2018

СПК

Настройка обмена с верхним уровнем

Руководство для начинающих пользователей

Версия: 1.1
Дата: 30.01.2018

Оглавление

1. Цель документа. Способы связи СПК с верхним уровнем

Этот документ представляет собой руководство по настройке обмена данными с верхним уровнем АСУ (SCADA-системами) для панельных контроллеров Овен СПК. Подразумевается, что читатель обладает базовыми навыками работы с **CODESYS** и **СПК**, поэтому общие вопросы (например, создание и загрузка проектов) в данном документе не рассматриваются; они подробно описаны в документах **СПК. Первый старт** и **СПК. FAQ**, которые доступны на сайте Овен в разделе **CODESYS V3/Документация**.

В документе рассмотрены следующие вопросы:

1. использование web-визуализации. Этот компонент **CODESYS 3.5** позволяет создавать экраны визуализации, с которыми можно будет работать на компьютере (планшете, телефоне) через web-браузер. При этом клиент визуализации и СПК должны находиться в одной сети (локальной или созданной с помощью VPN и др. технологий). Если пользователю не требуется «сложная» визуализация и обработка/хранение данных на ПК, то представляется разумным использовать web-визуализацию вместо SCADA-системы.

Преимущества:
- легкость настройки и простота использования;
- не требуется использования дополнительного ПО;
- бесплатность (* требуется контроллер с поддержкой web-визуализации);

Недостатки:
- дополнительная нагрузка на контроллер;
- возможности визуализации ограничены функционалом CODESYS;
- отсутствие возможности обрабатывать и хранить данные на ПК.

2. использование сетевых переменных. Этот компонент позволяет в несколько кликов настроить обмен данными между контроллерами, программируемыми в **CODESYS 3.5** и находящимися в одной локальной сети.

Преимущества:
- легкость настройки и простота использования;
- бесплатность (* требуется контроллер с Ethernet);

Недостатки:
- все контроллеры, участвующие в обмене, должны программироваться в **CODESYS 3.5**.

3. передача данных в SCADA-систему с помощью OPC-серверов. Использование OPC-сервера позволяет собирать данные с различных устройств по разным протоколам обмена, после чего передавать их в SCADA-систему для обработки и визуализации. В рамках документа рассмотрена связь СПК с четырьмя OPC-серверами: CODESYS OPC Server V3, Master OPC Universal Modbus Server, Lectus Modbus OPC/DDE Server и ОВЕН OPC (новый).

Преимущества:
- интегрированный комплекс ПО (**OPC + SCADA**) для сбора данных с различных устройств с последующим отображением, обработкой и архивированием;
- поддержка различных протоколов, наличие готовых конфигураций приборов.

Недостатки:
- в большинстве случаев требуется приобретение соответствующего ПО;
- сложность настройки;
- требуется наличие АРМ.

4. передача данных в облачный сервис OwenCloud. Облачный сервис применяется для удаленного мониторинга, управления и хранения архивов данных приборов, используемых в системах автоматизации. Подключение приборов к сервису осуществляется по интерфейсам **RS-485** (с помощью специальных сетевых шлюзов) или **Ethernet** (в этом случае требуется подключение приборов к сети с доступом к Интернету).

Преимущества:
- легкость настройки и простота использования;
- основные функции бесплатны;
- доступ к данным из любой точки мира через web-интерфейс или мобильное приложение;
- рассылка аварийных сообщений и push-уведомлений.

Недостатки:
- функционал сервиса уступает возможностям SCADA-систем.

2. Web-визуализация

Web-визуализация – это компонент **CODESYS 3.5**, который позволяет просматривать экраны визуализации контроллера в web-браузере. ***Обратите внимание***, что контроллер должен поддерживать данный компонент (примерами таких контроллеров являются СПК207 и ПЛК323 с модификациями **WEB**), а web-браузер – поддерживать **HTML5**.

Рис. 2.1. Пример работы с web-визуализацией в браузере

Web-визуализация автоматически включается в проект при добавлении компонента **Менеджер визуализации** (или создания первого экрана визуализации):

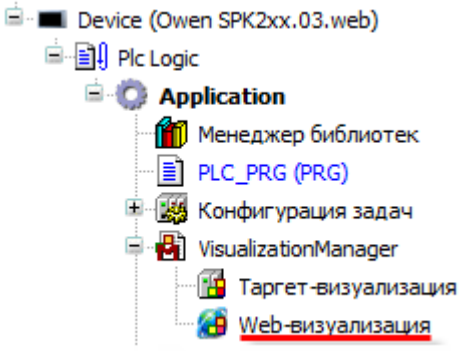


Рис. 2.2. Компонент web-визуализация в дереве проекта

При этом в проект может быть добавлено несколько экземпляров компонента (например, при необходимости создания нескольких веб-страниц).

Компонент обладает следующими настройками:

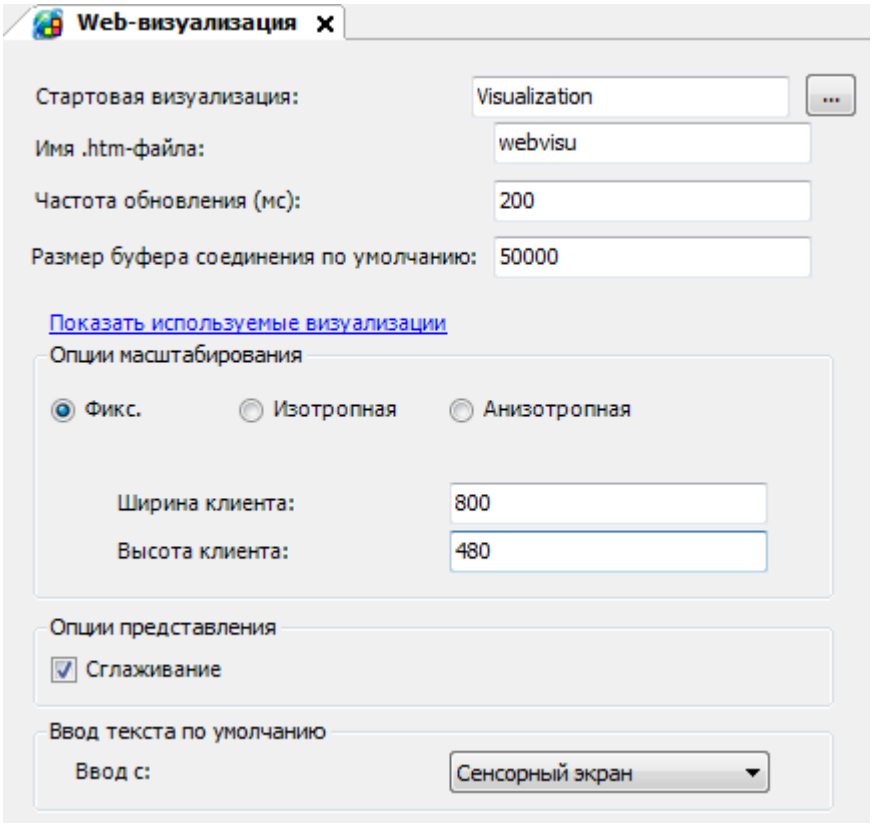


Рис. 2.3. Настройки компонента **Web-визуализация**

Стартовая визуализация – позволяет выбрать экран визуализации, который будет отображен на клиенте web-визуализации при открытии данной веб-страницы.

Имя .htm-файла – название страницы, которое указывается в web-адресе визуализации. По умолчанию имя страницы – **webvisu**, и адрес web-визуализации соответственно

http://<IP-адрес контроллера>:8080/webvisu.htm

Частота обновления – позволяет настроить частоту обновления экранов визуализации на клиенте web-визуализации. Значение задается в миллисекундах.

Размер буфера соединения по умолчанию – максимальный размер буфера данных (в байтах), передаваемых клиенту web-визуализации. Рекомендуется не изменять значение этого параметра.

Подгонка размера – позволяет выбрать режим масштабирования экранов визуализации на клиенте web-визуализации:

Fixed – позволяет задать фиксированный размер визуализации в пикселях;

Isotropic – экран визуализации будет ***масштабироваться*** до размеров дисплея клиента web-визуализации ***с сохранением*** соотношения сторон;

Anisotropic – экран визуализации будет ***масштабироваться*** до размеров дисплея клиента web-визуализации ***без сохранения*** соотношения сторон.

Сглаживание – позволяет включить сглаживание элементов. Это улучшает их внешний вид, но может привести к падению производительности и нежелательным графическим артефактам (например, у всплывающих подсказок и невидимых элементов).

Ввод текста по умолчанию – позволяет определить основное устройство ввода для клиента web-визуализации – сенсорный экран или клавиатуру.

Подробная информация о разработке экранов визуализации для контроллеров СПК приведена в документе **СПК. Визуализация**, доступном на диске с ПО из комплекта поставки, а также на сайте компании Овен в разделе **CODESYS V3/Документация**.

3. Сетевые переменные

3.1. Основные сведения о сетевых переменных

Сетевые переменные позволяют организовать обмен между несколькими контроллерами, программируемыми в **CODESYS 3.5**, по протоколу UDP, который работает поверх **Ethernet**. Соответственно, все контроллеры, участвующие в обмене, должны находиться в одной локальной сети. В настройках сетевого оборудования должна быть отключена блокировка UDP-пакетов.

Очевидно, что у пользователя есть возможность организовать обмен по **Modbus TCP**; в этом случае ему придется добавлять в проект соответствующие компоненты (Ethernet, Modbus TCP Master, Modbus TCP Slave), настраивать их, разбираться в используемых функциях и адресации регистров. Преимуществом использования сетевых переменных является простота их настройки – достаточно создать на одном устройстве список читаемых/записываемых переменных и импортировать его на другом. В то же время, протокол **UDP** по сравнению с **TCP** обладает рядом недостатков (см., например, соответствующую статью на Wikipedia). Надо отметить, что часть этих недостатков может быть компенсирована настройками **CODESYS** (контроль CRC, подтверждение получения).

В рамках каждого списка сетевых переменных, обмен происходит только в одном направлении. Иными словами, у любого списка есть устройство-отправитель (в каждый момент времени – только одно) и устройства-получатели (их может быть несколько). При этом каждое устройство может содержать несколько списков отправляемых и несколько списков получаемых сетевых переменных.

Связь между устройством-отправителем и устройством-получателем определяется тремя параметрами:

1. **Порт**, через который осуществляется передача UDP-пакетов;

2. **Адрес рассылки** – пул адресов, на которые отправляются UDP-пакеты;

3. **Идентификатор списка** – номер используемого списка сетевых переменных.

Эти параметры должны быть идентичными для отправителя и всех получателей. При этом каждый из списков переменных устройства должен иметь уникальный идентификатор и номер порта.

Структурная схема обмена сетевыми переменными между двумя контроллерами СПК приведена на рис. 3.1. В данном случае каждое устройство является как отправителем, так и получателем.

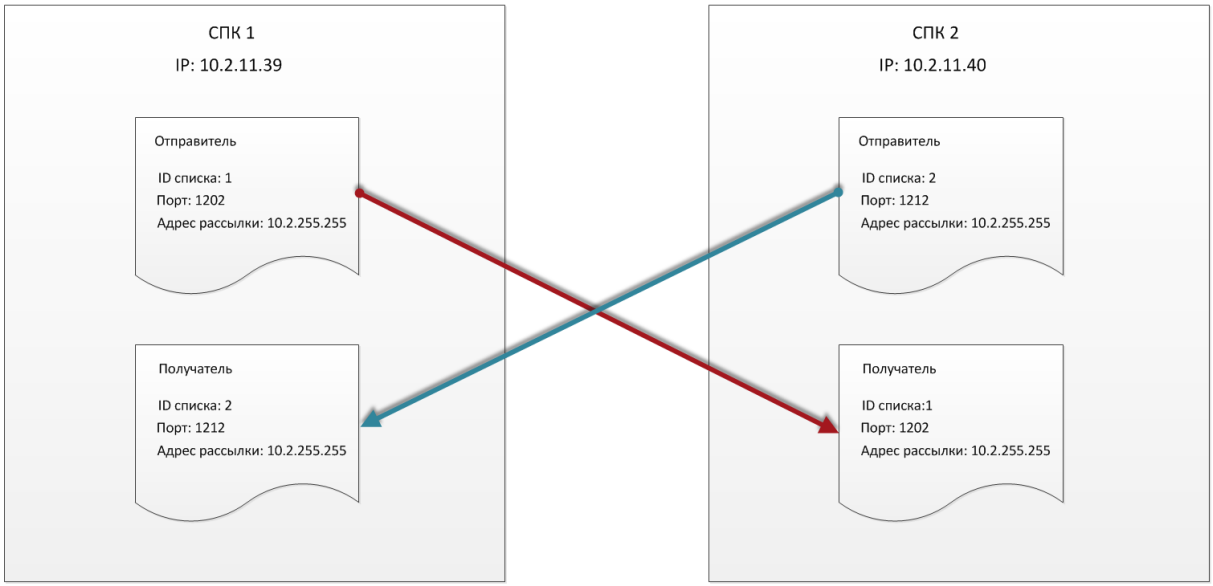


Рис. 3.1. Структурная схема обмена сетевыми переменными между контроллерами СПК

При масштабировании системы от пользователя потребуется только добавить на новые устройства соответствующие списки. Например, если в схеме, приведенной на рис 3.1, потребуется установка СПК 3, который должен будет получать сетевые переменные от СПК 1, то достаточно будет в проект для СПК 3 добавить список получаемых сетевых переменных из проекта СПК 2.

CODESYS 3.5 позволяет в пределах одного проекта создавать программы сразу для нескольких контроллеров, что также упрощает процесс разработки.

В п. 3.2 и п. 3.3 рассмотрены настройки компонентов Список сетевых переменных (отправитель) и Список сетевых переменных (получатель).

В п. 3.4 рассмотрен пример обмена сетевыми переменными между двумя контроллерами СПК согласно рис. 3.1.

3.2. Добавление и настройка компонента Список сетевых переменных (отправитель)

Для добавления в проект компонента **Список сетевых переменных (отправитель)** необходимо в дереве проекта нажать **ПКМ** на узел **Application** и в контекстном меню выбрать команду **Добавление объекта**:

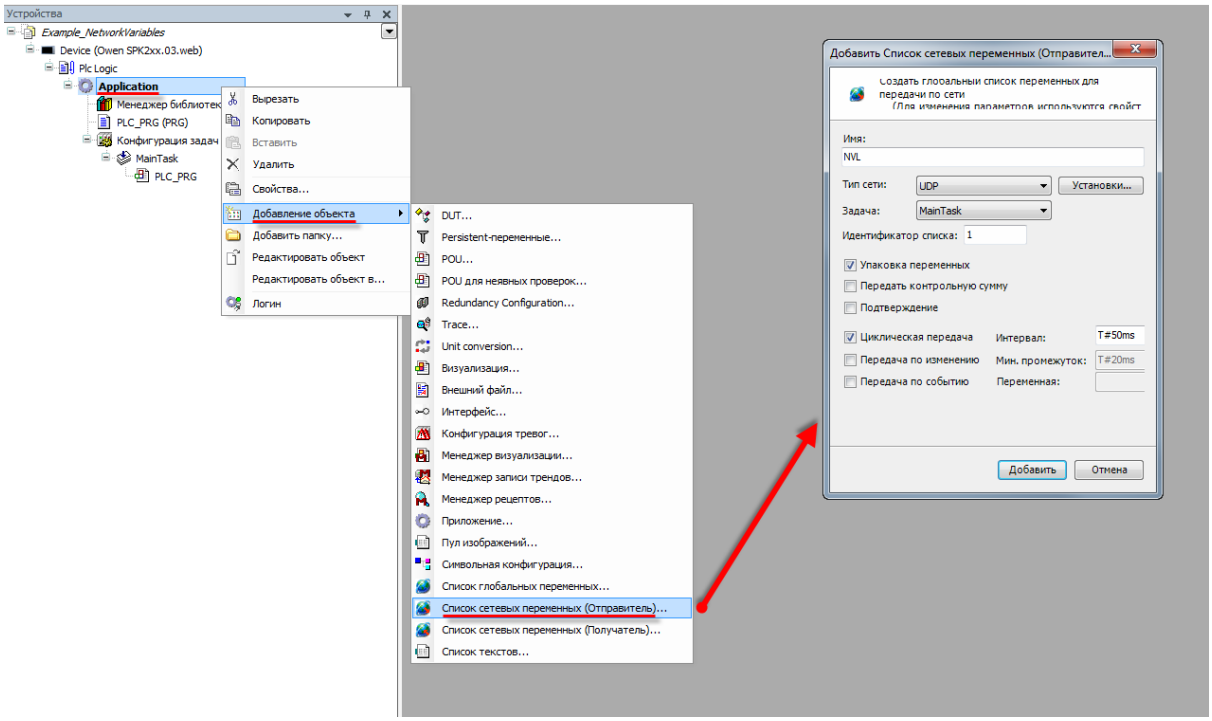


Рис. 3.2. Добавление компонента **Список сетевых переменных (отправитель)**

После создания списка в проект будет автоматически добавлена библиотека **NetVarUdp**:

