолжительное время и блокирует графический интерфейс пользователя. Обновление значений переменных внутреннего представления OPC сервера при этом создает дополнительную нагрузку на процессор АРМа.</p> <p>Другим вариантом является уменьшение отмеченных галочками переменных в символьной конфигурации проекта Epsilon LD до минимального необходимого объема</p>

Соотнесение основных параметров приложения с параметрами файла конфигурации представлено в таблице В.1 приложения В.

Журнал событий

Перейдите в пункт меню **Инструменты** ⇒ **Журнал работы приложения...**(Рисунок 7).

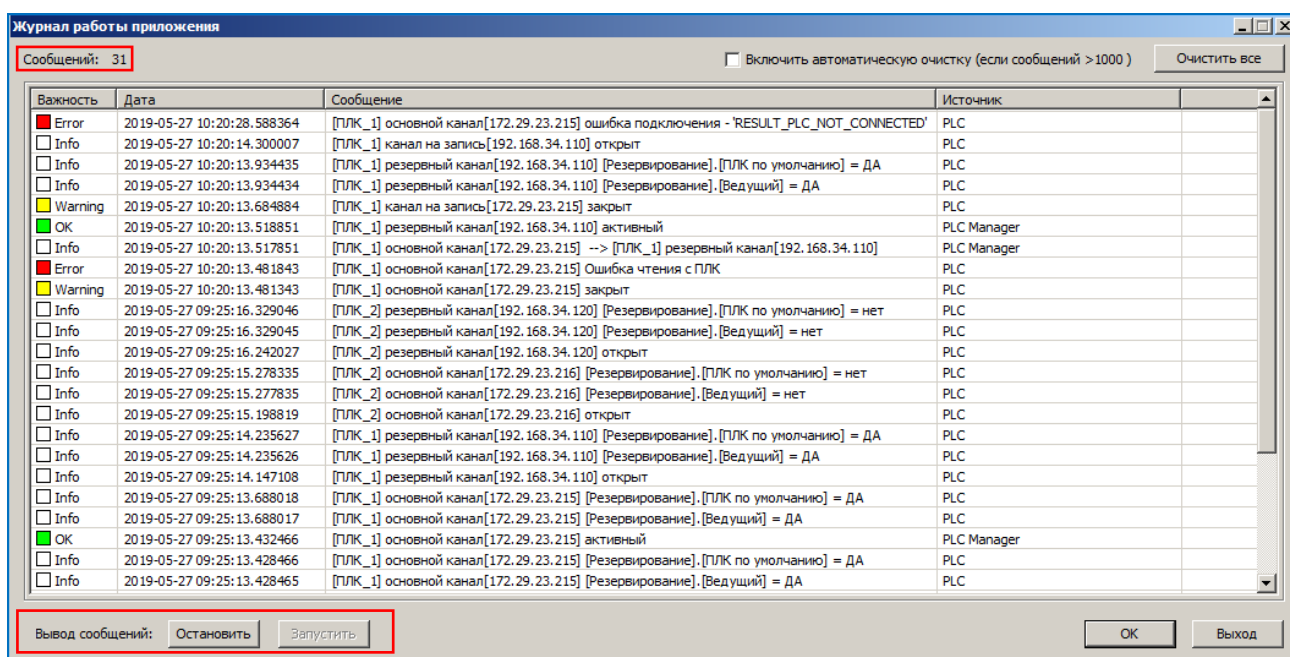


Рисунок 7 – Журнал работы приложения

Журнал служит для отображения событий, позволяющих пользователю наблюдать за ходом подключения к ПЛК и самостоятельно выявлять следующие исключительные ситуации:

- отсутствие связи с ПЛК по указанному IP-адресу (ошибка ping);
- неверные (или отсутствующие) логин и пароль для подключения к ПЛК;
- переключение активного канала связи с ПЛК по причине:
 - изменения состояния объекта *Redundancy*,
 - потери связи с ПЛК по отдельному интерфейсу,

- ошибки чтения/записи данных
- остановки контрольного счетчика (ошибка среды выполнения ПЛК при возникновении исключительной ситуации);
- ошибки запроса данных со стороны OPC-клиентов для переменных, отсутствующих в символьной конфигурации ПЛК.

В верхней части окна отображается количество накопленных сообщений. Существует ограничение в 1000 сообщений, при превышении которого вывод новых сообщений приостанавливается, о чем выводится предупреждающее сообщение (Рисунок 8).

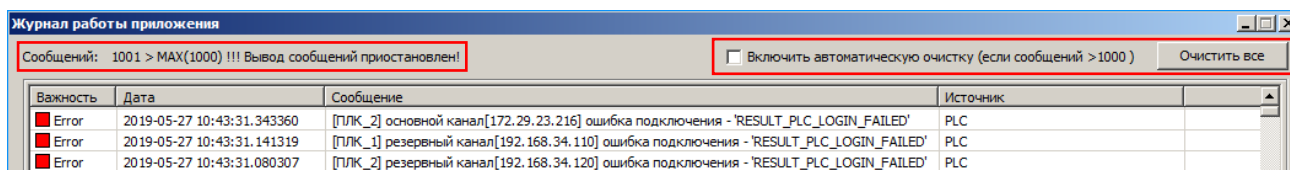


Рисунок 8 – Предупреждающее сообщение

В этом случае пользователь может либо очистить все текущие сообщения вручную, нажав кнопку **Очистить все**, либо установить флажок в строке **Включить автоматическую очистку...** Новые сообщения добавляются вверх списка. После очистки необходимо нажать кнопку **Запустить** для возобновления вывода сообщений в окно.

Пункт меню **Инструменты** ⇒ **Сформировать отчет по логам** при возникновении нештатной ситуации в работе OPC сервера позволяет автоматически сохранить в один архив все лог-файлы приложения и файл настроек для дальнейшей отправки в службу техподдержки Prosoft-Systems (см. раздел «Обращение в службу технической поддержки»).

Управление экземплярами OPC сервера

По умолчанию, устанавливается один экземпляр OPC сервера, доступный для OPC-клиентов по имени *psregulopcda_01*. Он позволяет подключиться к одному ПЛК либо к двум ПЛК (в резерве или с одинаковыми проектами). В ситуациях, когда с одного АРМа требуется производить контроль более чем одной подсистемой управления, построенной на базе ПЛК REGUL, OPC сервер предоставляет возможность добавления дополнительных независимых экземпляров приложения с идентичной функциональностью.

Для добавления/удаления экземпляров приложения, выберите в главном меню пункт **Инструменты** ⇒ **Экземпляры приложения** (Рисунок 9). В открытом диалоге будет отображаться список зарегистрированных в системе экземпляров OPC сервера. Сразу после установки будет доступен единственный экземпляр.