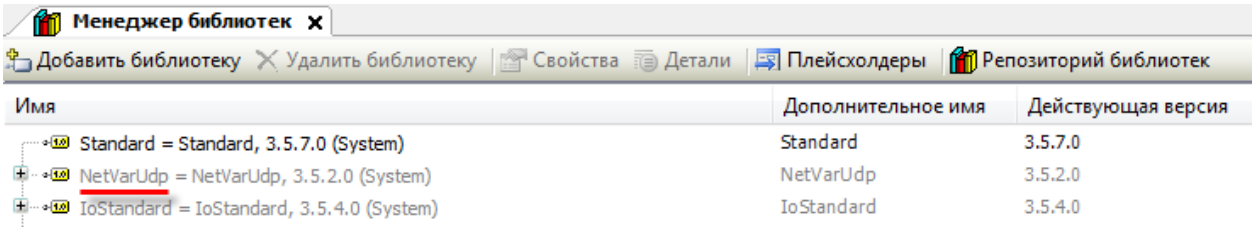


Рис. 3.3. Библиотека **NetVarUdp** в **Менеджере библиотек**

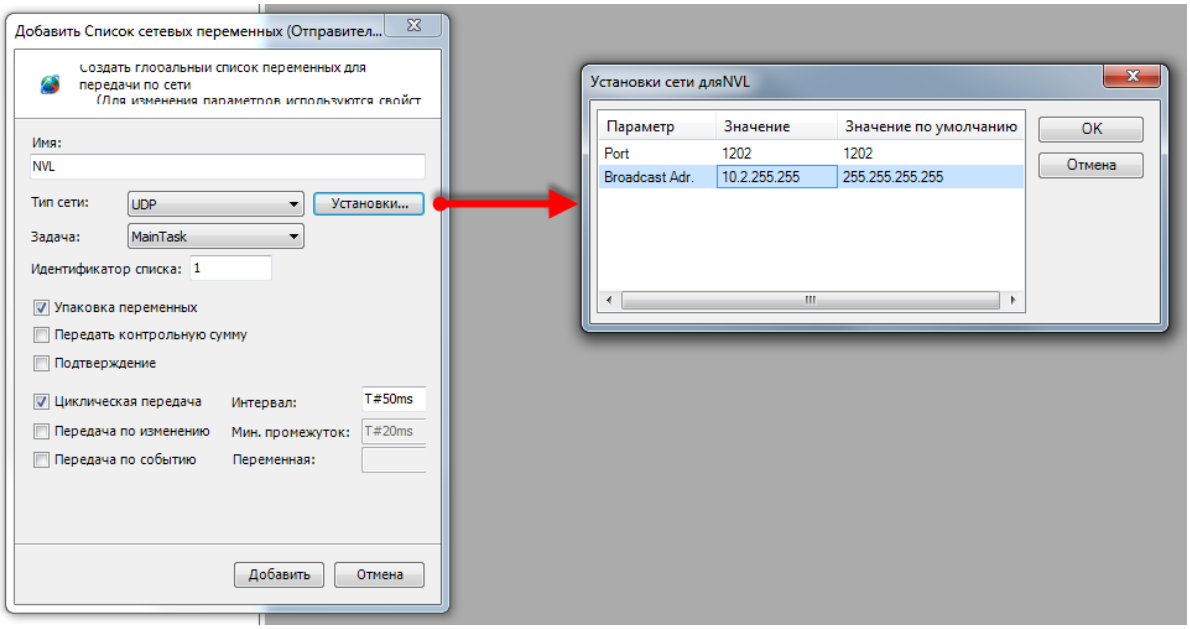


Рис. 3.4. Настройки компонента **Список сетевых переменных (отправитель)**

1. **Тип сети** – протокол, используемый для передачи сетевых переменных. В данный момент поддерживается только протокол **UDP**.

2. **Установки** – в данном меню выбирается **порт** контроллера и **адрес широковещательной рассылки** (Broadcast address).

Обратите внимание, что в пределах одного устройства для каждого списка сетевых переменных (как отправляемых, так и получаемых), должен использоваться уникальный номер порта. В качестве номера порта можно, например, использовать любое число из диапазона 1200-1299, за исключением 1217 (поскольку порт **1217** используется для связи контроллера и **CODESYS**).

Обратите внимание, что адрес рассылки должен соответствовать локальной сети. Например, если СПК имеет IP-адрес **10.2.11.10**, то адрес рассылки должен быть задан как **10.2.255.255**. В данном случае, получателем сетевых переменных может являться любое устройство с IP-адресом **10.2.x.x**.

При использовании значения по умолчанию (**255.255.255.255**) обмен сетевыми переменными будет невозможен.

3. **Задача** – задача, к которой привязан процесс обмена сетевыми переменными. Рекомендуется выбирать задачу с наименьшим временем цикла.

4. **Идентификатор списка** – номер данного списка. ***Обратите внимание***, что в пределах одного устройства для каждого списка сетевых переменных (как отправляемых, так и получаемых), должен использоваться уникальный идентификатор.

5. **Упаковка переменных** – при наличии галочки, переменные будут упаковываться в пакеты (датаграммы), размер которых будет определяться настройками сети. При отсутствии галочки, каждая переменная отправляется отдельным пакетом.

6. **Передавать контрольную сумму** – при наличии галочки, в пакет будет добавлена контрольная сумма. Устройство-получатель будет отбрасывать пакеты с несовпадающей контрольной суммой.

7. **Подтверждение передачи** – при наличии галочки, отправитель будет ждать подтверждения получения на каждый отправленный пакет. При отсутствии подтверждения будет выставлен соответствующий флаг в переменных диагностики.

Далее пользователь должен выбрать один из режимов передачи сетевых переменных:

8. **Циклическая передача** – в этом режиме сетевые переменные будут передаваться с заданным интервалом времени.

9. **Передача по изменению** – в этом режиме сетевые переменные будут передаваться при изменении их значений, при этом пользователь выбирает минимальный интервал времени между двумя передачами (т.е. если в пределах этого интервала значение переменной изменилось, то она все равно не будет отправлена до его истечения).

10. **Передача по событию** – в этом режиме сетевые переменные будут передаваться по переднему фронту заданной логической переменной.

Обратите внимание, что при загрузке контроллера происходит автоматическая отправка сетевых переменных вне зависимости от выполнения условий из пп. 8-10.

После создания списка, необходимо наполнить его нужными переменными:

NVL

```
1  VAR_GLOBAL
2      wVar1: WORD;
3      rVar1: REAL;
4      aValues: ARRAY [0..1] OF INT;
5      Struct1: MyStruct;
6  END_VAR
```

Рис. 3.5. Заполнение списка сетевых переменных

Если необходимо изменить настройки созданного списка, то следует нажать на нем **ПКМ** и в контекстном меню выбрать пункт **Свойства**, после чего перейти на вкладку **Свойства сети**.

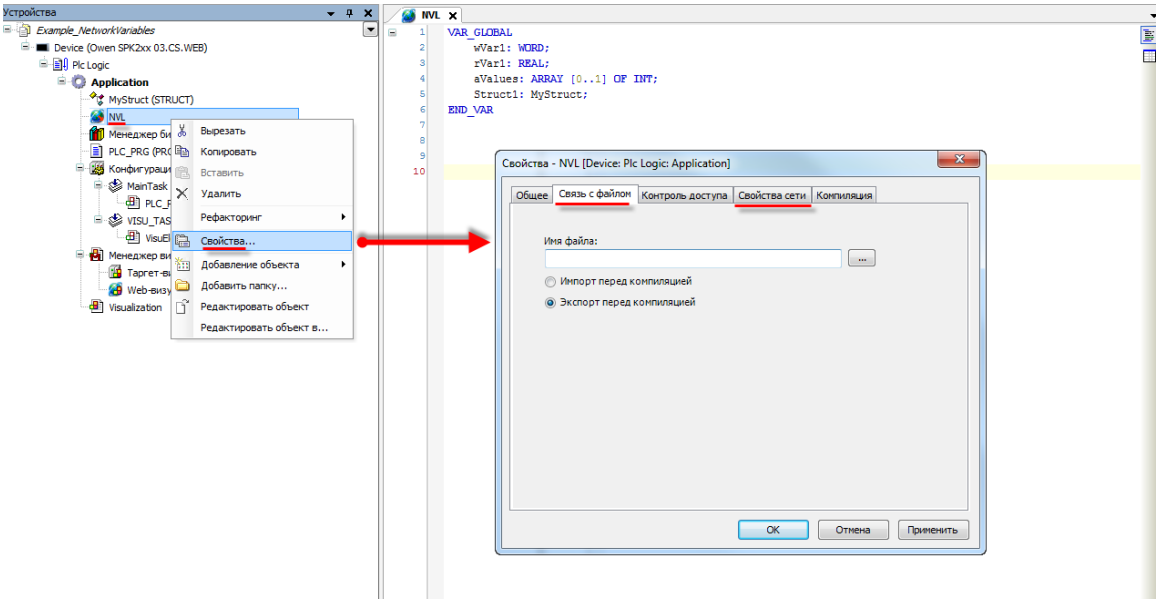


Рис. 3.6. Изменение настроек списка сетевых переменных

На вкладке **Связь с файлом** можно указать путь к файлу, в который будет экспортирован (или из которого будет импортирован) список глобальных переменных. Экспорт/импорт происходит после компиляции проекта. Экспортированный список можно импортировать в компонент **Список сетевых переменных (получатель)** другого контроллера.

Экспортированный список представляет собой файл формата **.gvl**, который содержит сетевые переменные и сетевые настройки. Его можно открыть любым текстовым редактором:

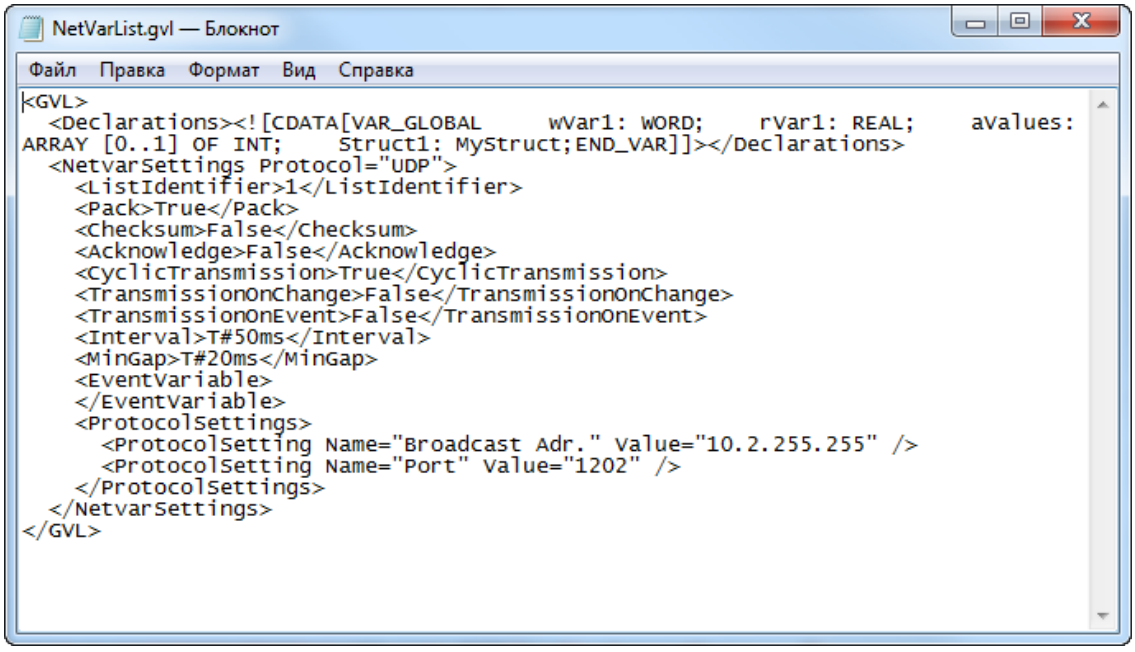


Рис. 3.7. Содержимое файла формата .**gvl**

3.3. Добавление и настройка компонента Список сетевых переменных (получатель)

Для добавления в проект компонента **Список сетевых переменных (получатель)** необходимо в дереве проекта нажать **ПКМ** на узел **Application** и в контекстном меню выбрать команду **Добавление объекта**:

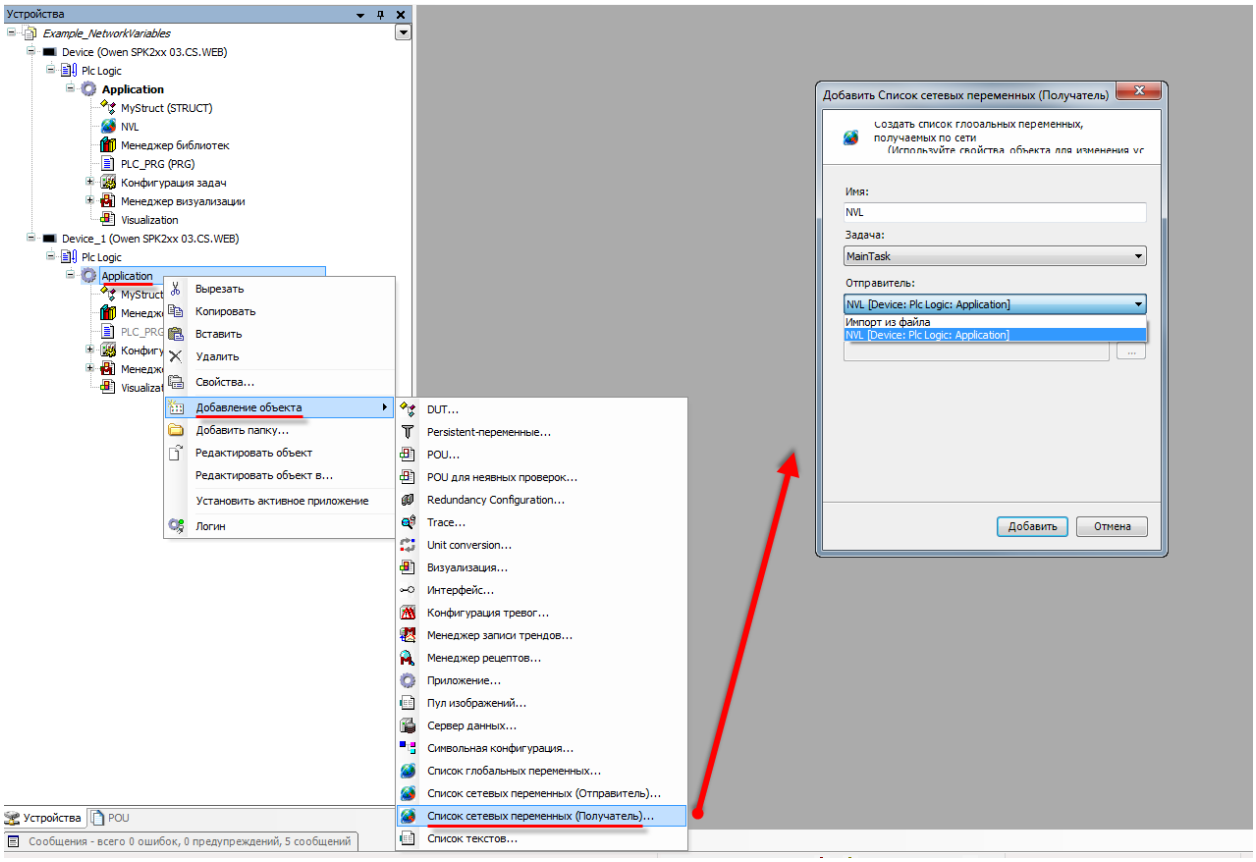


Рис. 3.8. Добавление компонента **Список сетевых переменных (получатель)**

При добавлении компонента пользователю нужно указать, откуда будет импортирован список сетевых переменных, созданный на устройстве-отправителе – из другого устройства проекта или же из файла формата **.gvl** (см. рис. 3.7.)

После этого список отправителя (включая все сетевые настройки) будет импортирован на устройство-получатель. Никаких дополнительных настроек не требуется.

3.4. Пример работы с сетевыми переменными

Рассмотрим пример обмена сетевыми переменными между двумя контроллерами СПК, находящимися в одной локальной сети.

Структурная схема примера приведена на рис. 3.9:

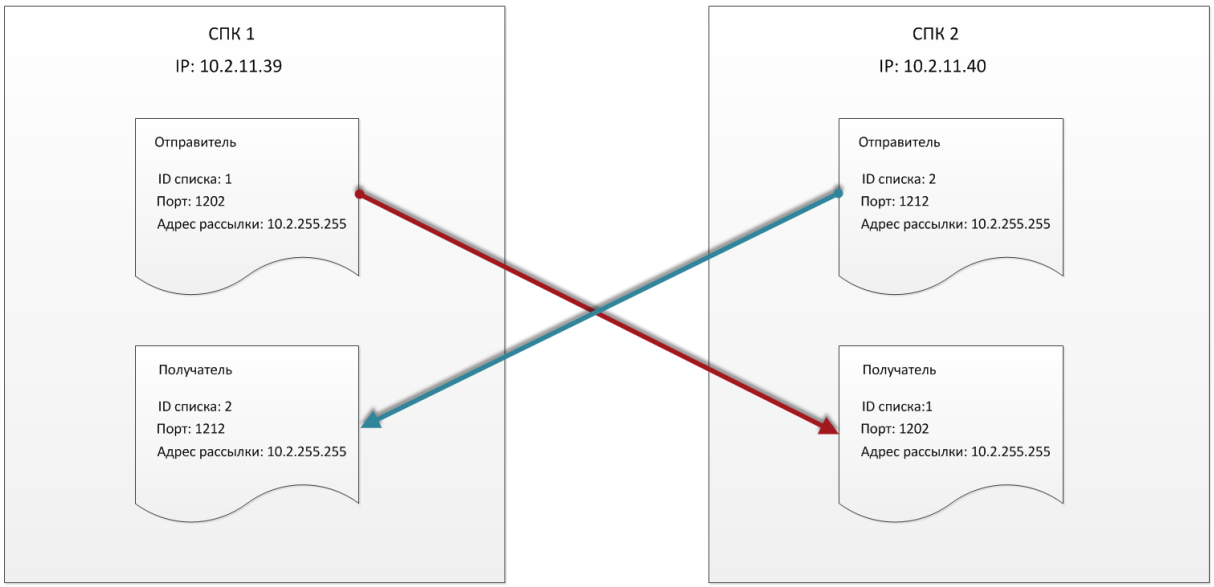


Рис. 3.9. Структурная схема примера **Сетевые переменные**

Пример создан в среде **CODESYS 3.5 SP7 Patch4** и подразумевает запуск на **СПК207.03.CS(-WEB)** с таргет-файлом **3.5.4.20 (023)**.