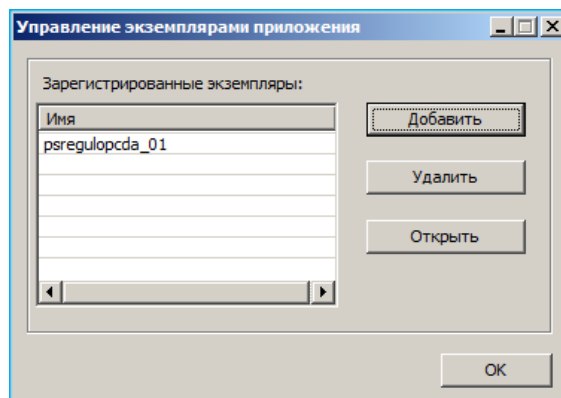


Рисунок 9 – Диалоговое окно управления экземплярами

Добавление экземпляра

Чтобы добавить новый экземпляр приложения, нажмите на кнопку *Добавить*. Для добавления доступны варианты с нумерацией от 02 до 99. Программа предложит добавить экземпляр с ближайшим свободным номером. При этом вручную задать имя нового экземпляра невозможно. После подтверждения добавления и успешной регистрации в системе нового экземпляра OPC сервера, программа добавит его в список экземпляров, и он будет доступен для OPC-клиентов под указанным именем (Рисунок 10).

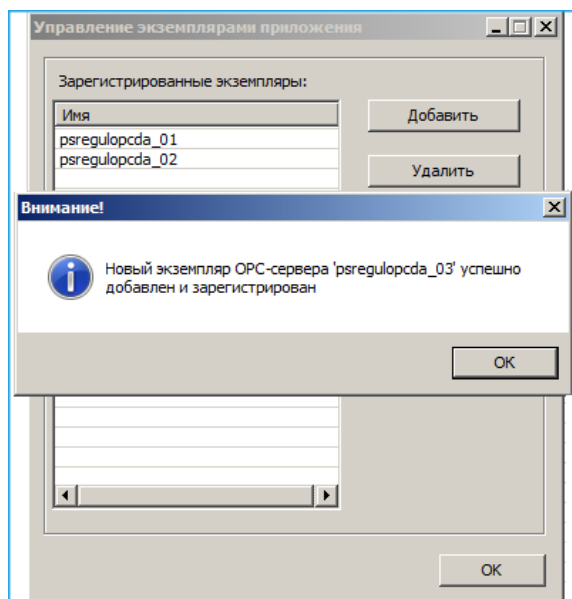


Рисунок 10 – Добавление экземпляра

Удаление экземпляра

Для удаления экземпляра необходимо выбрать его в списке и нажать кнопку *Удалить*.



ВНИМАНИЕ!

Экземпляр **psregulopda_01** недоступен для удаления с помощью указанного диалога. Он удаляется только при деинсталляции программного продукта. Также невозможно удалить собственный запущенный экземпляр приложения. Для этого нужно выполнить процедуру удаления в окне другого экземпляра.

ВАЖНО! Перед удалением программного продукта пользователю необходимо самостоятельно удалить все экземпляры (кроме первого) с помощью данного диалога

Запуск экземпляра

После регистрации нового экземпляра он становится доступным для подключения и браузинга в OPC-клиенте.

Также экземпляр можно запустить, выбрав его имя в списке и нажав кнопку **Открыть**.

ОПИСАНИЕ РАБОТЫ СЕРВЕРА

При подключении OPC сервера к контроллерам хотя бы по одному каналу (здесь канал – это подключение по одному из IP-адресов к любому ПЛК), в левой части окна приложения в ветках **Root.PLC/PLC1/PLC2/_consistent** отобразится дерево переменных символьной конфигурации проекта (см. раздел выше), загруженного в контроллер (Рисунок 11).



ВНИМАНИЕ!

Во всех ветках набор переменных будет одинаков, так как предполагается, что OPC сервер работает с двумя ПЛК в резерве, либо с двумя ПЛК, на которые загружен один и тот же проект

Имя тега	Значение	Время	Ка...	Тип	Д...
active_plc	1	2020-09-04 10:01:53.331	Good	VT_I4	R
link1_name	172.29.34.0	2020-09-04 10:01:52.271	Good	VT_BSTR	R
link1_status	1	2020-09-04 10:01:52.382	Good	VT_BOOL	R
link2_name		2020-09-04 10:01:52.271	Good	VT_BSTR	R
link2_status	0	2020-09-04 10:01:52.272	Good	VT_BOOL	R
plc_11	2	2020-09-04 10:01:53.493	Good	VT_I4	R
plc_11_ip	172.29.34.134	2020-09-04 10:01:52.223	Good	VT_BSTR	R
plc_12	0	2020-09-04 10:01:52.264297	Good	VT_I4	R
plc_12_ip		2020-09-04 10:01:52.264297	Good	VT_BSTR	R
plc_21	0	2020-09-04 10:01:52.264297	Good	VT_I4	R
plc_21_ip		2020-09-04 10:01:52.264297	Good	VT_BSTR	R
plc_22	0	2020-09-04 10:01:52.264297	Good	VT_I4	R
plc_22_ip		2020-09-04 10:01:52.264297	Good	VT_BSTR	R
plc_link_error	0	2020-09-04 10:01:52.264297	Good	VT_BOOL	R
set_active_plc	0	2020-09-04 10:01:52.264297	Good	VT_I4	R/W

Рисунок 11 - Дерево переменных проекта

В ветке **Root.State** отображаются теги состояния подключения и сетевых интерфейсов (Таблица 6).

Таблица 6 – Теги состояния подключения

Теги	Описание
Root.State.plc_xy (x – номер ПЛК, y – номер интерфейса)	Содержат код состояния подключения к контроллеру: <ul style="list-style-type: none"> – 0 – нет подключения; – 1 – подключение установлено; – 2 – подключение установлено и этот канал является активным на данный момент

Теги	Описание
Root.State.plc_xy_ip	Содержат IP-адреса каналов из конфигурационного файла
Root.State.linkX_name	Показывают номер подсети соответствующего сетевого интерфейса
Root.State.linkX_status	Статус соответствующего сетевого интерфейса (сетевой кабель подключен – 1, обрыв – 0)
Root.State.plc_link_error	Принимает значение 1 в случае потери связи между контроллерами в схеме с резервированием (оба являются активными)
Root.State.set_active_plc	Позволяет управлять выбором ведущего контроллера с верхнего уровня (АРМа) согласно алгоритму (см. Приложение А) и настроечным параметрам п.п.2.8 и 2.9 (см. Приложение В). Запись в тег значения 1 или 2 задает номер ведущего контроллера, прочие значения выключают управление с верхнего уровня
Root.State.active_plc	Отображается номер (1,2) текущего активного контроллера или 0 при его отсутствии

Ветка **Root.PLC** является «общей» – в ней отображаются значения переменных, полученных по активному на данный момент каналу связи с активного (с точки зрения OPC сервера) контроллера в схеме с резервированием (и без).

Дополнительные ветки **Root.PLC1** и **Root.PLC2** используются для обмена данными напрямую с каждым контроллером по отдельности.

Ветки **Root.PLC_consistent**, **Root.PLC1_consistent** и **Root.PLC2_consistent** используются аналогично для подключения OPC-клиента к текущему активному ПЛК и напрямую ПЛК1/ПЛК2 с поддержкой синхронизации с ИЕС задачами операций чтения/записи только для переменных из этих подгрупп.



ВНИМАНИЕ!

В интерфейсе программы отображаются/обновляются значения только тех переменных, на которые есть подписка со стороны OPC-клиентов, и которые вычитываются с контроллеров

По умолчанию, для работы напрямую с первым и вторым контроллером создаются ветки **Root.PLC1 (Root.PLC1_consistent)** и **Root.PLC2 (Root.PLC2_consistent)**. OPC-клиенты могут работать с тегами из этих веток для чтения и записи в определенный ПЛК. При OPC-браузинге путь к тегам показывается, начиная с префиксов **PLC1 (Root.PLC1_consistent)** и **PLC2 (Root.PLC2_consistent)**. Под «первым» и «вторым» ПЛК здесь понимаются контроллеры, для которых IP-адреса заданы в настройках *[plc1_port1, plc1_port2]* и *[plc2_port1, plc2_port2]* соответственно.



ВНИМАНИЕ!

Для того, чтобы OPC-группа была привязана к конкретному контроллеру, OPC-клиент должен добавлять в эту OPC-группу только OPC-теги с соответствующим префиксом – либо **PLC1 (Root.PLC1_consistent)**, либо **PLC2 (Root.PLC2_consistent)**

Если, по ошибке, в одну OPC-группу будут добавлены OPC-теги с разными префиксами, то OPC-группа будет привязана к конкретному ПЛК по префиксу последнего добавленного OPC-тега. При этом запись в эти теги будет производиться в конкретный контроллер (при наличии с ним связи), независимо от того, является ли он активным на данный момент.

Чтение переменных контроллера (при наличии с ним связи), связанных с такими OPC тегами, будет также производиться из этого контроллера независимо от того, является ли этот контроллер активным на данный момент.

При потере связи с контроллером у всех связанных с ним тегов в соответствующей подгруппе выставляется качество BAD.

Возможно включение настройки **Динамически отслеживать изменение символьной конфигурации** (см. п.п. 2.26 таблицы В.1) на время разработки и отладки проекта, когда при заливке приложения в контроллер возможно изменение символьной конфигурации проекта и это требуется отслеживать без перезапуска OPC сервера. Большое число переменных в символьной конфигурации (>10000) приводит к существенным задержкам на этапе сравнения старой и новой конфигурации, поэтому в рабочем режиме эту настройку рекомендуется отключать.

НАСТРОЙКА СРЕДЫ EPSILON LD

Для того, чтобы данные с ПЛК передавались посредством протокола OPC, необходимо в программе Epsilon LD добавить компонент **Symbol Configuration** (символьная конфигурация) в опциях проекта IEC-приложения. В контекстном меню приложения (Application) выберите **Добавить объект** (Add object) ⇒ **Symbol Configuration...** (Рисунок 12).

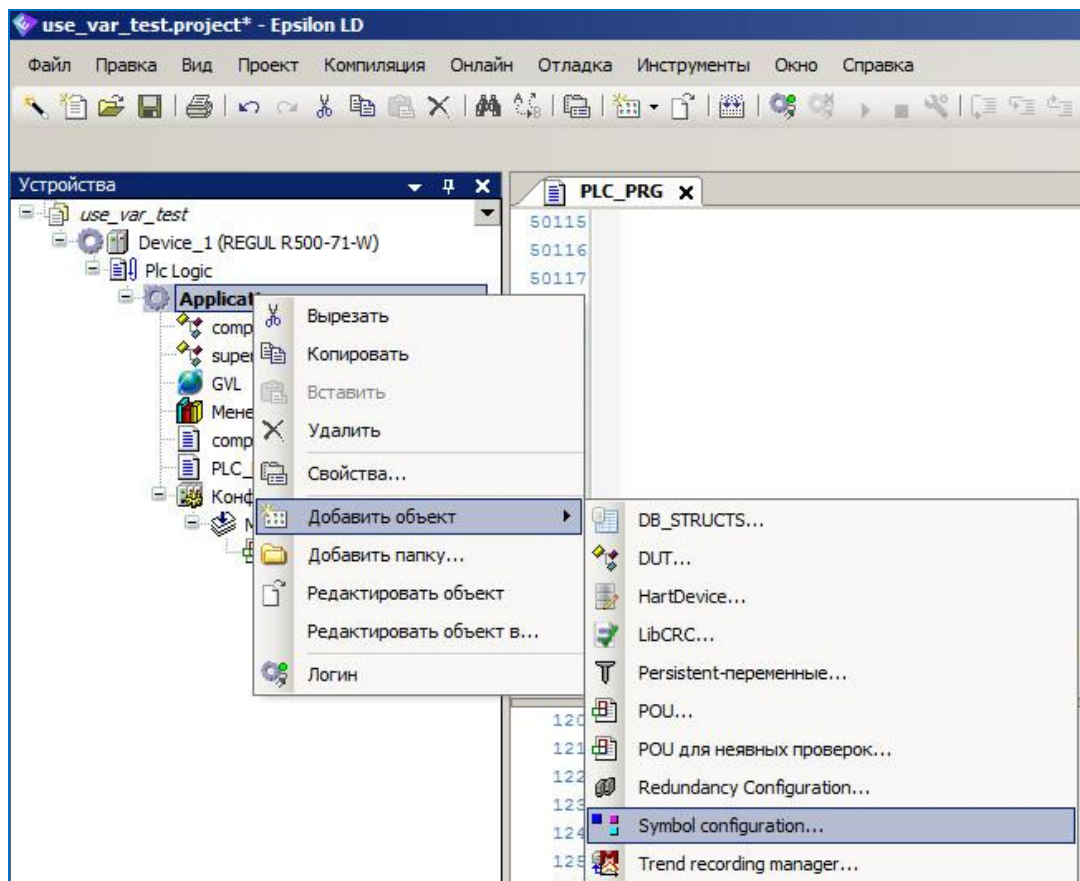


Рисунок 12 - Добавление компонента символьной конфигурации

Для добавления переменных приложения выберите объект **Symbol Configuration** в дереве устройств (двойной щелчок мыши по названию объекта). Откроется вкладка **Symbol Configuration**. Выберите нужную папку (PLC_PRG, GVL и т.д.) и в ней появится список переменных, определенных в IEC-приложении (Рисунок 13). Установите флажок напротив тех переменных, взаимодействие с которыми будет обеспечиваться протоколом OPC.