Пример доступен для скачивания: Example_NetworkVariables.projectarchive

Сетевые параметры и используемые переменные приведены в табл. 3.1.

Табл. 3.1. Сетевые параметры и переменные примера **Сетевые переменные**

Параметр	**СПК 1**	**СПК 2**
IP-адрес	10.2.11.39	10.2.11.40
Порт UDP	1202 (отправление) 1212 (получение)	1212 (отправление) 1202 (получение)
Broadcast адрес	10.2.255.255	
Идентификатор списка	1 (отправление) 2 (получение)	2 (отправление) 1 (получение)
Отправляемая сетевая переменная	wVar12	wVar21
Получаемая сетевая переменная	wVar21	wVar12

1. Создайте новый проект **CODESYS** для **СПК207.03** (язык программы не имеет значения, поскольку проект не будет содержать программы).

2. Переименуйте узел **Device** в **SPK1** (поскольку позже мы добавим в проект еще один контроллер, изменение имени упростит понимание их взаимодействия):

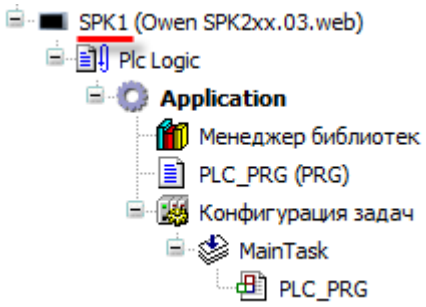


Рис. 3.10. Переименование узла **Device** в **SPK1**

3. Добавьте компонент Список сетевых переменных (отправитель) с названием **From_SPK1_To_SPK2**. Сетевые настройки будут соответствовать табл. 3.1:

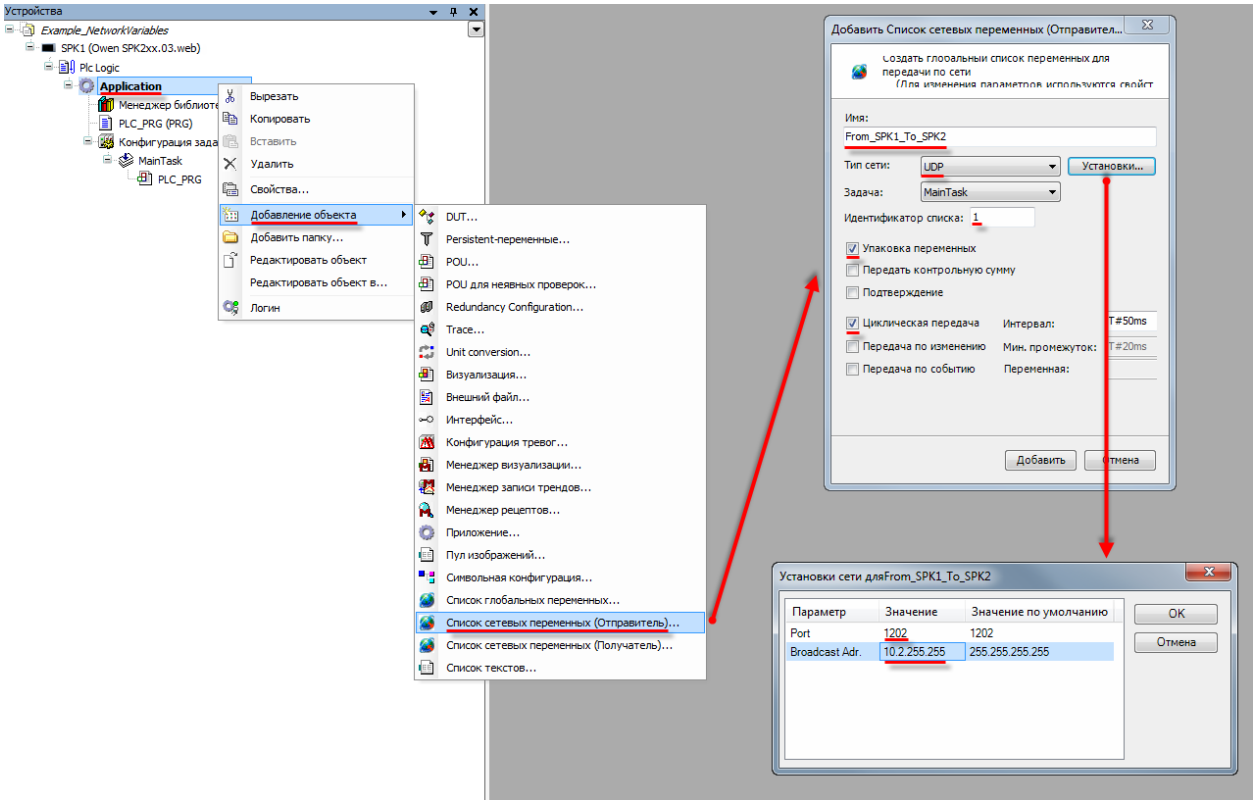


Рис. 3.11. Настройки компонента **Список сетевых переменных (отправитель SPK1)**

4. В созданном списке объявите переменную **wVar12** типа **WORD**. Значение этой переменной будет передаваться в **СПК2** каждые 50 мс.

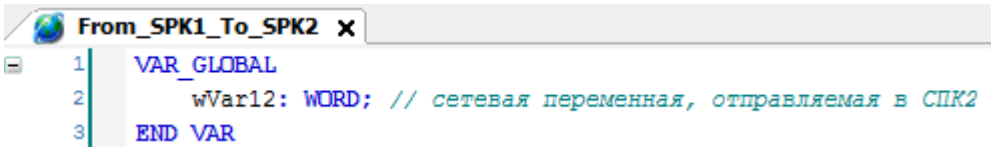


Рис. 3.12. Объявление сетевой переменной (**SPK1**)

5. В проект **CODESYS**, содержащий **SPK1**, добавьте контроллер **SPK2**.

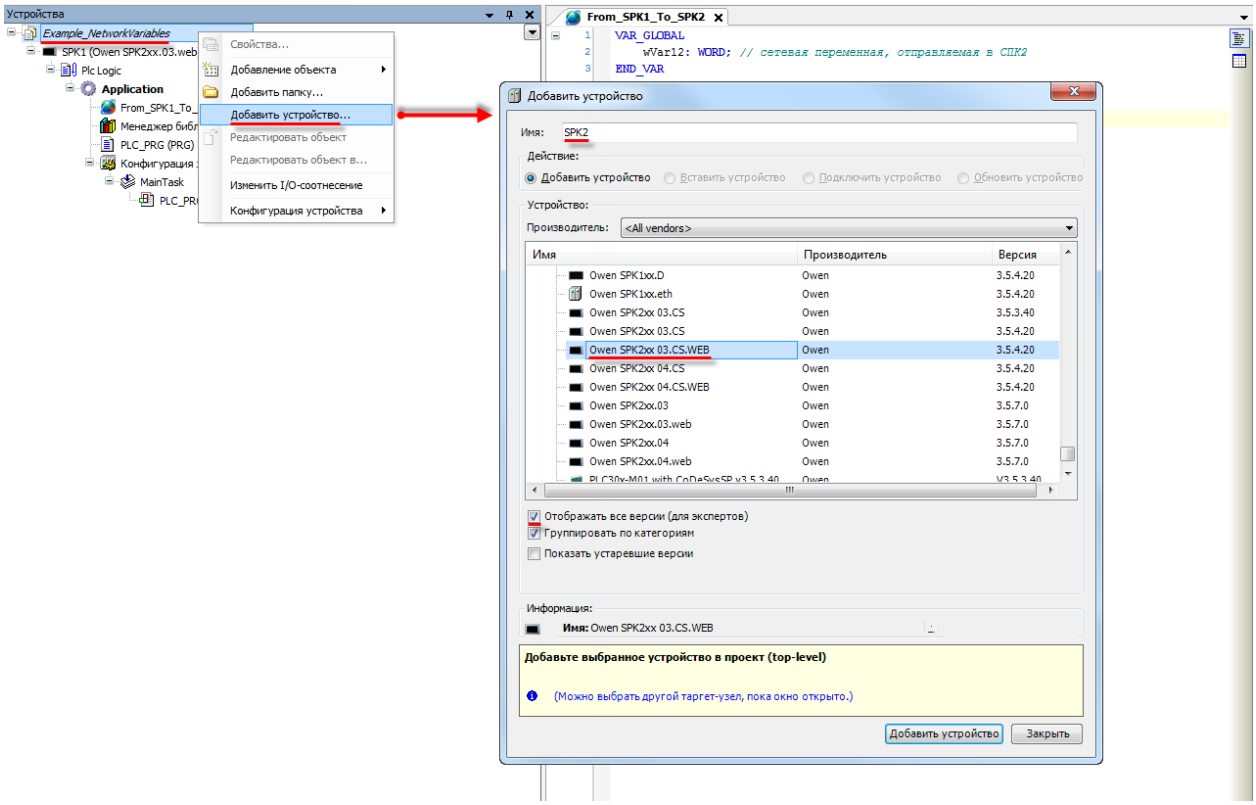


Рис. 3.13. Добавление контроллера **SPK2**

6. В приложение **Application** устройства **SPK2** добавьте компонент **Конфигурация задач** с задачей **MainTask**. При необходимости можно добавить также программу, но в рамках примера это делать необязательно.

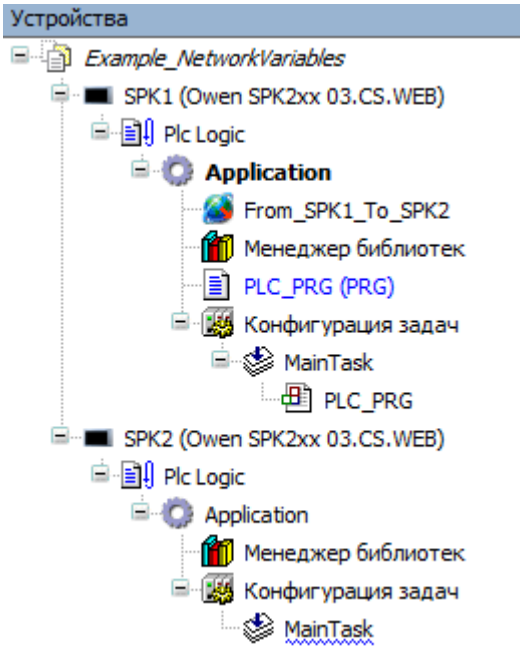


Рис. 3.14. Дерево проекта после добавления **SPK2**

7. В приложение **Application** устройства **SPK2** добавьте компонент Список сетевых переменных (получатель) с импортом списка **From_SPK1_To_SPK2**:

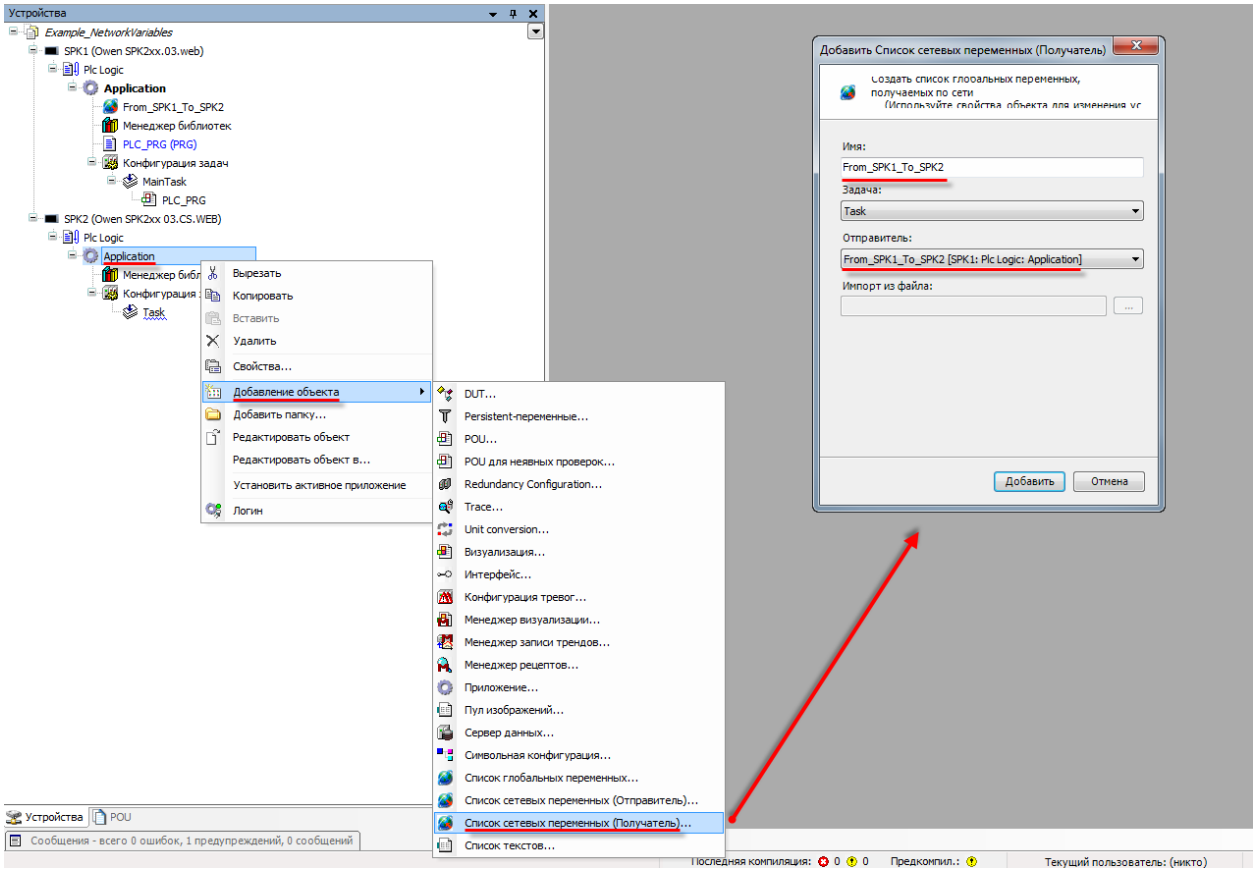


Рис. 3.15. Импорт списка сетевых переменных из **SPK1**

В результате в **SPK2** будут импортированы сетевые переменные и сетевые настройки списка **From_SPK1_To_SPK2**:

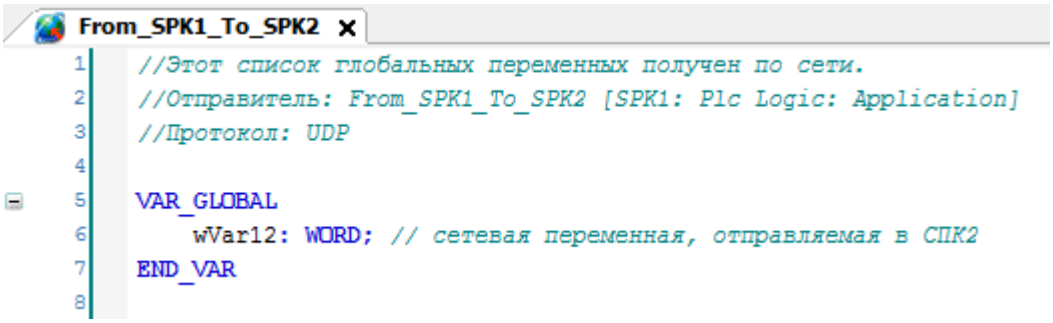


Рис. 3.16. Список получаемых сетевых переменных **SPK2**

На этом организация передачи сетевой переменной **wVar12** из **SPK1** в **SPK2** завершена. Теперь организуем передачу переменной из **SPK2** в **SPK1**.