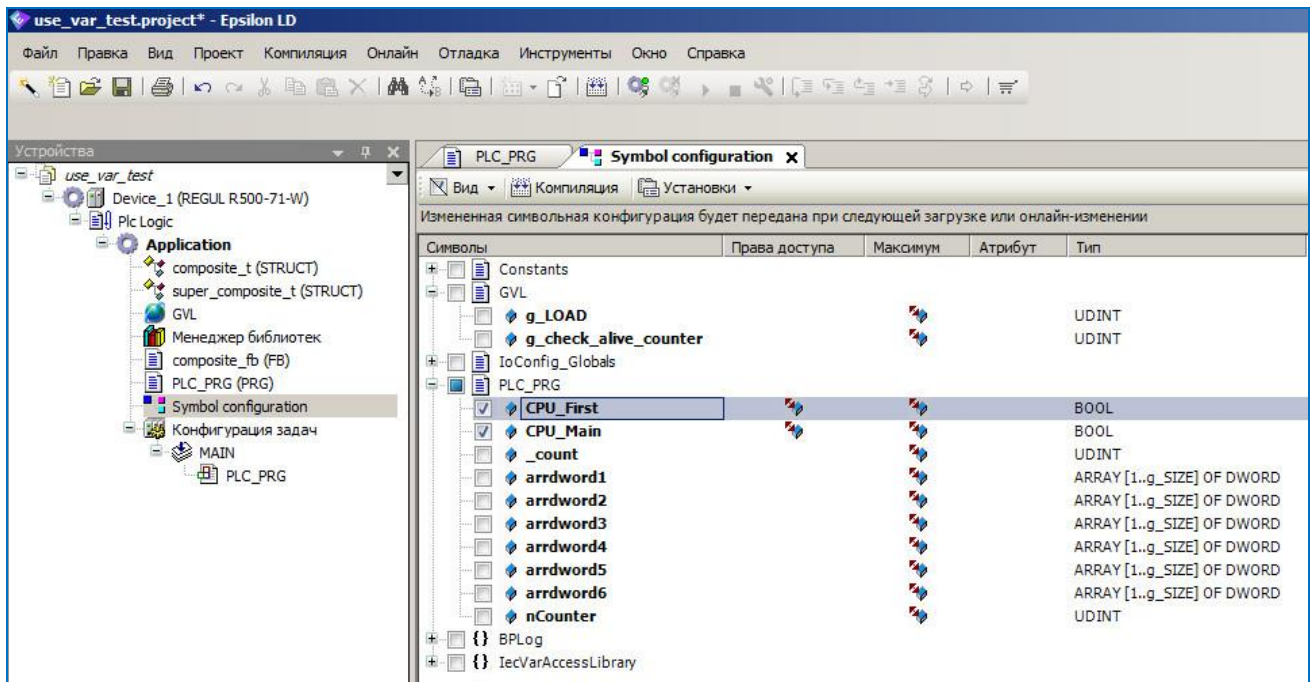


Рисунок 13 - Выбор переменных проекта для экспорта

ПОДДЕРЖКА РЕЗЕРВИРОВАНИЯ

OPC сервер предлагает набор механизмов поддержки резервирования, позволяющих отслеживать активность одного из контроллеров и, соответственно, переключать передачу данных в OPC-интерфейс по выбранному каналу обмена данными с контроллером согласно следующим алгоритмам, выбранному типу резервирования и настройкам OPC сервера.

Аппаратное резервирование

В случае использования аппаратного резервирования согласно документации «Конфигурирование резервированной системы на контроллерах серии Regul RX00» необходимо произвести настройку согласно п.п.2.6, 2.7 (см. таблицу В.1), добавив соответствующие переменные в символьную конфигурацию проекта Epsilon LD. Выбор активного контроллера при этом будет синхронизирован с работой компонента Redundancy.

Программное резервирование

Выбор активного контроллера (из двух контроллеров с одинаковым проектом) выполняется на основании:

- наличия подключения к обоим контроллерам по двум интерфейсам;
- явного выбора активного контроллера через тег **Root.State.set_active_plc**;
- настроек согласно п.п.2.5, 2.8, 2.9 (см. таблицу В.1).

Вспомогательные механизмы

Контрольный счетчик – активируется в настройках согласно п.п.2.20, 2.21 (см. таблицу В.1). Служит для обработки внештатной ситуации (исключения) в работе приложения/среды исполнения, когда все значения переменных символьной конфигурации, включая флаги аппаратного резервирования, замораживаются (с достоверным качеством значения), а выполнение приложения (рабочего цикла) останавливается. Может применяться в обоих случаях резервирования. Выполняется на самом последнем этапе выбора активного контроллера.

Состояние приложения (RUN/STOP) на контроллере – настройка согласно п.п.2.10 (см. таблицу В.1). Рекомендуется использовать в отладочных целях, так как достоверный результат обеспечивается, только если приложение на контроллере одно или все приложения на контроллере имеют одинаковый статус выполнения. Дополнительно, каждый запрос статуса приложения на ПЛК приводит к генерации системных событий Login / Logout, что может быть нежелательно для конечного пользователя.

ПОДДЕРЖКА ЦЕЛОСТНОСТИ ДАННЫХ

OPC сервер поддерживает целостность данных при записи/чтении набора переменных с помощью следующего механизма синхронизации с IEC задачами среды исполнения:

1. Клиентский запрос на чтение/запись ожидает момента, когда появится достаточный временной промежуток между выполняемыми IEC задачами;
2. Блокируется запуск IEC задач на время выполнения операции чтения/записи переменных;
3. После выполнения разрешается работа планировщика задач.



ВНИМАНИЕ!

При использовании описанного выше механизма синхронизации не гарантируется стабильная работа контроллера

Включение синхронизации возможно двумя способами:

- **Первый.** Включение в настройках OPC сервера опции **Синхронизация с IEC задачами для всех переменных** (см. таблицу 2). В этом случае синхронизация будет использована для всех операций чтения/записи для всех переменных, к которым обращаются OPC-клиенты.
- **Второй.** При подключении к OPC серверу OPC-клиент использует теги из специальных подгрупп - **Root.PLC_consistent**, **Root.PLC1_consistent** и **Root.PLC2_consistent**. При этом синхронизация включается только для операций чтения/записи наборов переменных, в которых присутствует хотя бы один тег с префиксом **Root.PLCx_consistent**.