8. В приложение **Application** устройства **SPK2** добавьте компонент Список сетевых переменных (отправитель) названием **From_SPK2_To_SPK1** и настройками в соответствии с табл. 3.1. ***Обратите внимание***, что идентификатор и используемый порт будут уникальными для каждого списка:

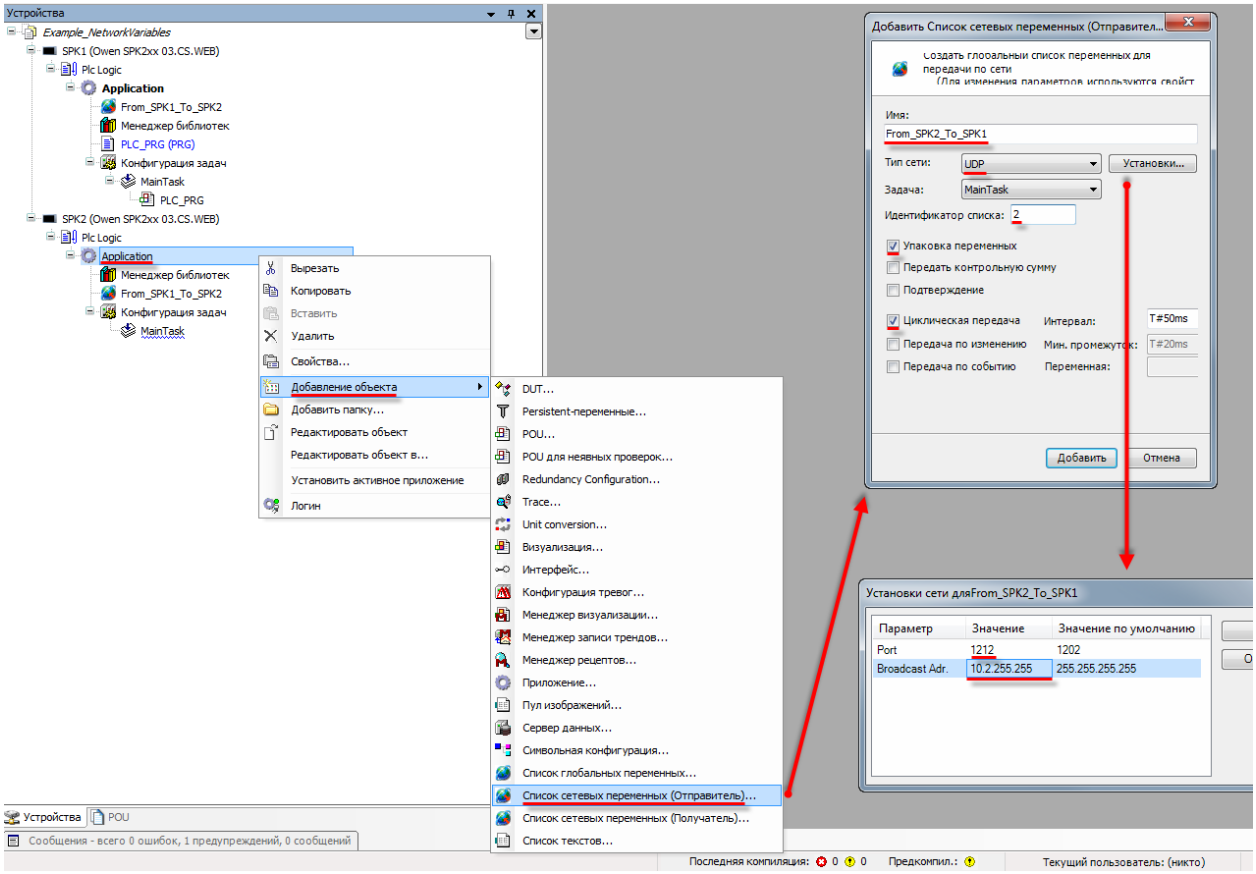


Рис. 3.17. Настройки компонента **Список сетевых переменных (отправитель SPK2)**

9. В созданном списке объявите переменную **wVar21** типа **WORD**. Значение этой переменной будет передаваться в **СПК1** каждые 50 мс.

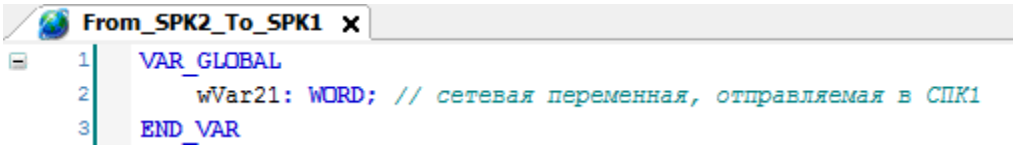


Рис. 3.18. Объявление сетевой переменной (**SPK2**)

10. В приложение **Application** устройства **SPK1** добавьте компонент Список сетевых переменных (получатель) с импортом списка **From_SPK2_To_SPK1**:

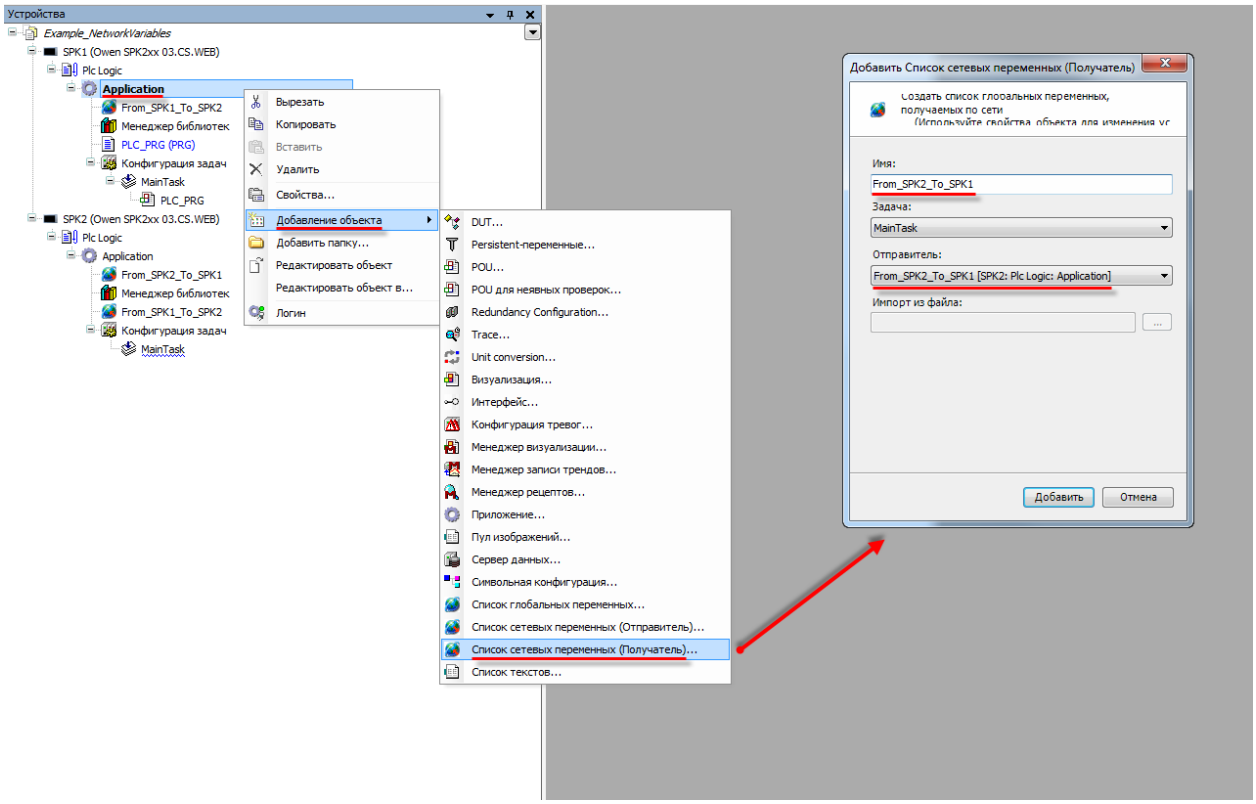


Рис. 3.19. Импорт списка сетевых переменных из **SPK2**

В результате в **SPK1** будут импортированы сетевые переменные и сетевые настройки списка **From_SPK2_To_SPK1**:

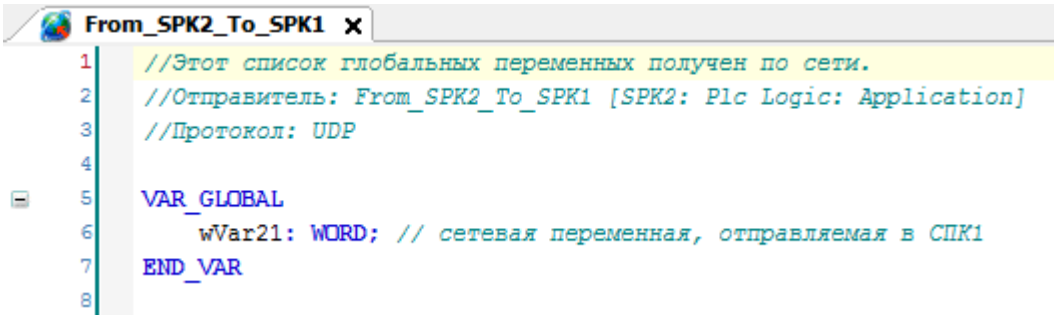


Рис. 3.20. Список получаемых сетевых переменных **SPK1**

На этом организация передачи сетевой передачи **wVar21** из **SPK2** в **SPK1** завершена.

11. Для отображения и изменения сетевых переменных, создадим для каждого из контроллеров экран визуализации с элементами **Бегунок** и **Отображение линейки**:

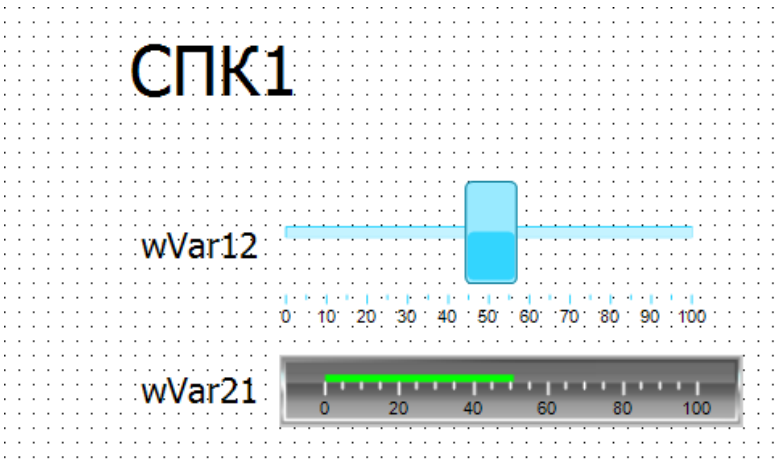


Рис. 3.21. Экран визуализации контроллера **СПК1**

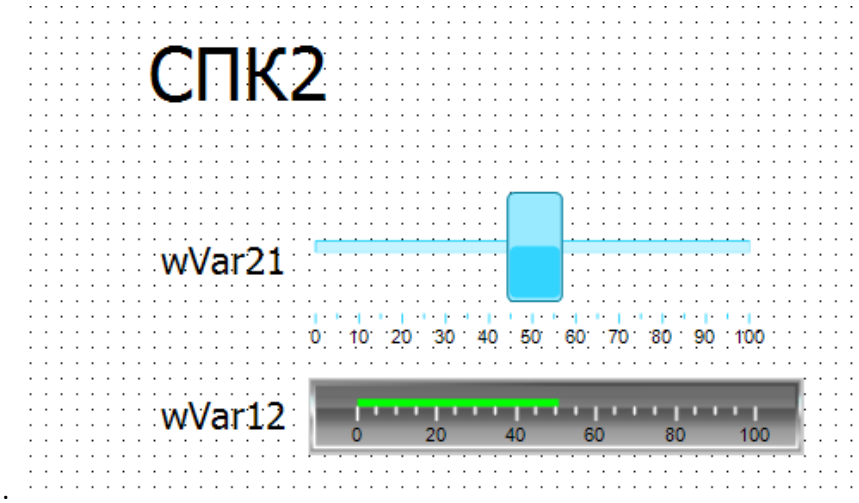


Рис. 3.22. Экран визуализации контроллера **СПК2**

К каждому из элементов привяжем соответствующую переменную из его приложения:

Контроллер	**Элемент**	**Переменная**
SPK1	Бегунок	From_SPK1_To_SPK2.wVar12
	Отображение линейки	From_SPK2_To_SPK1.wVar21
SPK2	Бегунок	From_SPK2_To_SPK1.wVar21
	Отображение линейки	From_SPK1_To_SPK2.wVar12

В свойствах экранов укажем разрешение **800x480**. В **Менеджере визуализации** поставим галочку **Использовать строки Unicode**.

Более подробно вопросы создания экранов визуализации рассмотрены в документе **СПК. Визуализация.**

12. Загрузите проекты в каждый из СПК. На **СПК1** изменяйте значение переменной **wVar12** с помощью ползунка и наблюдайте соответствующие изменения на дисплее **СПК2**. На **СПК2** изменяйте значение переменной **wVar21** с помощью ползунка и наблюдайте соответствующие изменения на дисплее **СПК1**.

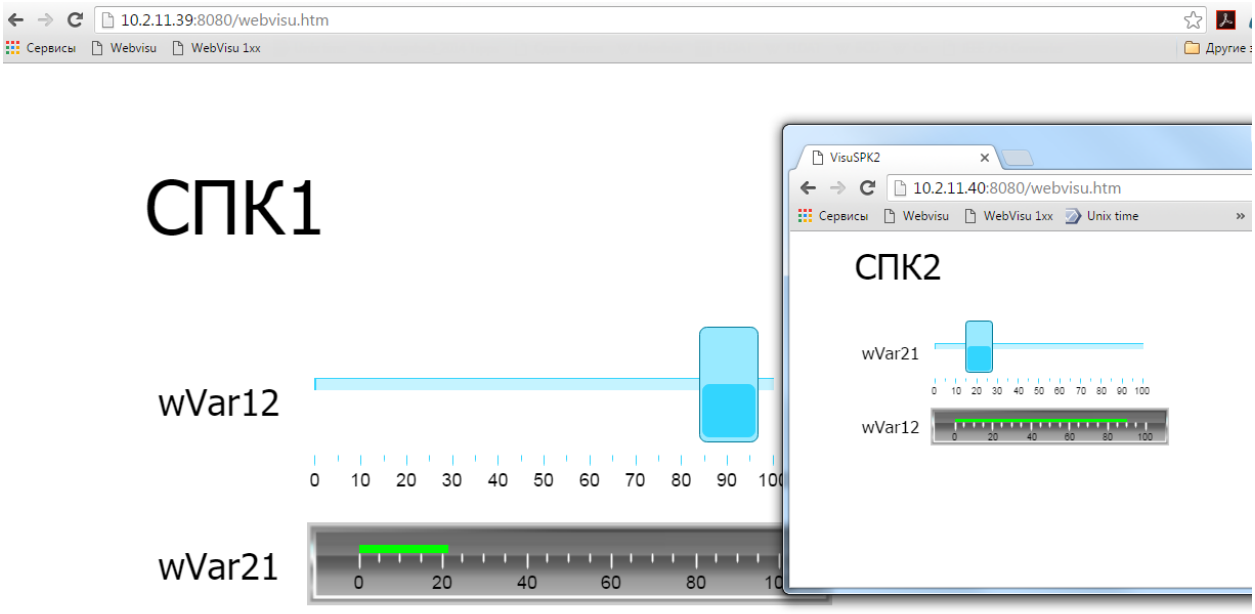


Рис. 3.23. Работа с примером в web-визуализации

4. Связь со SCADA-системой через OPC-сервер

4.1. Основные сведения об OPC

OPC (OLE for **P**rocess **C**ontrol) – стандарт семейства программных технологий, предоставляющих единый интерфейс для управления объектами автоматизации и технологическими процессами. Одной из наиболее распространенных в настоящее время спецификаций является OPC DA (**d**ata **a**ccess), использующаяся для обмена данными между различными устройствами.

Главной целью разработки стандарта OPC являлось обеспечение возможности интеграции средств автоматизации, функционирующих на разных платформах, в разных промышленных сетях и производимых различными фирмами. В настоящее время, OPC-сервер является неотъемлемым компонентом практически любой продвинутой АСУ и используется для сбора данных и их последующей передачи в SCADA-систему.

В рамках данного документа рассмотрены вопросы настройки обмена сенсорных панельных контроллеров **Овен СПК** со SCADA-системой MasterSCADA при использовании различных OPC-серверов:

- CODESYS V3 OPC (протокол **Gateway** поверх Ethernet)
- MasterOPC Universal Modbus Server (протокол **Modbus TCP**)
- Lectus Modbus OPC/DDE Server (протокол **Modbus TCP**)
- ОВЕН OPC (протокол **Modbus TCP**)

Сравнительные характеристики OPC-серверов приведены в табл. 4.1.

Типичная структурная схема обмена приведена на рис. 4.1:

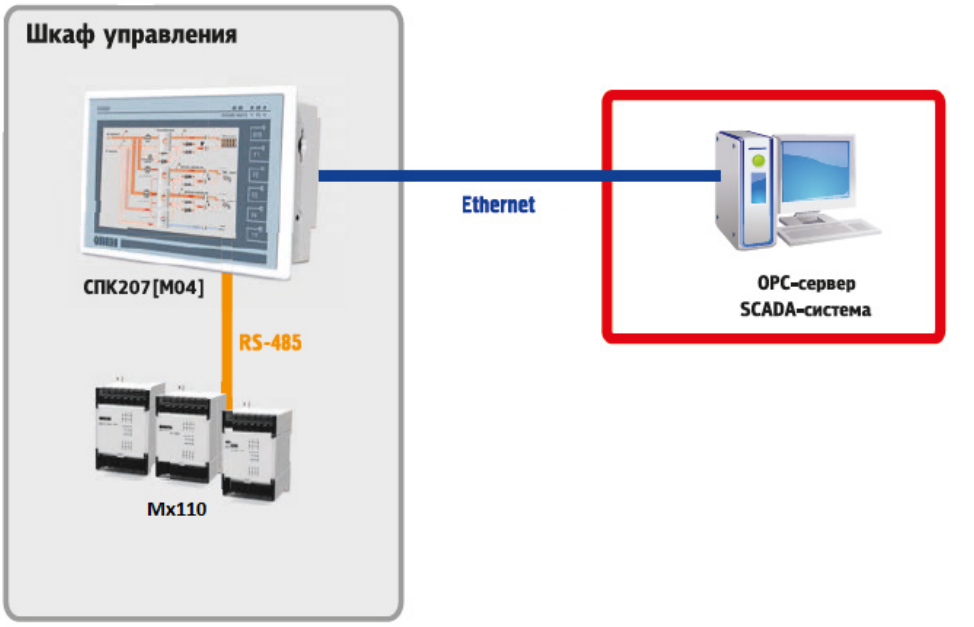


Рис. 4.1. Структурная схема связи СПК и SCADA-системы через OPC-сервер

Табл. 4.1. Сравнительные характеристики OPC-серверов

Функция	**CODESYS OPC Server V3**	**MasterOPC Universal Modbus Server**	**Lectus OPC**	**ОВЕН OPC (новый)**
Modbus RTU	+	+	+	+
Modbus ASCII	-	+	+	+
Modbus TCP	+	+	+	+
Чтение архивов ПЛК (20-я функция Modbus)	-	+	+	-
Визуальный контроль значений переменных	-	+	+	+
Поддержка скриптов	-	+	+	-
Работа с SQL-сервером	-	+	+	-
Готовые конфигурации для приборов ОВЕН	-	+- (*не для всех приборов*)	+- (*не для всех приборов*)	+
Экспорт таблицы переменных из OWEN Logic	-	-	-	+
Модель распространения	бесплатный	Платный (*доступна бесплатная версия на 32 тега*)	Платный (*доступна пробная версия на 30 дней*)	бесплатный

Методика настройки обмена СПК и SCADA-системы через OPC-сервер следующая:

1. Настройка СПК (настройка **символьной конфигурации** или настройка СПК как **Modbus Slave**);

2. Настройка OPC-сервера;

3. Подключение OPC-сервера к SCADA-системе.

4.2. CODESYS OPC Server V3

4.2.1. Настройка СПК

CODESYS OPC Server V3 – самый простой с точки зрения настройки OPC-сервер для организации обмена с СПК, так как он интегрирован в среду разработки и позволяет автоматически импортировать переменные проекта.

1. Создайте новый проект **CODESYS** для **СПК207.03** (язык программы не имеет значения).

2. В программе **PLC_PRG** объявите следующие переменные:

```
PLC_PRG
1  PROGRAM PLC_PRG
2  VAR
3      xVar: BOOL;    // логическое значение
4      iVar: INT;     // целое число
5      rVar: REAL;    // число с плавающей точкой
6  END_VAR
```

Рис. 4.2.1. Объявление переменных программы **PLC_PRG**

3. Добавьте в проект компонент **Символьная конфигурация**:

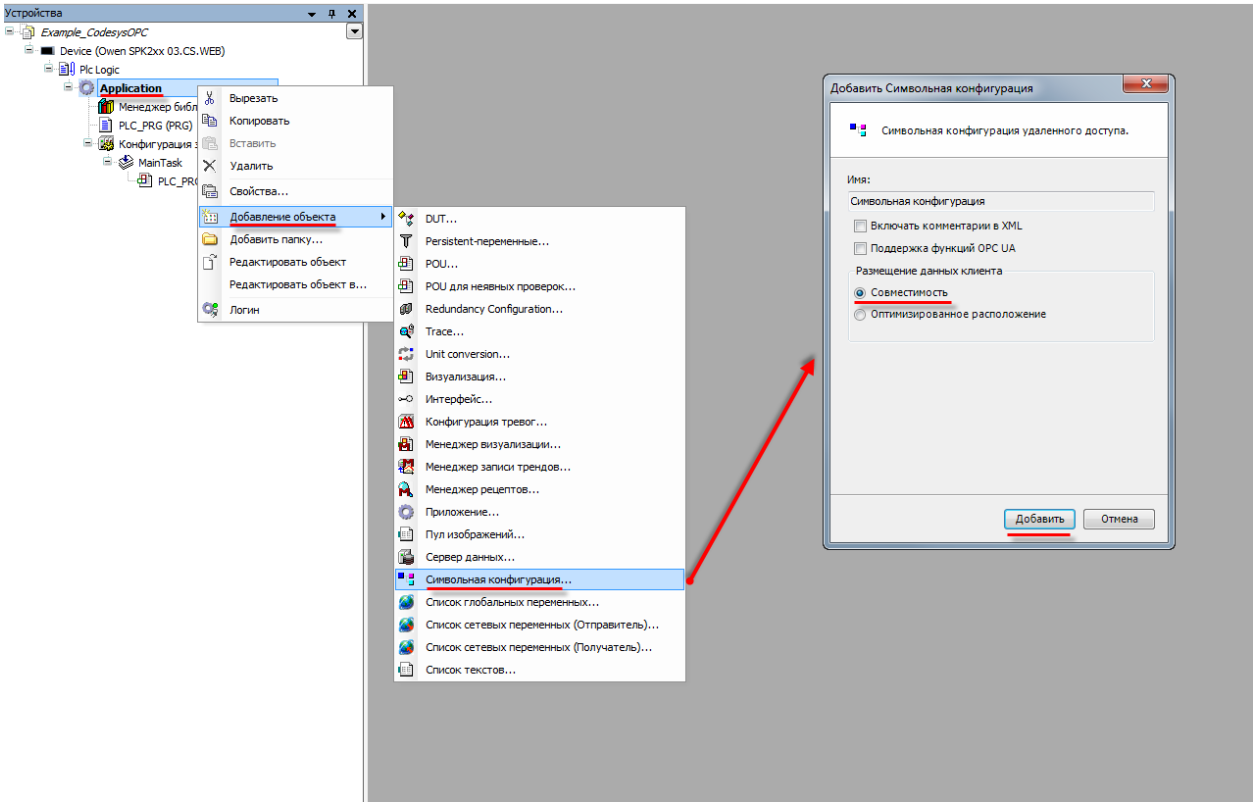


Рис. 4.2.2. Добавление компонента **Символьная конфигурация**

При добавлении компонента пользователь может выбрать следующие настройки:

Включить комментарии в XML – при наличии галочки в файл символьной конфигурации будут включены комментарии к переменным;

Поддержка функций OPC UA – при наличии галочки в файл символьной конфигурации добавляется дополнительная информация, необходимая для поддержки функций **OPC UA сервера**. OPC UA сервер является дополнительным платным компонентом CODESYS, который ***в данный момент не поддерживается СПК***;

Размещение данных клиента – у пользователя имеется возможность выбрать структуру файла символьной конфигурации – совместимую со старыми версиями или оптимизированную. Оптимизированная структура поддерживается начиная с **CODESYS 3.5 SP7**. Рекомендуется всегда использовать совместимую структуру.

4. После добавления компонента **Символьная конфигурация** необходимо выполнить компиляцию проекта:

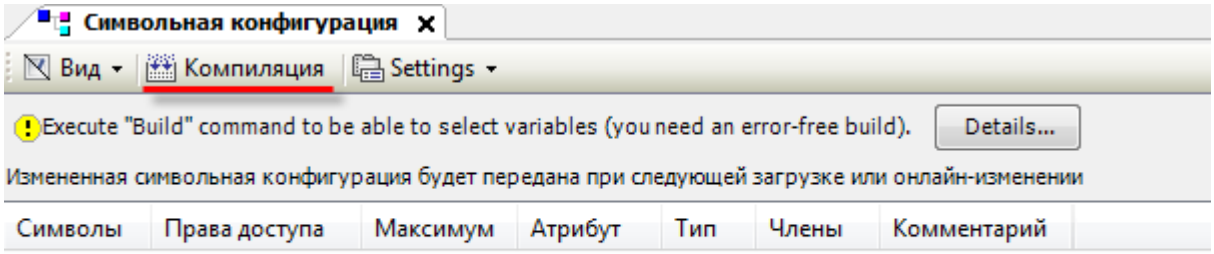


Рис. 4.2.3. Кнопка компиляции проекта после создания символьной конфигурации

Обратите внимание, что при добавлении в проект новых переменных, для внесения изменений в символьную конфигурацию предварительно требуется выполнить компиляцию проекта.

5. Пометьте галочками переменные, которые будут считываться/изменяться OPC-сервером и укажите для каждой из них права доступа (со стороны OPC-сервера). Для прав доступа используются следующие пиктограммы:

 - только чтение;

 - только запись;

 - чтение/запись.

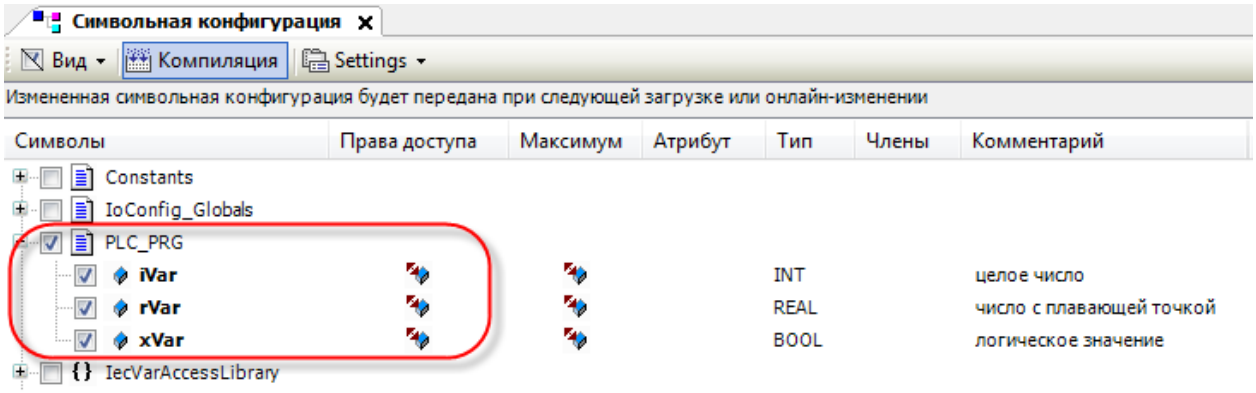


Рис. 4.2.4. Добавление компонента **Символьная конфигурация**

На этом настройка проекта СПК завершена. После загрузки проекта необходимо выполнить команду **Создать загрузочное приложение** из меню **Онлайн**.

Созданный в данном пункте проект доступен для скачивания: Example_CodesysOPC.projectarchive

Пример создан в среде **CODESYS 3.5 SP7 Patch4** и подразумевает запуск на **СПК207.03.CS(-WEB)** с таргет-файлом **3.5.4.20 (023).**

4.2.2. Настройка OPC-сервера

1. Запустите приложение **OPC Configurator** (из **меню Пуск** или папки **CODESYS OPC Server V3**, расположенной в директории установки **CODESYS**).

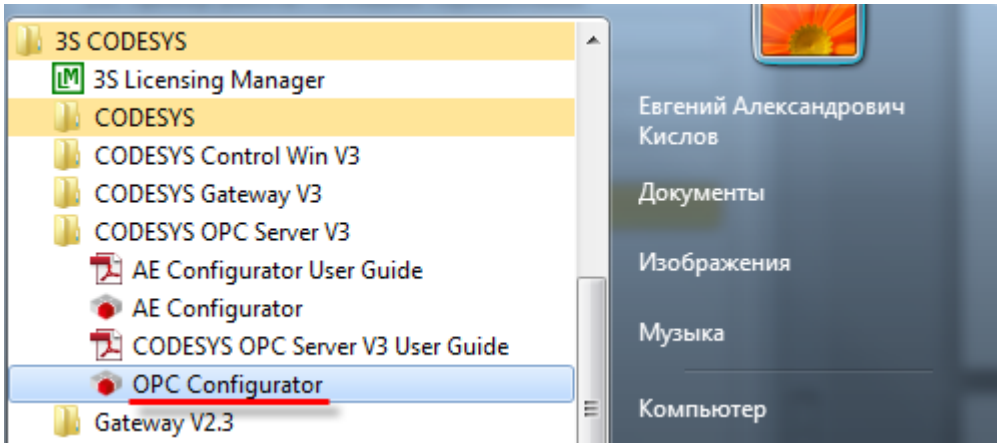


Рис. 4.2.5. Запуск приложения **OPC Configurator**

2. Нажмите **ПКМ** на узел **Server** и в контекстном меню выберите команду **Append PLC**:

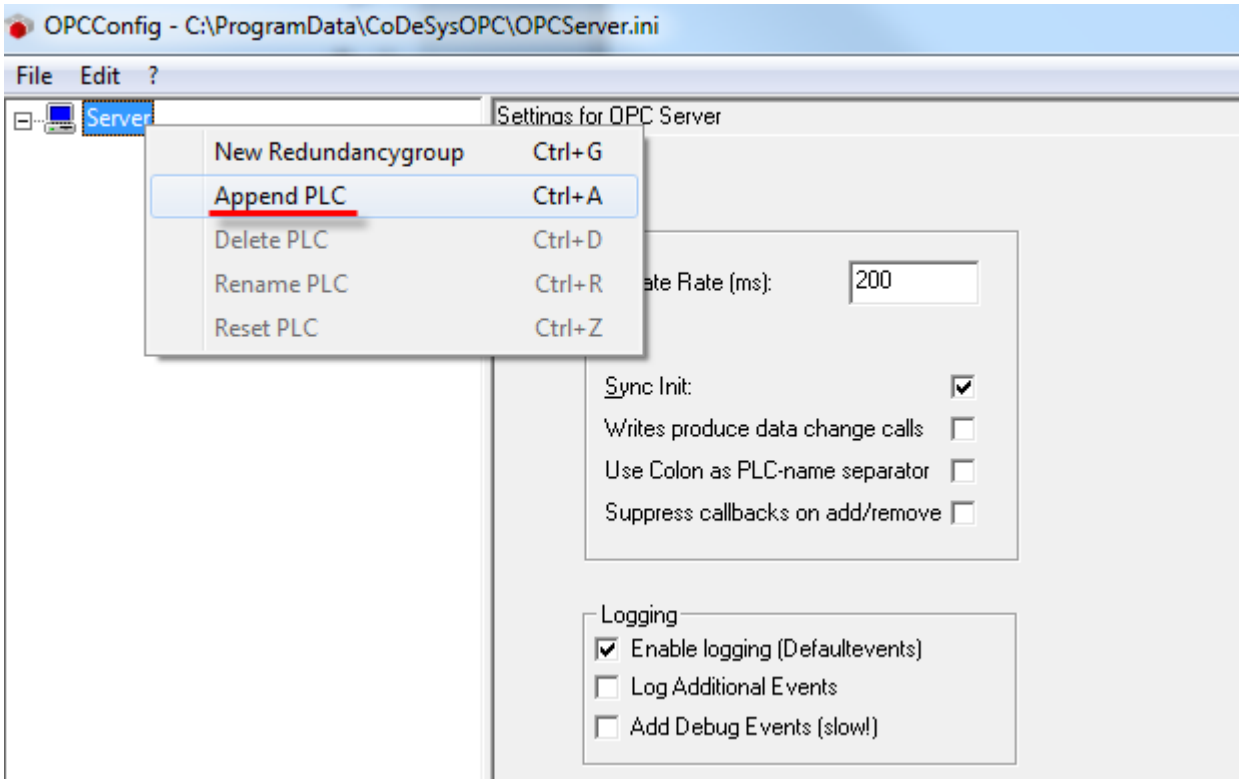


Рис. 4.2.6. Добавление контроллера в OPC-сервер

3. На вкладке **PLC1** укажите интерфейс, по которому будут связаны контроллер и OPC-сервер – **GATEWAY3** (**Ethernet**) или **ARTI3** (**Serial Line**). В данном примере контроллер будет подключаться по **Ethernet**.

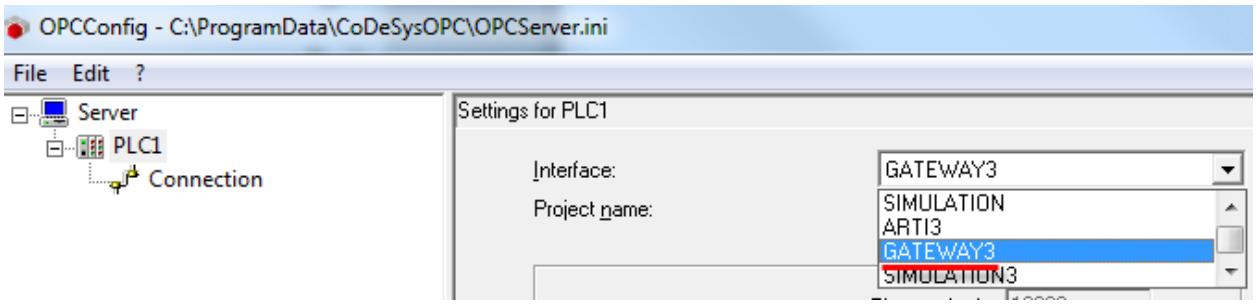


Рис. 4.2.7. Выбор интерфейса связи контроллера и OPC-сервера

4. На вкладке **Connection** нажмите кнопку **Edit** и укажите IP-адрес вашего контроллера.