Дополнительно требуется настройка проекта Epsilon LD. Для поддержки работы синхронизации со стороны среды исполнения **ВСЕ** приложения должны поддерживать этот режим, иначе операции чтения/записи будут завершаться с ошибкой. Для включения данной опции в приложении необходимо:

1. Для приложений **БЕЗ** объекта символьной конфигурации включение производится только через диалог **Свойства...** в контекстном меню объекта контроллера (Рисунок 14).

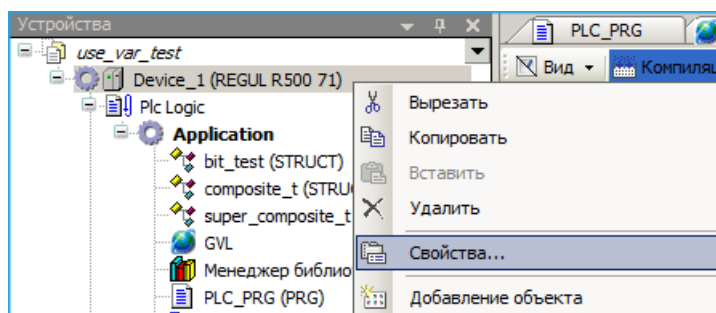


Рисунок 14 – Настройка проекта

В диалоге нужно открыть вкладку **Опции** и установить флажок в поле **Синхронизация переменных доступа и МЭК-задач** (Рисунок 15).

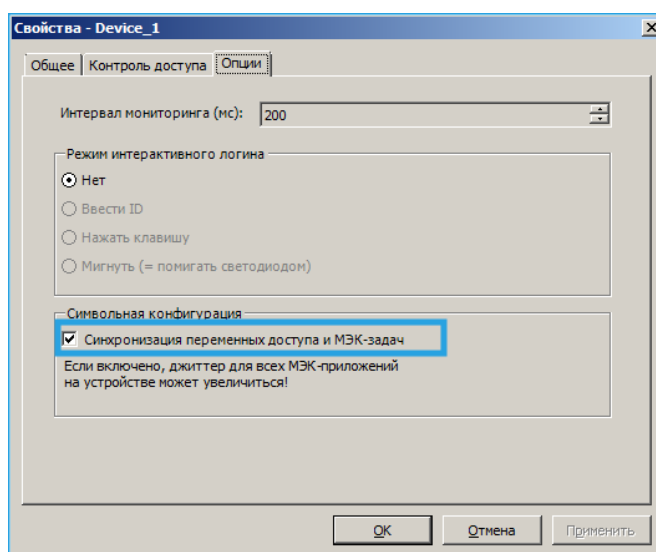


Рисунок 15 – Включение синхронизации

- Для приложений, включающих объект символьной конфигурации, поддержку синхронизации можно, дополнительно к первому способу, включить через свойства самой символьной конфигурации. Для этого после двойного клика на объекте символьной конфигурации в открывшейся вкладке нужно выбрать в меню **Установки** ⇒ **Настроить синхронизацию с МЭК-задачами...** (Рисунок 16).

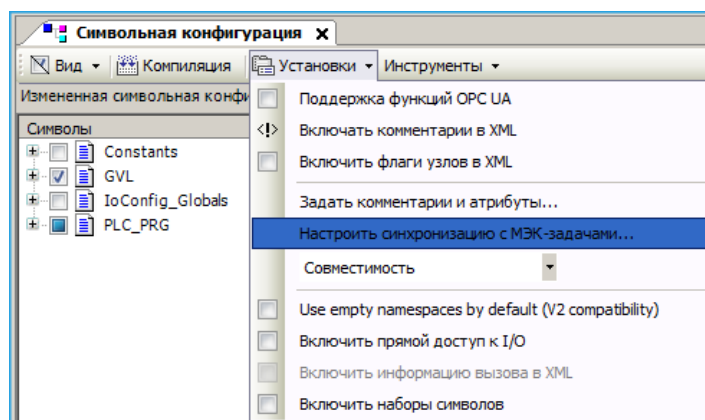


Рисунок 16 - Настройка синхронизации

В открывшемся диалоге свойств объекта контроллера (аналогично п.1) установите опцию **Синхронизация переменных доступа и МЭК-задач**.

После изменения настройки необходимо с помощью загрузки обновить все приложения на ПЛК, включая их загрузочные проекты.

Синхронизация операций чтения/записи приводит к задержке запуска ИЕС задач и увеличению джиттера. Чем больше переменных обрабатывается с синхронизацией, тем большее влияние оказывается на планирование выполнения ИЕС-задач. Задержке подвергаются все приложения среды исполнения, независимо от того, имеют они символьную конфигурацию или нет, принадлежат к разным проектам или к одному. Сама синхронизация возможна только при поддержке этого режима всеми приложениями, загруженными в ПЛК

С другой стороны, выполнение операций чтения/записи также зависит от выполняемых ИЕС задач - при большой нагрузке на процессор ПЛК (~100%) и/или длительности выполнения задач в интервале нескольких сот миллисекунд операция чтения/записи может завершиться с ошибкой по таймауту.

Для минимизации негативных эффектов следует придерживаться следующих правил:

- использовать синхронизацию только для тех переменных, для которых это необходимо;
- разделять переменные с синхронизацией и без в отдельные наборы данных;
- разбивать большие списки переменных с синхронизацией на несколько небольших;
- увеличивать период чтения циклических переменных.

ОБРАЩЕНИЕ В СЛУЖБУ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ

Для обращения в техническую поддержку Пользователю необходимо сформировать запрос на сайте технической поддержки: <https://support.prosoftsystems.ru>, либо отправить письмо по электронной почте: tp@prosoftsystems.ru. В первом случае требуется предварительная регистрация.

Обращение обязательно должно содержать следующие сведения:

- подробное описание сложившейся ситуации;
- наименование объекта и его месторасположение;
- наименование системы автоматизации;
- модель ПЛК;
- серийный номер ПЛК;
- версия среды разработки Epsilon LD;
- версия СПО-контроллера;
- версия OPC DA Servera;
- архив с лог-файлами (см. раздел «Журнал событий»);
- архив с лог-файлами, включающими в себя период времени, когда произошел отказ;
- дата и время возникновения отказа. А также периодичность и устойчивость повторения подобных отказов в случае, если такая информация имеется.

Желательно прислать проект для Epsilon LD, так как это может значительно упростить и ускорить процесс поиска причины отказа.

Для того, чтобы узнать, как получить необходимую информацию (сведений о версии Epsilon LD, версии СПО и так далее), ознакомьтесь с содержимым документа «Epsilon LD User Guide DPA 302».