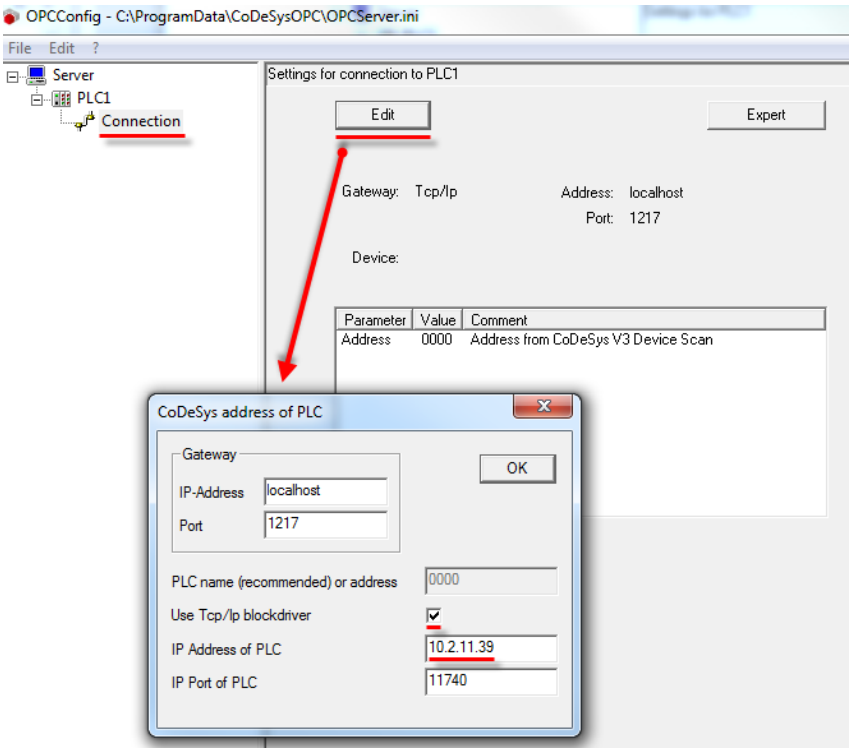


Рис. 4.2.8. Указание IP-адреса контроллера

5. Сохраните настройки OPC-сервера:

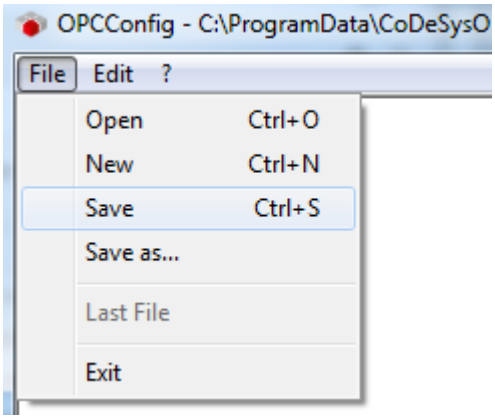


Рис. 4.2.9. Сохранение настроек OPC-сервера

На этом настройка OPC-сервера завершена. Приложение **OPC Configurator** можно закрыть.

Теперь вам необходимо загрузить проект, созданный в п. 4.2.1 в СПК и убедиться, что контроллер находится в одной локальной сети с OPC-сервером. После этого можно переходить к п. 4.6.

4.3. MasterOPC Universal Modbus Server

4.3.1. Настройка СПК

1. Создайте новый проект **CODESYS** для **СПК207.03** (язык программы не имеет значения).

2. Добавьте в проект объединение с именем **Real_Word**:

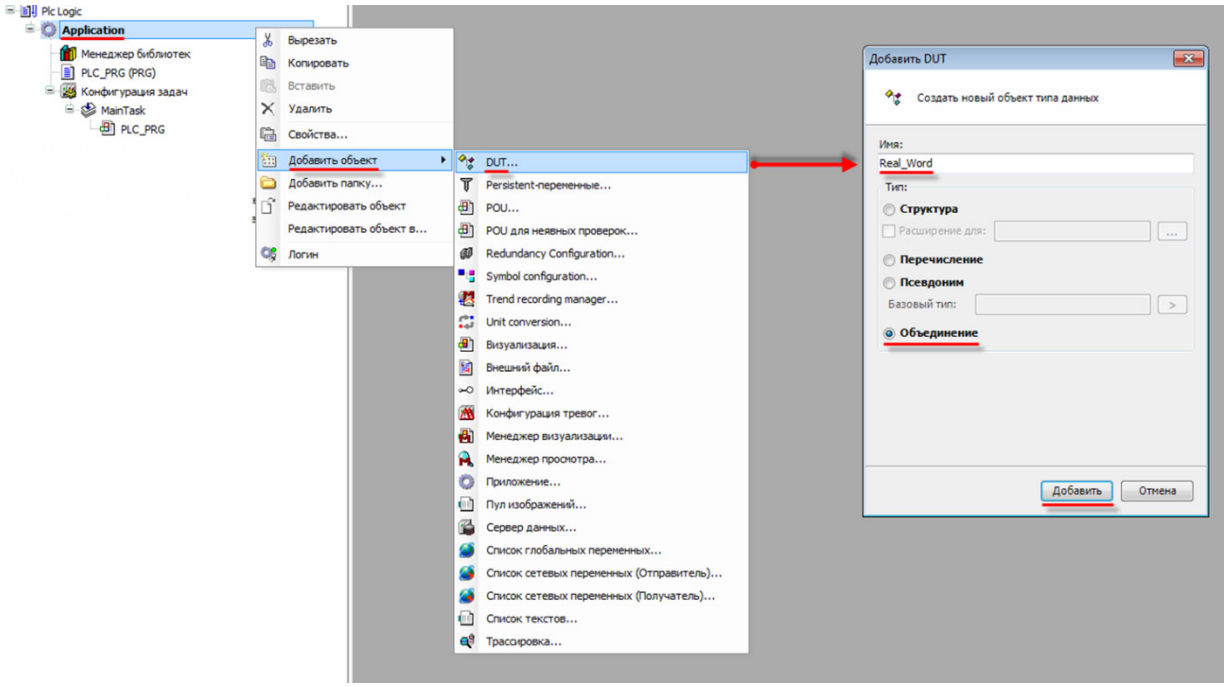


Рис. 4.3.1. Добавление в проект объединения

В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

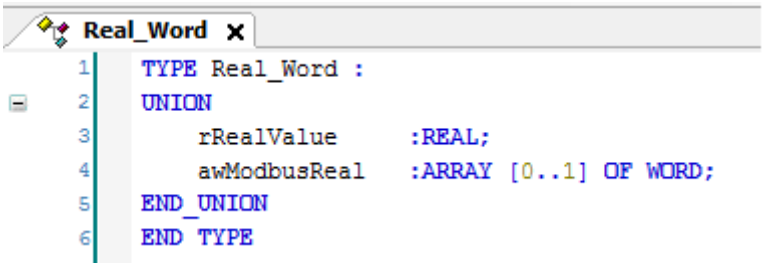


Рис. 4.3.2. Объявление переменных объединения

3. В программе **PLC_PRG** объявите следующие переменные:

```
PROGRAM PLC_PRG
VAR
    (*данные, которые считывает OPC-сервер*)
    xVar_OPC_read:          BOOL;       // логическое значение
    wVar_OPC_read:          WORD;       // целое число
    _rVar_OPC_read:         Real_Word;  // число с плавающей точкой

    (*данные, которые записывает OPC-сервер*)
    xVar_OPC_write:         BOOL;       // логическое значение
    wVar_OPC_write:         WORD;       // целое число
    _rVar_OPC_write:        Real_Word;  // число с плавающей точкой
END_VAR
```

Рис. 4.3.3. Объявление переменных программы **PLC_PRG**

4. Добавьте в проект компонент **Ethernet**. ***Обратите внимание***, что версия компонента не должна превышать версию **target-файла** СПК.

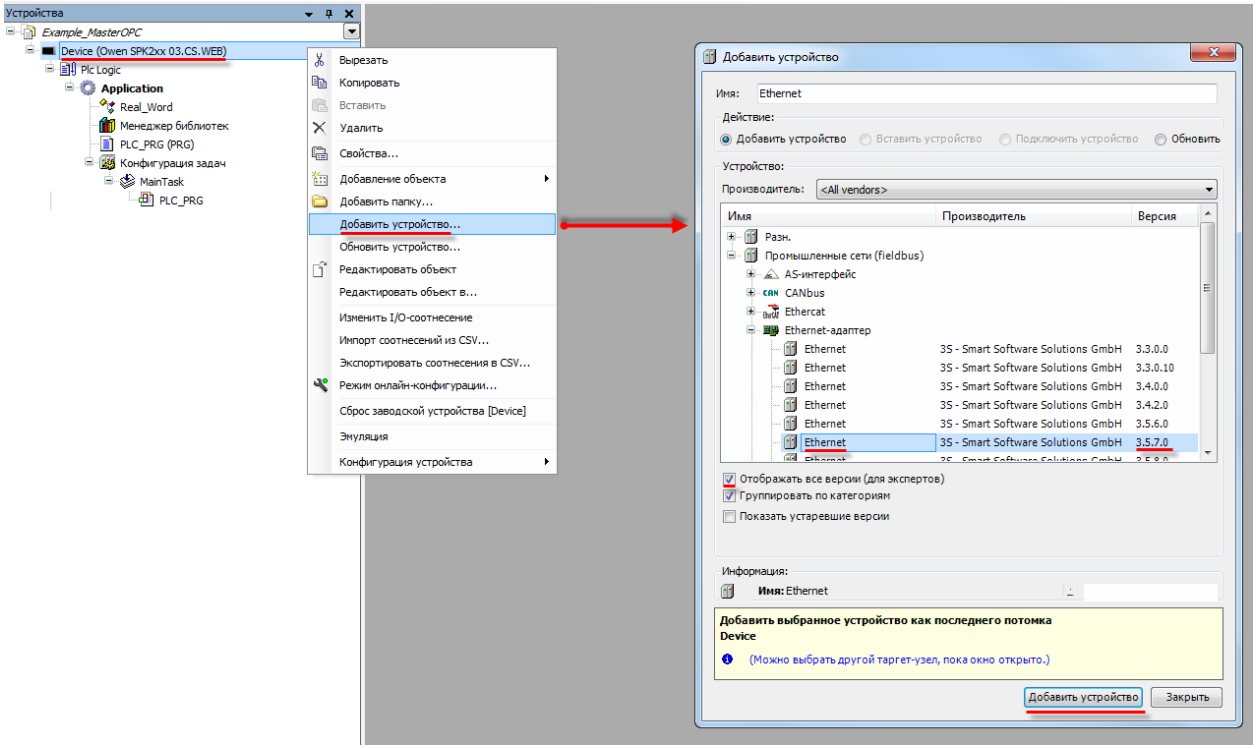


Рис. 4.3.4. Добавление компонента **Ethernet**

На вкладке **Конфигурация Ethernet** укажите сетевые настройки контроллера (должны соответствовать настройкам в **конфигураторе СПК**):

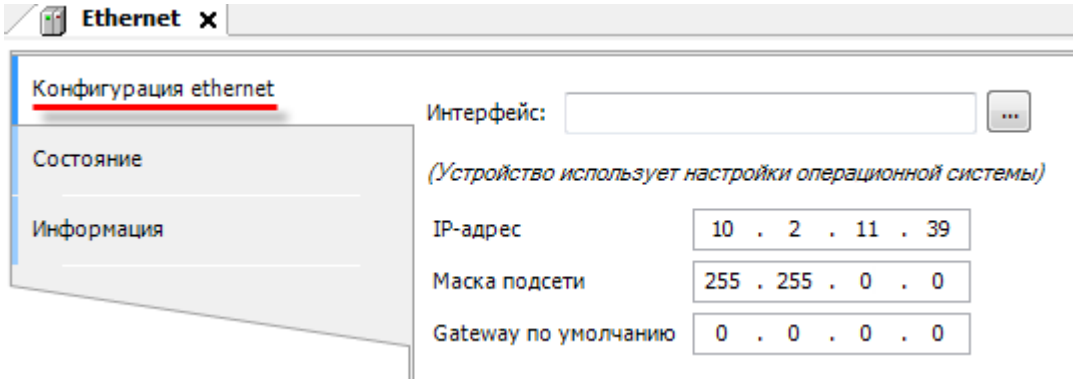


Рис. 4.3.5. Настройки компонента **Ethernet**

5. В компонент **Ethernet** добавьте компонент **Modbus TCP Slave Device**. ***Обратите внимание***, что версия компонента не должна превышать версию **target-файла** СПК.

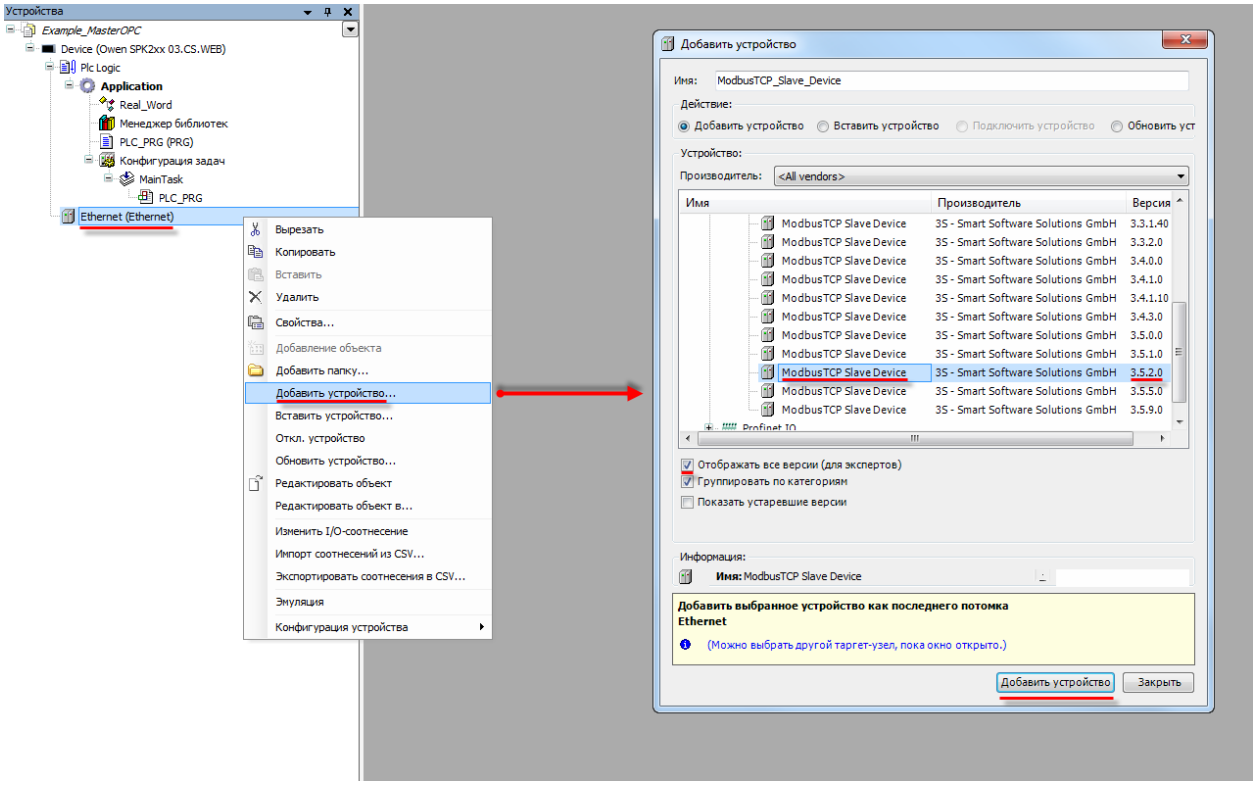


Рис. 4.3.6. Добавление компонента **Modbus TCP Slave Device**

В настройках компонента на вкладке **Страница конфигурации** укажите адрес slave-устройства (в примере используется адрес **1**) и порт для Modbus TCP (**502**).

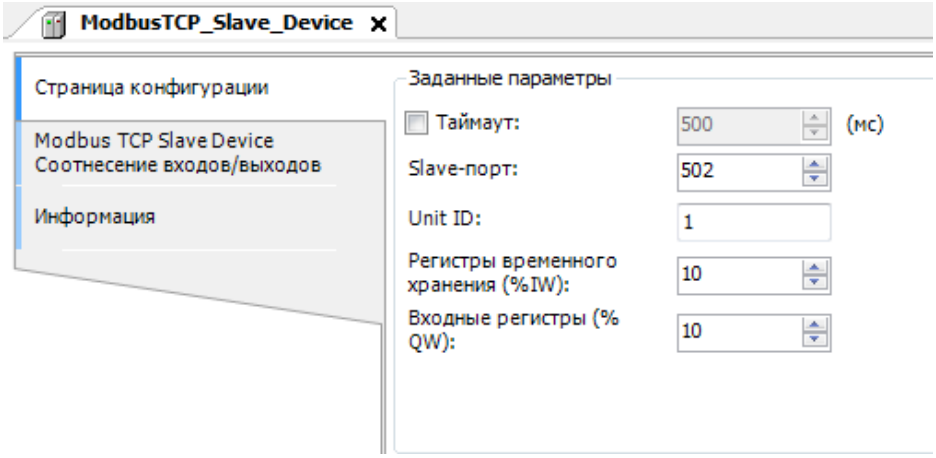


Рис. 4.3.7. Настройки компонента **Modbus TCP Slave Device**

На вкладке **Modbus TCP Slave Device Соотнесение входов/выходов** привяжите к регистрам переменные программы. Не забудьте у параметра **Всегда обновлять переменные** выставить значение **Включено 2**.

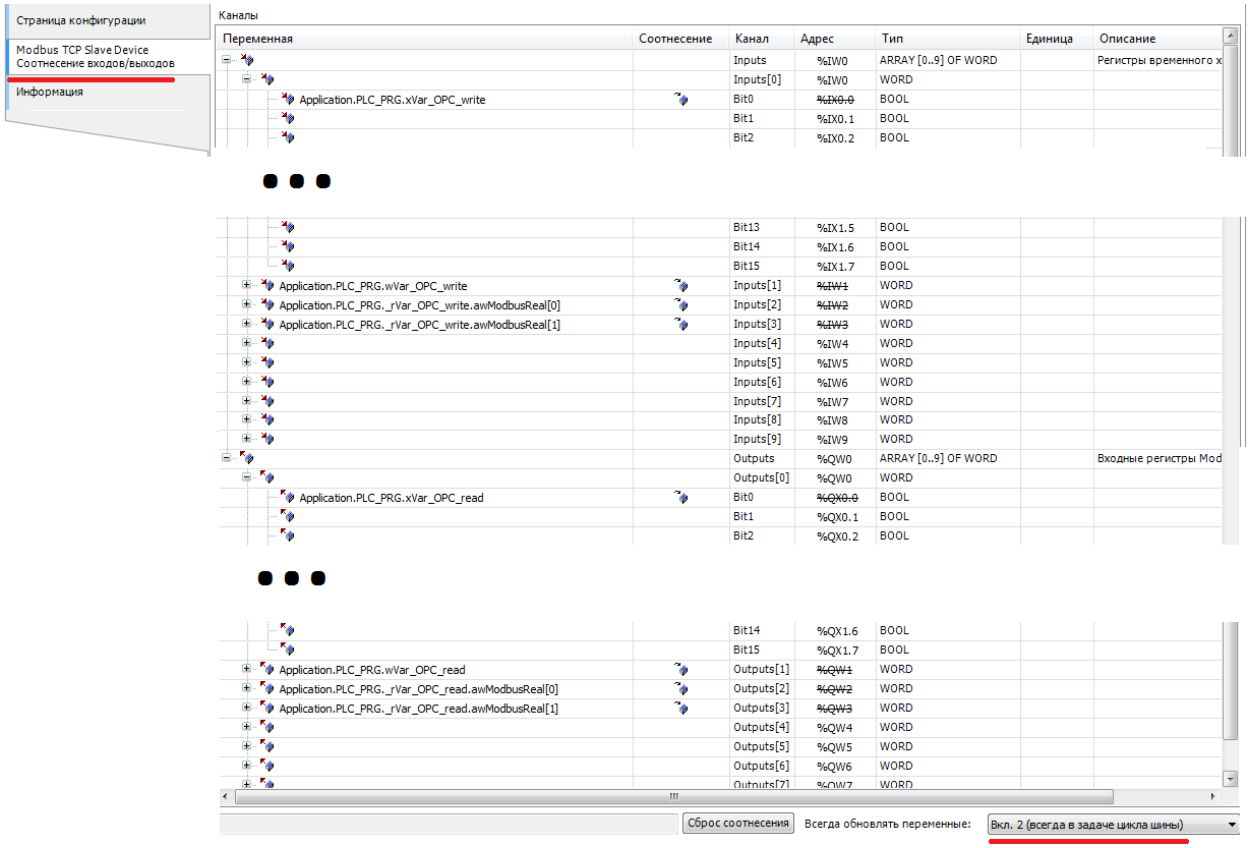


Рис. 4.3.8. Привязка переменных к регистрам

Обратите внимание, что **holding** регистры (к которым привязаны **OPC_write** переменные), могут записываться мастер-устройством (в данном случае – OPC-сервером), но не могут быть изменены из программы СПК, а **input** регистры (к которым привязаны **OPC_read** переменные) могут быть изменены из программы, но не могут быть записаны OPC-сервером.

В результате, на СПК будет сформирована следующая карта регистров:

Область памяти	Адрес	Переменная	Тип переменной
Holding Registers	0x0 (бит 0)	xVar_OPC_write	BOOL
	0x1	wVar_OPC_write	WORD
	0x2 – 0x3	rVar_OPC_write	REAL
Input Registers	0x0 (бит 0)	xVar_OPC_read	BOOL
	0x1	wVar_OPC_read	WORD
	0x2 – 0x3	rVar_OPC_read	REAL

Следует отметить, что **Modbus Slave,** созданный стандартными средствами конфигурирования, на СПК с прошивками **3.9xx** не поддерживает работу с битовыми функциями. В данном примере OPC-сервер будет для чтения/записи бит использовать функции работы с регистрами; в большинстве случаев представляется разумным работать с **BOOL** переменными с помощью битовой маски.

Более подробно вопросы настройки **Modbus Slave** рассмотрены в документе **СПК. Modbus**.

Созданный в данном пункте проект доступен для скачивания: Example_MasterOPC_LectusOPC.zip

Пример создан в среде **CODESYS 3.5 SP7 Patch4** и подразумевает запуск на **СПК207.03.CS(-WEB)** с таргет-файлом **3.5.4.20 (023).**

4.3.2. Настройка OPC-сервера

1. Установите и запустите MasterOPC Universal Modbus Server.

2. Нажмите **ПКМ** на узел **Server** и добавьте коммуникационный узел с названием **SPK**. В его настройках укажите тип **TCP/IP**, а также сетевые настройки (**IP-адрес** и **порт**). Сетевые настройки должны соответствовать настройкам СПК (см. п. 4.3.1, пп. 4-5).

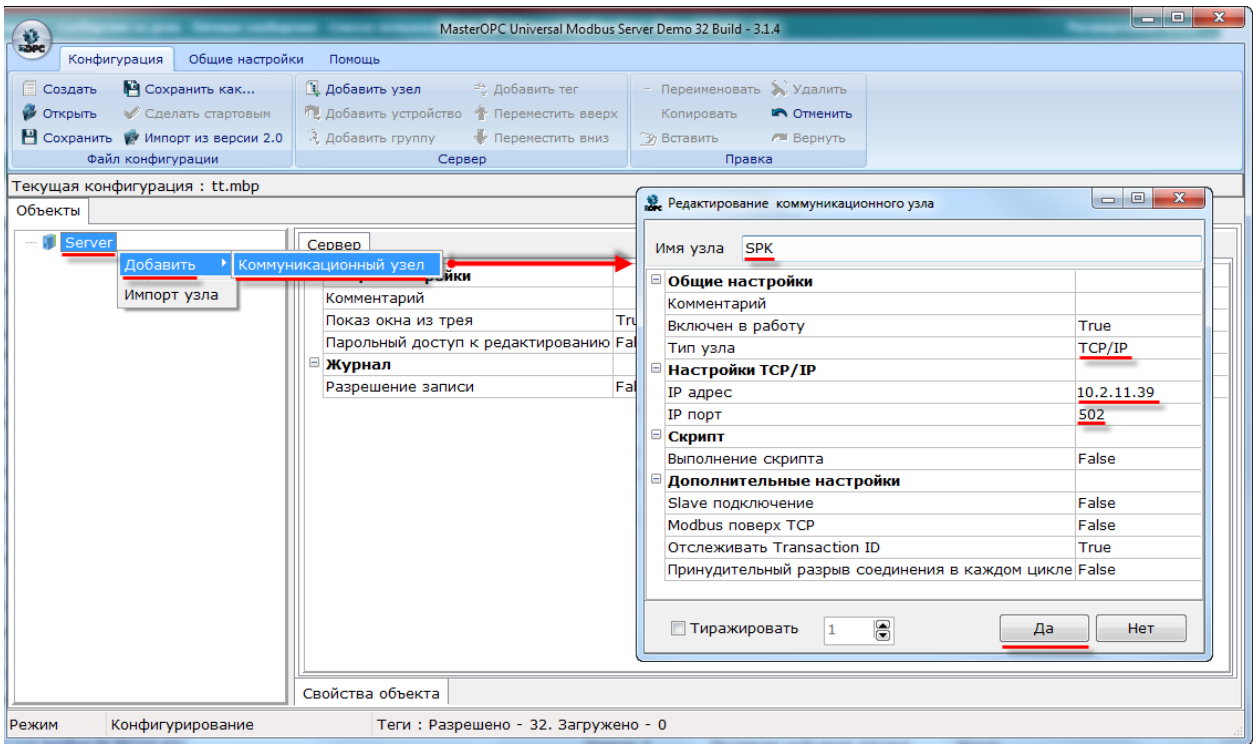


Рис. 4.3.9. Добавление коммуникационного узла

3. Нажмите **ПКМ** на узел **SPK** и добавьте устройство с названием **Slave ID 1**. В его настройках укажите адрес **1** (в соответствии с адресом СПК, см. п. 4.3.1, пп. 5). По умолчанию период опроса составляет 1000 мс – при необходимости следует уменьшить это значение.

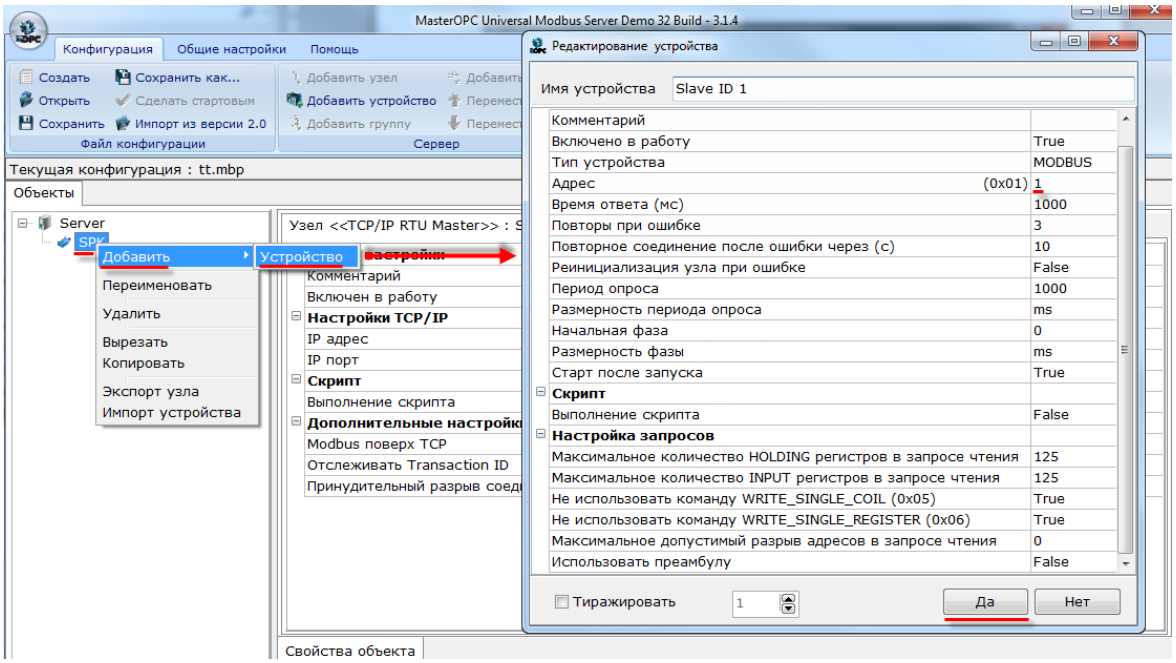


Рис. 4.3.10. Добавление устройства

4. Нажмите **ПКМ** на устройство **Slave ID 1** и добавьте 6 тегов. Число тегов соответствует числу переменных, считываемых/записываемых в СПК. Настройки тегов приведены ниже.