ПРИЛОЖЕНИЕ А. АЛГОРИТМ ВЫБОРА АКТИВНОГО КОНТРОЛЛЕРА

При настроенном резервировании OPC сервер ориентируется на переменные приложения контроллера. Эти переменные определяют контроллер, активный на данный момент (параметр п.п.2.6 – `plc_main_check_tag = Application.GVL.CPU_Active`) и контроллер по умолчанию (параметр п.п.2.7 – `plc_main_check_tag2= Application.GVL.CPU_Default`).

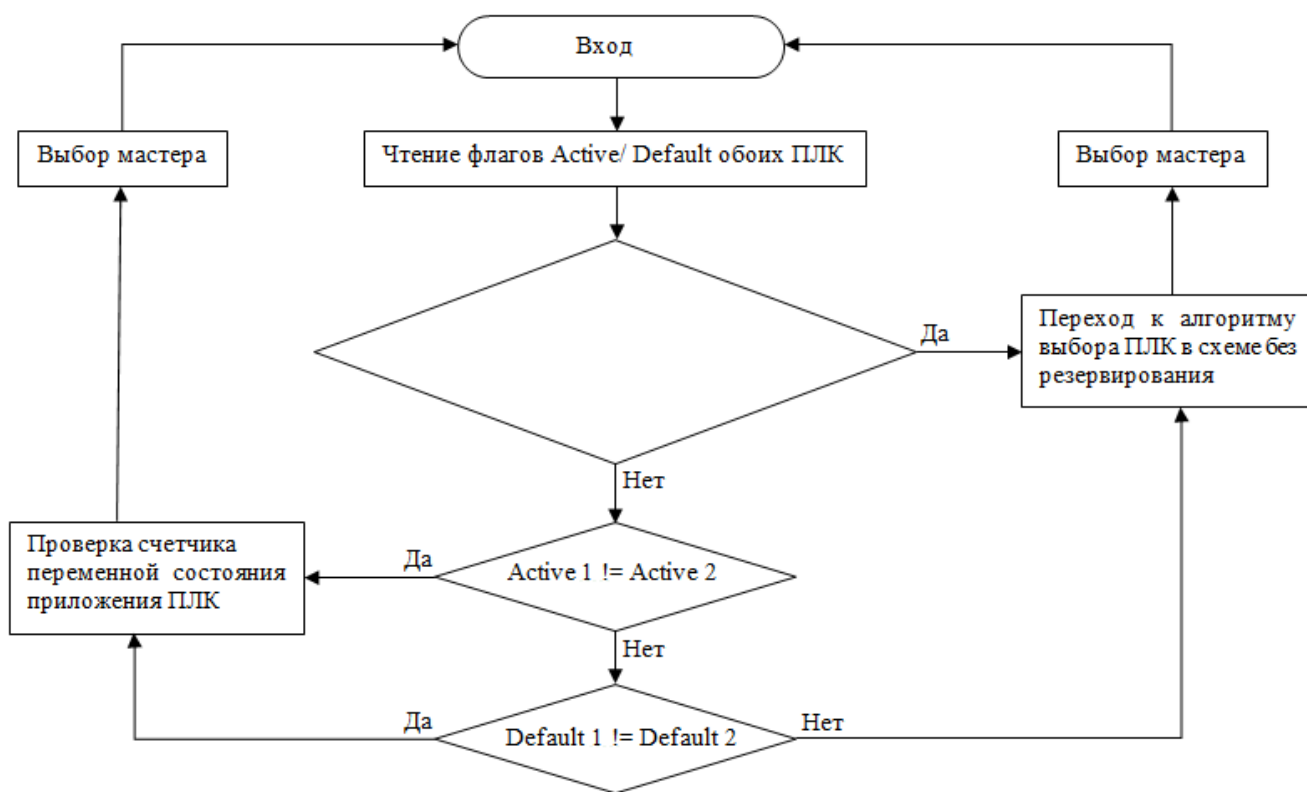


Рисунок 17 - Алгоритм выбора активного контроллера

Иначе выбор происходит согласно алгоритму без резервирования. При этом анализируется:

- наличие подключения по четырем каналам к двум контроллерам;
- настройки параметров согласно п.п.2.5, 2.8, 2.9 (см. таблицу В.1);
- запись в тег выбора активного контроллера `Root.State.set_active_plc`.

Если выполнена настройка параметра согласно п.п.2.20 (см. таблицу В.1), то на последнем шаге после выбора активного контроллера дополнительно происходит проверка счетчика переменной отслеживания активности приложения/среды выполнения для этого контроллера и окончательный выбор, какой контроллер считать активным.

ПРИЛОЖЕНИЕ Б. ОГРАНИЧЕНИЕ ПРОПУСКНОЙ СПОСОБНОСТИ

При начальном подключении OPC сервера к контроллеру и первом вычитывании переменных по каждому из каналов создается высокая нагрузка на процессор, что может привести к потере связи (из-за таймаута ожидания ответа от ПЛК) по остальным каналам подключения (активным или нет) с соседних АРМов и нежелательной многократной смене номера активного канала (и, в конечном счете, контроллера).

Стабильность подключений с нескольких АРМов к одному ПЛК достигается путем увеличения таймаута обмена данными между OPC сервером и ПЛК (согласно п.п.2.22, см. таблицу В.1).



ВНИМАНИЕ!

Минусом увеличения значения таймаута обмена является увеличение времени обнаружения обрыва связи по активному каналу – оно составляет удвоенное значение таймаута обмена

Для примера (конкретные временные задержки зависят от конфигурации АРМа!), при фиксированной «нулевой» загрузке процессора ПЛК Regul R500 (пустой проект) и заданном значении таймаута обмена данными 1000 мс получаем следующие значения в таблице Б.1 (под клиентом подразумевается один канал связи между OPC сервером и ПЛК).

Таблица Б.1 – Зависимость временных и нагрузочных характеристик от конфигурации

Кол-во тегов/кол-во подключений	1000	2000	5000	10000
Время первого цикла опроса при одновременном подключении к ПЛК, мс				
1 клиент	~70	~130	~330	~720
2 клиента	~80/160	~150/200	~400/460	~900/1200
3 клиента	~110/160/200	~200/300/450	~550/800/1100	~1100/2500/3000
5 клиентов	~170/210/250/320/540	~350/520/610/720/1000	610/1200/1600/2100/2400	1400/2200/3800/4100/5200
Время последующих циклов опроса, мс				
1 клиент	~12	~23	~60	~115
2 клиента	~12	~24-25	~60	~130
3 клиента	~12	~22-24	~63	~160
5 клиентов	~12	~22-24	~70	~160-400
Процент загрузки центрального процессора ПЛК				
1 клиент	~2%	4-5%	10-11%	20-21%
2 клиента	~4%	8-9%	21%	42%

Кол-во тегов/кол-во подключений	1000	2000	5000	10000
3 клиента	~7%	12-13%	32-33%	62-64%
5 клиентов	10-11%	22-23%	53-54%	80-82%

Как следует из таблицы:

- в наихудшем варианте – при одновременном подключении нескольких OPC серверов к одному контроллеру – при росте числа тегов увеличивается время первоначального считывания и уменьшается стабильность последующих подключений;
- если подключению не хватает выставленного значения таймаута в 1000 мс - требуется либо увеличить таймаут, либо подключения установятся при следующих попытках с тем же таймаутом в 1000 мс, когда первые подключения перейдут к циклическому чтению и нагрузка на процессор и среду исполнения снизится;
- при разнесенных по времени нескольких подключениях OPC серверов к контроллеру, ошибок ожидания таймаута при подключении и первом цикле чтения возникать не должно; но создаваемая при этом нагрузка может привести к таймауту обмена в «соседнем» канале, который уже перешел на циклическое чтение и для него задан небольшой таймаут обмена; в этом случае значение таймаута следует увеличить;
- время циклического чтения почти не зависит от количества установленных подключений и растет с увеличением числа тегов;
- процент загрузки процессора контроллера зависит и от числа тегов, и от числа подключений.

В рамках работы одного OPC сервера, подключения к одному контроллеру по двум интерфейсам синхронизируются и разнесены по времени. Синхронизации подключений между OPC серверами, запущенными на разных АРМах, нет.

При увеличении стартовой загрузки контроллера, все времена вычитывания (и первоначальное, и последующие) увеличиваются, соответственно, падает стабильность подключения каналов, что требует соответствующего увеличения значения таймаута обмена.

ПРИЛОЖЕНИЕ В. ОПИСАНИЕ ФАЙЛА КОНФИГУРАЦИИ

Таблица В.1 - Параметры конфигурации

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
1	Секция [options]		
1.1		log = 1	Включение (1)/ выключение (0) файлового журнала работы программы
1.2		logname = filelog.log	Базовое имя для лог-файлов и имя для текущего лог-файла. Более старые лог-файлы нумеруются с нарастающей нумерацией от 001 до XXX
1.3		logcount = 10	Максимальное количество лог-файлов N. Самый старый лог-файл с номером N+1 удаляется
1.4		logsize = 10.0	Максимальный размер каждого лог-файла в МБ
1.5		confirm_exit = 1	Если задано значение 1, то при ручном закрытии приложения будет дополнительно отображаться диалог-запрос подтверждения закрытия
1.6		exit_mode = 1	При заданном значении 0 закрытие приложения будет происходить мгновенно с аварийным закрытием главного процесса. Иначе, выгрузка проводится в штатном режиме с ожиданием всех потоков и освобождением ресурсов
1.7	МАХ кол-во значений по подписке в одной транзакции:	max_opc_tags_to_send = 5000	Максимальное количество OPC-тегов, отдаваемых OPC-клиенту по подписке в рамках одной транзакции
1.8	Закрывать OPC-сервер при отключении последнего OPC-клиента	close_on_last_opc_client = 1	Если задано значение 1, то приложение будет автоматически выгружено при отключении последнего OPC-клиента (кроме случаев, когда OPC сервер изначально был запущен вручную)

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
1.9	МАХ кол-во ошибок отправки данных по подписке:	opc_client_max_errors = 10	Максимальное количество ошибок отправки данных OPC-клиенту по подписке (OnDataChange), после которого данная OPC-группа деактивируется в предположении, что OPC-клиент нештатно разорвал подключение
1.10	Игнорировать изменение ТОЛЬКО временной метки значения тега	update_tags_if_only_timestamp_changed = 0	Если задано значение 0, то значения тегов будут обновляться при изменении значения (value) или качества (quality) переменной контроллера, изменения только временной метки (timestamp) игнорируются. При установке значения 1 теги будут обновляться даже если изменилась только временная метка (синхронный режим)
1.11 и 1.12	При запуске приложения: - Показать окно	on_startup_hide_in_taskbar = 0 on_startup_hide_in_tray = 0	Если не заданы значения (= 0), то при старте будет показано окно приложения
1.11	- Свернуть окно в панель задач	on_startup_hide_in_tray =	Если задано значение 1, то при старте окно приложения будет свернуто в панель управления
1.12	- Свернуть окно в системный трей	on_startup_hide_in_tray =	Если задано значение 1, то при старте окно приложения будет свернуто в системный лоток (tray)
1.13	Не создавать и не обновлять дерево переменных ПЛК	dont_create_params_tree = 0	Если задано значение 1, то, после подключения к ПЛК и получения его символьной конфигурации, дерево переменных в окне приложения отображаться не будет
2	Секция [plc]		
2.1	Главный ПЛК: IP адрес [#1] IP адрес [#2]	plc1_port1 = xxx.xxx.xxx.xxx plc1_port2 = xxx.xxx.xxx.xxx	IP-адреса каналов подключения к двум контроллерам (plc1 и plc2) по двум интерфейсам (port1 и port2) для каждого ПЛК; предполагается работа приложения либо с двумя ПЛК в резерве, либо с двумя ПЛК, на которых загружен один и тот же проект. Настройка подключения к ПЛК с разными проектами недопустима
	Резервный ПЛК: IP адрес [#1] IP адрес [#2]	plc2_port1 = xxx.xxx.xxx.xxx plc2_port2 = xxx.xxx.xxx.xxx	

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
2.2	Использовать учетные данные проекта Epsilon LD:	plc_use_user_login = 0	Подключение к контроллеру, для которого в Epsilon LD был заведен Online User. Если задано значение 1, то при подключении используется пара Login/Password из пп.2.3-2.4
2.3	Логин:	plc_user_login =	Имя пользователя (Login) для подключения
2.4	Пароль:	plc_user_password =	Пароль пользователя (Password) для подключения
2.5		plc_main = 1	Номер (1 или 2) ведущего контроллера на момент старта приложения, до момента вычитывания этой информации из самих контроллеров
2.6	Полное имя переменной 'Активный ПЛК' в проекте Epsilon LD	plc_main_check_tag = Application.GVL.CPU_Active	Полное имя переменной в проекте Epsilon LD, определяющей статус ведущего/резервного (True/False) контроллера, в случае наличия связи между контроллерами в схеме с резервированием. Может иметь произвольное название в проекте – оно же задается в этом параметре
2.7	Полное имя переменной 'ПЛК по умолчанию' в проекте Epsilon LD	plc_main_check_tag2= Application.GVL.CPU_Default	Полное имя переменной в проекте Epsilon LD, определяющей статус ведущего/резервного (True/False) контроллера по умолчанию при ошибке синхронизации между двумя контроллерами (в случае, когда у обоих флаг CPU_Active = True). Может иметь произвольное название в проекте – оно же задается в этом параметре
2.8	Выбор активного ПЛК через тег 'ROOT.State.set_active_plc'	plc_use_hard_arm_selection = 0	Если задано значение 1, то через тег Root.State.set_active_plc происходит безусловный выбор активного контроллера вне зависимости от наличия с ним связи – если связь с ним есть, то данные берутся из него, если связи нет, то все теги в общей группе PLC будут BAD и переключения на другой контроллер не произойдет; значение по умолчанию 0 – в этом случае OPC сервер пробует сделать активным контроллер, заданный в Root.State.set_active_plc, если же с ним связи нет, то активным будет второй контроллер (если он на

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
			связи), также поведение зависит от параметра пп.2.9
2.9	Сделать ПЛК активным при восстановлении связи	plc_set_active_on_link_restore = 0	<p>Если задано значение 1, то, после восстановления связи с контроллером, заданным в Root.State.set_active_plc, он снова станет активным; по умолчанию значение 0, при этом:</p> <p>при записи в Root.State.set_active_plc будет выполнен алгоритм как при plc_use_hard_arm_selection = 0 (см. пп.2.8) – активным будет либо заданный контроллер, либо второй (в зависимости от наличия связи);</p> <p>если через Root.State.set_active_plc был выбран ПЛК1, затем с ПЛК1 связь пропала, активным стал ПЛК2, затем связь с ПЛК1 восстановилась – активным останется ПЛК2</p>
2.10		plc_enable_app_state_check = 0	<p>Если задано значение 1, то OPC сервер дополнительно отслеживает состояние приложения в контроллере (RUN/STOP), которое может быть установлено в режиме отладки из среды Epsilon LD или с помощью переключателя на процессорном модуле.</p> <p>Рекомендуется использовать только в отладочных целях. На форме настроек в приложении значение инвертировано</p>
2.11		plc1_tags_prefixes= PLC1; TAG_PREFIX_2; TAG_PREFIX_3;...	<p>Префиксы в именах OPC-тегов, определяющих привязку OPC-группы этих тегов к конкретному контроллеру, в данном случае к первому в конфигурации. Префикс PLC1 задан по умолчанию и соответствует ветке тегов Root.PLC1</p>

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
2.12		plc2_tags_prefixes= PLC2; TAG_PREFIX_4; TAG_PREFIX_5;...	Префиксы в именах OPC-тегов, определяющих привязку OPC-группы этих тегов к конкретному контроллеру, в данном случае ко второму в конфигурации. Префикс PLC2 задан по умолчанию и соответствует ветке тегов Root.PLC2
2.13	Период чтения данных с контроллера [мс]	plc_params_check_period = 500	Период запроса (частота обновления) текущих значений всех переменных контроллера (точнее, подмножества переменных, на которые есть подписка от OPC-клиентов), в мс
2.14		plc_reconnect_timeout= 5000	Пауза, в мс, после обрыва связи с контроллером перед новой попыткой установления соединения
2.15		plc_link_status_check_period = 100	Период проверки статуса подключения проводных сетевых интерфейсов сервера (АРМа), на котором запущено приложение OPC сервера
2.16		plc_browse_enable_arrays = 0	Включение(1)/выключение(0) отображения в дереве переменных корневых узлов массивов
2.17		plc_browse_enable_structs = 0	Включение(1)/выключение(0) отображения в дереве переменных корневых узлов структур (сложных типов данных)
2.18		plc_logging = 0	Включение(1)/выключение(0) дополнительного логирования работы библиотеки API PLCHandler
2.19		plc_use_server_timestamps = 1	Если задано значение 1, то меткой времени значения переменной, считанного из контроллера, является локальное время сервера (АРМа), на котором запущен OPC сервер (с точностью до миллисекунд). Если задано значение 0, то метка времени предоставляется библиотекой PLCHandler (с точностью до секунд)

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
2.20	Контрольный счетчик: - полное имя переменной счетчика	plc_check_alive_counter_tag = Application.GVL.Check_alive_counter	Полное имя переменной (Check_alive_counter – указан как пример) проекта ПЛК, реализующей дополнительный механизм контроля состояния приложения/среды выполнения ПЛК. Переменная должна иметь тип UDINT и представляет собой простой счетчик, инкрементируемый в одной из задач приложения. Переменная должна быть добавлена в символьную конфигурацию проекта. При наличии подключения к контроллеру OPC сервер периодически вычитывает значение счетчика и, если его значение не меняется в течение периода, заданного в п.2.21, то считается, что приложение на данном контроллере находится в состоянии ошибки (исключения/останова)
2.21	Контрольный счетчик: - таймаут контроля счетчика [мс]	plc_check_alive_counter_to = 500	Таймаут проверки изменения переменной счетчика пп.2.20, в мс
2.22	Таймаут чтения данных из ПЛК [мс]	plc_data_exchange_timeout = 1000	Таймаут обмена данными между OPC сервером и ПЛК, в мс. Влияет на стабильность каналов подключения к ПЛК на чтение при запуске OPC сервера на нескольких АРМах
2.23		plc_write_data_exchange_timeout = 1000	Таймаут обмена данными между OPC сервером и ПЛК в мс. Для каналов подключения к ПЛК на запись
2.24		plc_opc_lock_timeout = 10	Таймаут потока слежения за блокировкой ресурсов в OPC интерфейсе. Отладочный параметр
2.25	При запуске загружать сохраненную символьную конфигурацию	plc_load_symbols_from_file = 1	Включает (1)/выключает (0) сохранение и загрузку символьной конфигурации проекта в файловый кэш. При старте приложения ускоряет загрузку информации о типах и атрибутах доступа переменных символьной конфигурации проекта

№ п/п	Параметры в приложении	Соответствующие параметры конфигурационного файла	Описание
2.26	Динамически отслеживать изменение символьной конфигурации	plc_check_symbols_change = 0	Включает (1)/выключает (0) проверку изменения символьной конфигурации проекта ПЛК при повторном установлении подключения в канале связи и при переключении на соседний канал связи
2.27	Синхронизация с ИЕС задачами для всех переменных	plc_using_iec_task_synch = 0	Включает (1)/выключает (0) синхронизацию операций чтения/записи для ВСЕХ переменных ПЛК

Перезапуск приложения OPC сервера потребуется при изменении следующих настроек:

- Главный ПЛК: IP адрес ...(п.п. 2.1);
- Полное имя переменной 'Активный ПЛК' в проекте Epsilon LD:...(п.п. 2.6);
- Полное имя переменной 'ПЛК по умолчанию' в проекте Epsilon LD:...(п.п. 2.7);
- Контрольный счетчик:...(п.п. 2.20, 2.21);
- Использовать учетные данные проекта Epsilon LD: Логин, Пароль...(п.п. 2.2, 2.3, 2.4);
- Не создавать и не обновлять дерево переменных ПЛК (п.п. 1.13).