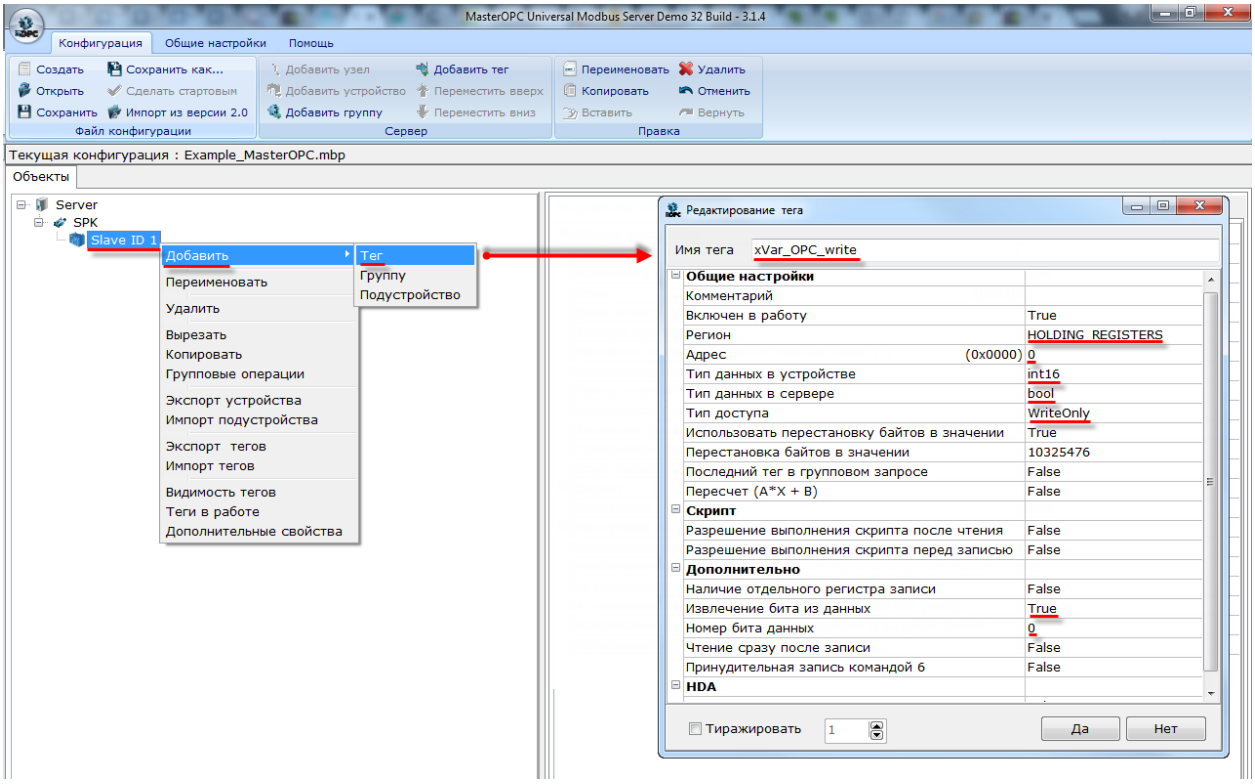


Рис. 4.3.11. Добавление и настройка тега **xVar_OPC_write**

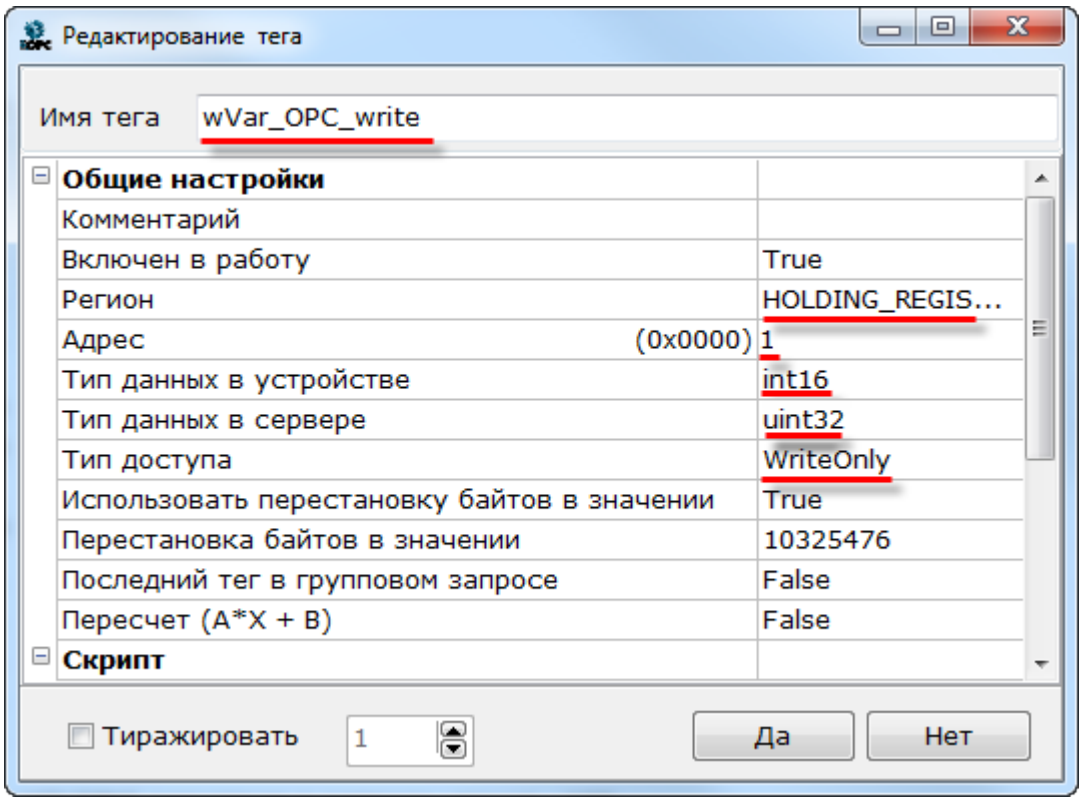


Рис. 4.3.12. Настройки тега **wVar_OPC_write**

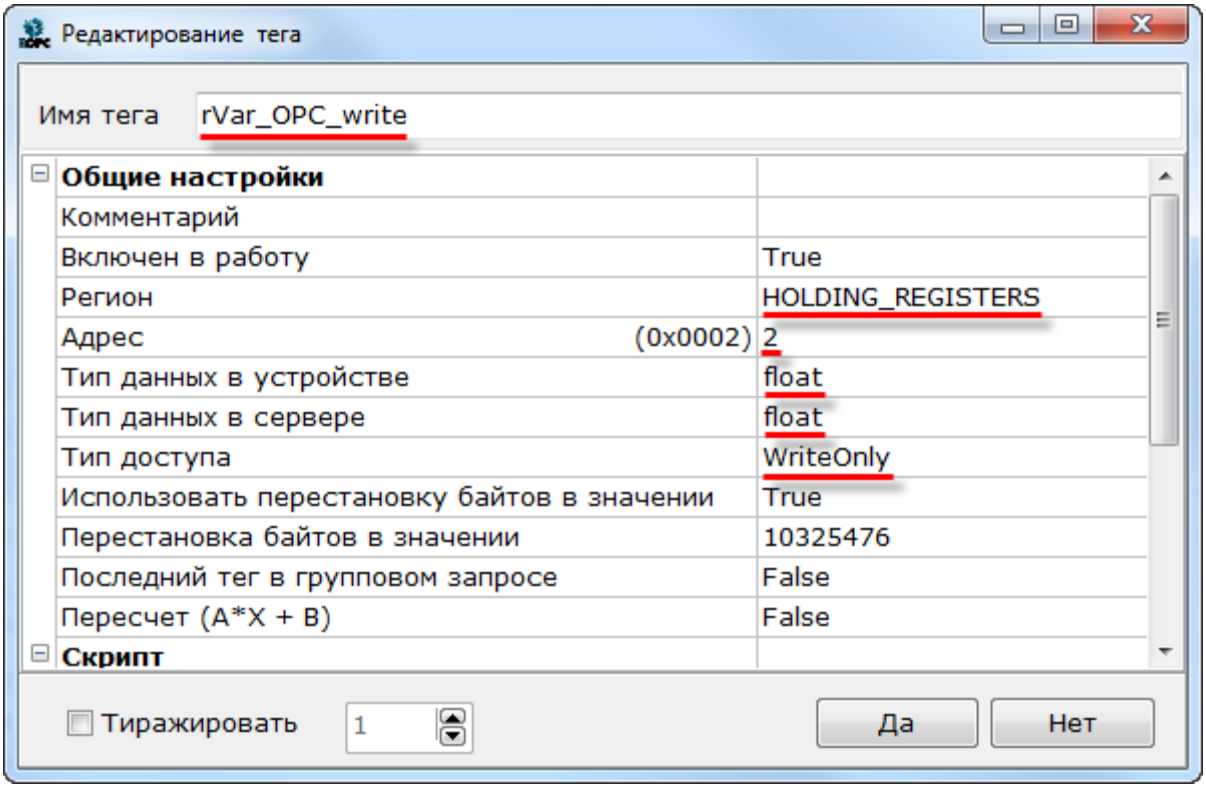


Рис. 4.3.13. Настройки тега **rVar_OPC_write**

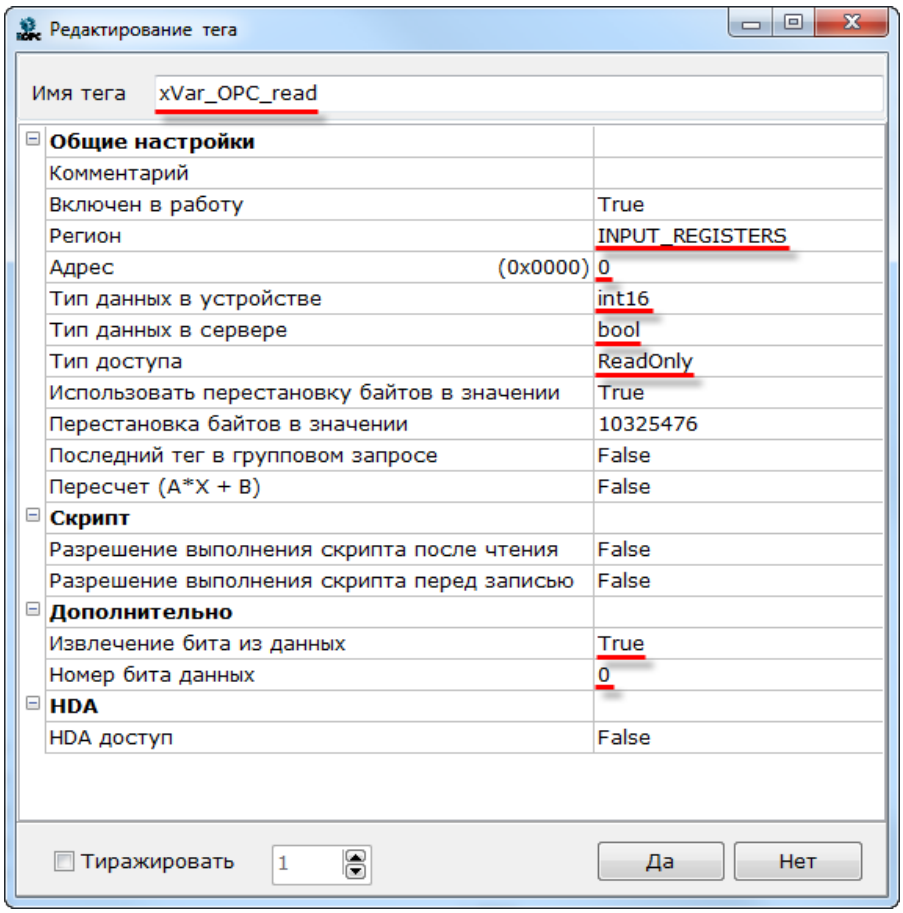


Рис. 4.3.14. Настройки тега **xVar_OPC_read**

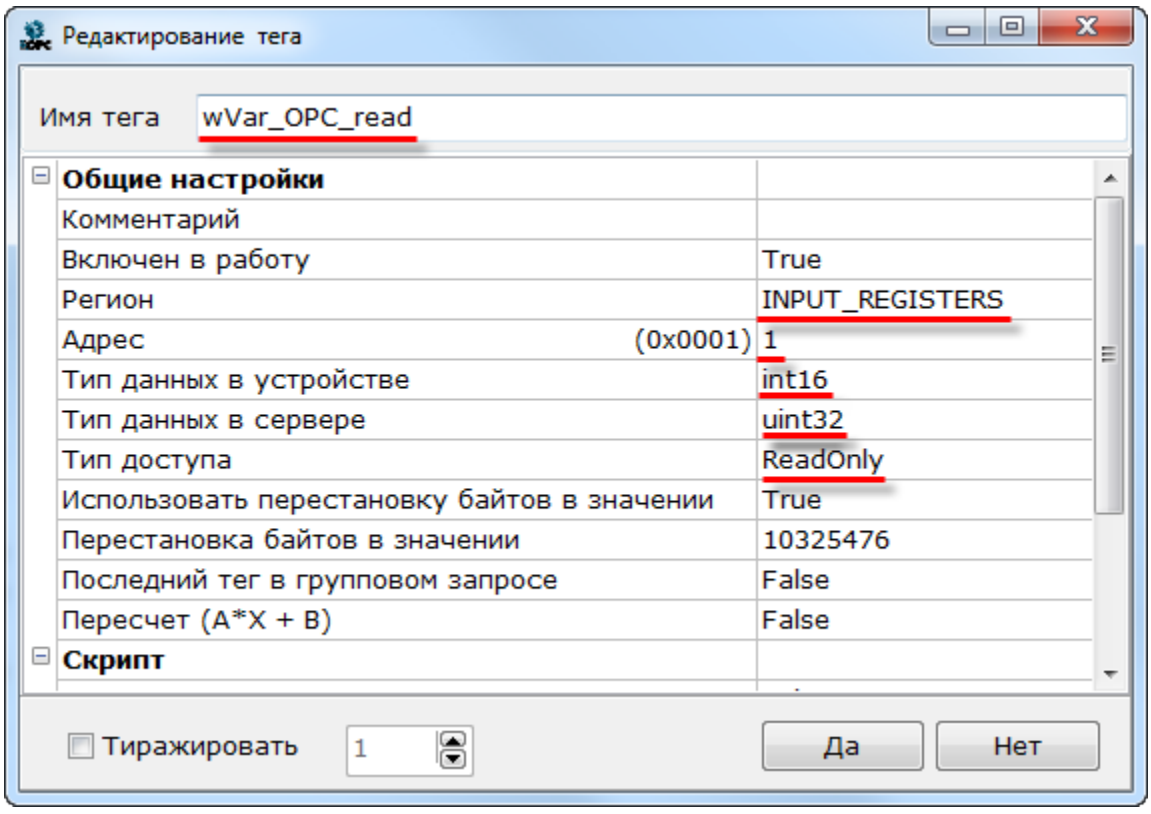


Рис. 4.3.15. Настройки тега **wVar_OPC_read**

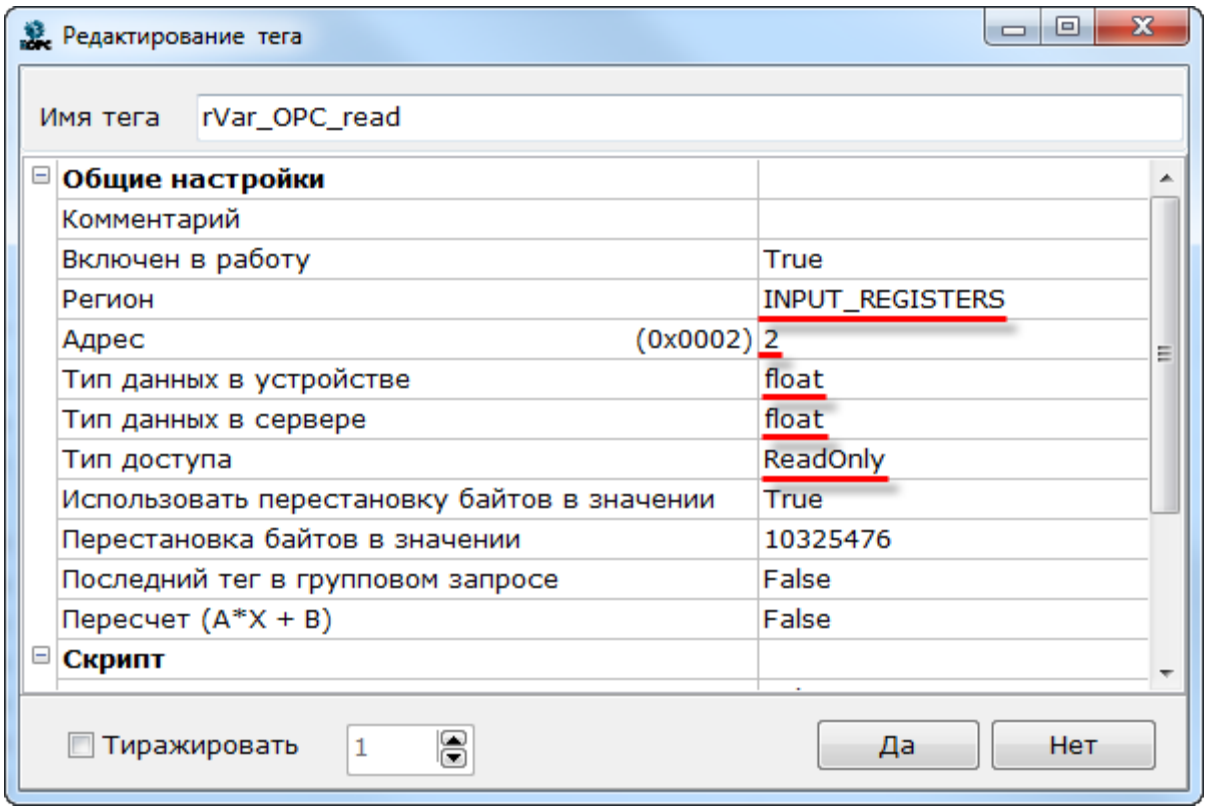


Рис. 4.3.16. Настройки тега **rVar_OPC_read**

5. После добавления и настройки тегов сохраните конфигурацию OPC-сервера:

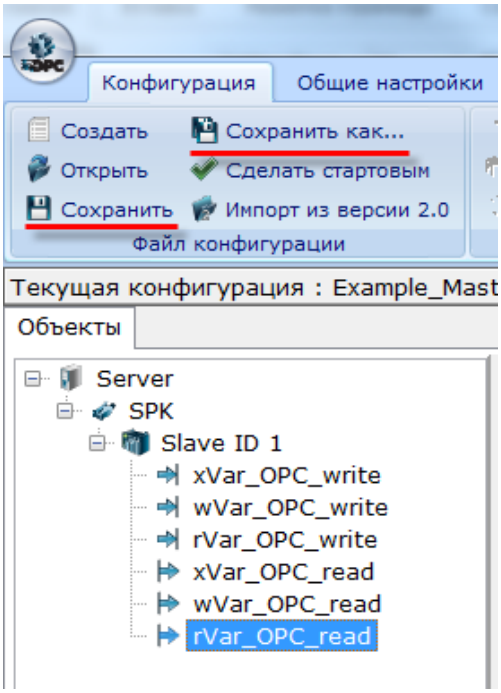


Рис. 4.3.17. Сохранение конфигурации OPC-сервера

Закройте OPC-сервер. Запускать его не требуется, поскольку SCADA-система производит этот процесс автоматически.

Теперь вам необходимо загрузить проект, созданный в п. 4.3.1 в СПК и убедиться, что контроллер находится в одной локальной сети с OPC-сервером. После этого можно переходить к п. 4.6.

Созданная в пункте конфигурация доступна для скачивания: Example_MasterOPC_LectusOPC_OwenOPC.zip

4.4. Lectus Modbus OPC/DDE Server

4.4.1. Настройка СПК

1. Создайте проект согласно п. 4.3.1.

4.4.2. Настройка OPC-сервера

1. Установите и запустите Lectus Modbus OPC/DDE Server.

2. Нажмите **ПКМ** на вкладку **Текущие данные** и добавьте узел с названием **OPC_read**. В его настройках укажите используемый тип подключения (**TCP клиент**), протокол (**Modbus TCP**), IP-адрес контроллера (в соответствии с п. 4.3.1 пп. 5), порт и адрес устройства (в соответствии с п. 4.3.1 пп. 6), а также функцию **Modbus**, используемую для чтения данных – **04** (**Read Input Registers**). Указывать функцию записи не имеет смысла, поскольку **input** регистры предназначены только для чтения. По умолчанию период опроса составляет 1 секунду – при необходимости следует уменьшить это значение.

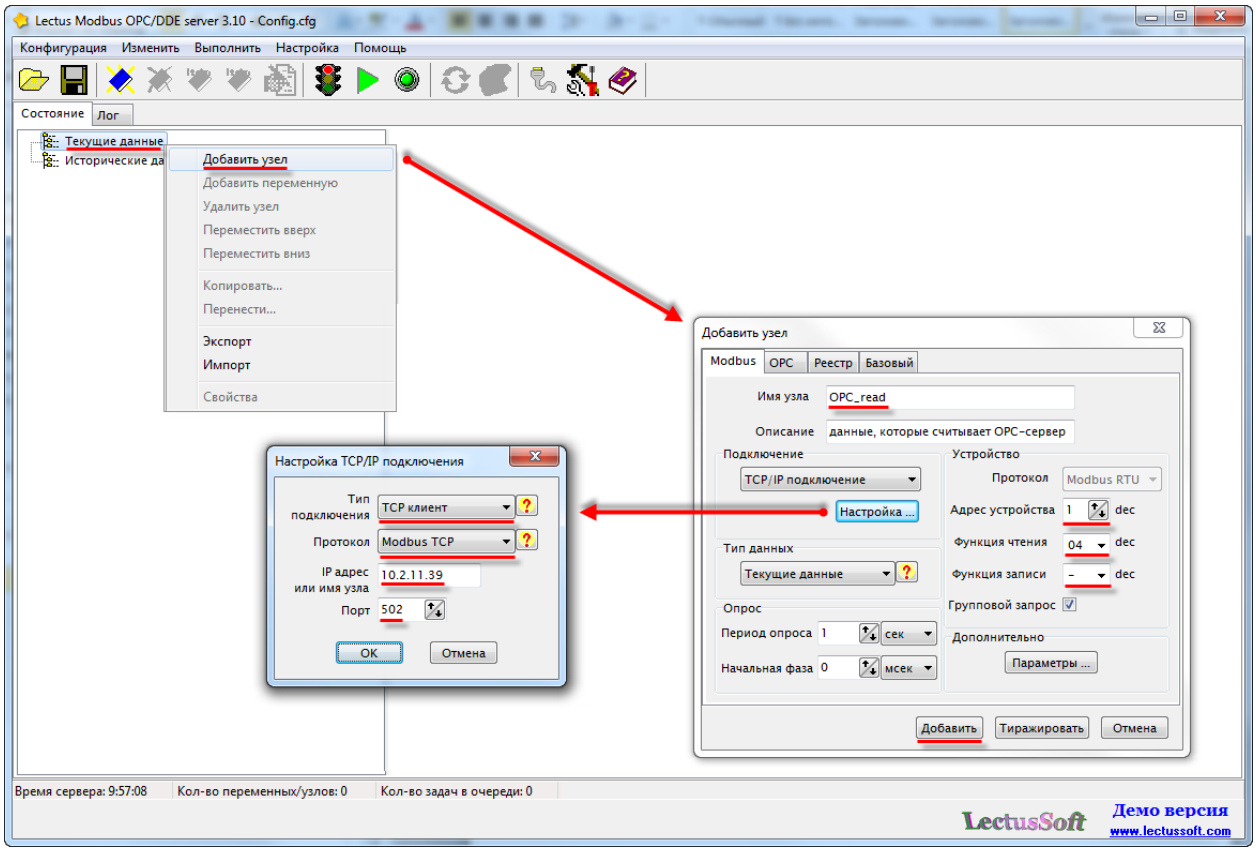


Рис. 4.4.1. Добавление узла **OPC_read**

3. Нажмите **ПКМ** на узел **OPC_read** и добавьте три переменные. Настройки переменных приведены ниже:

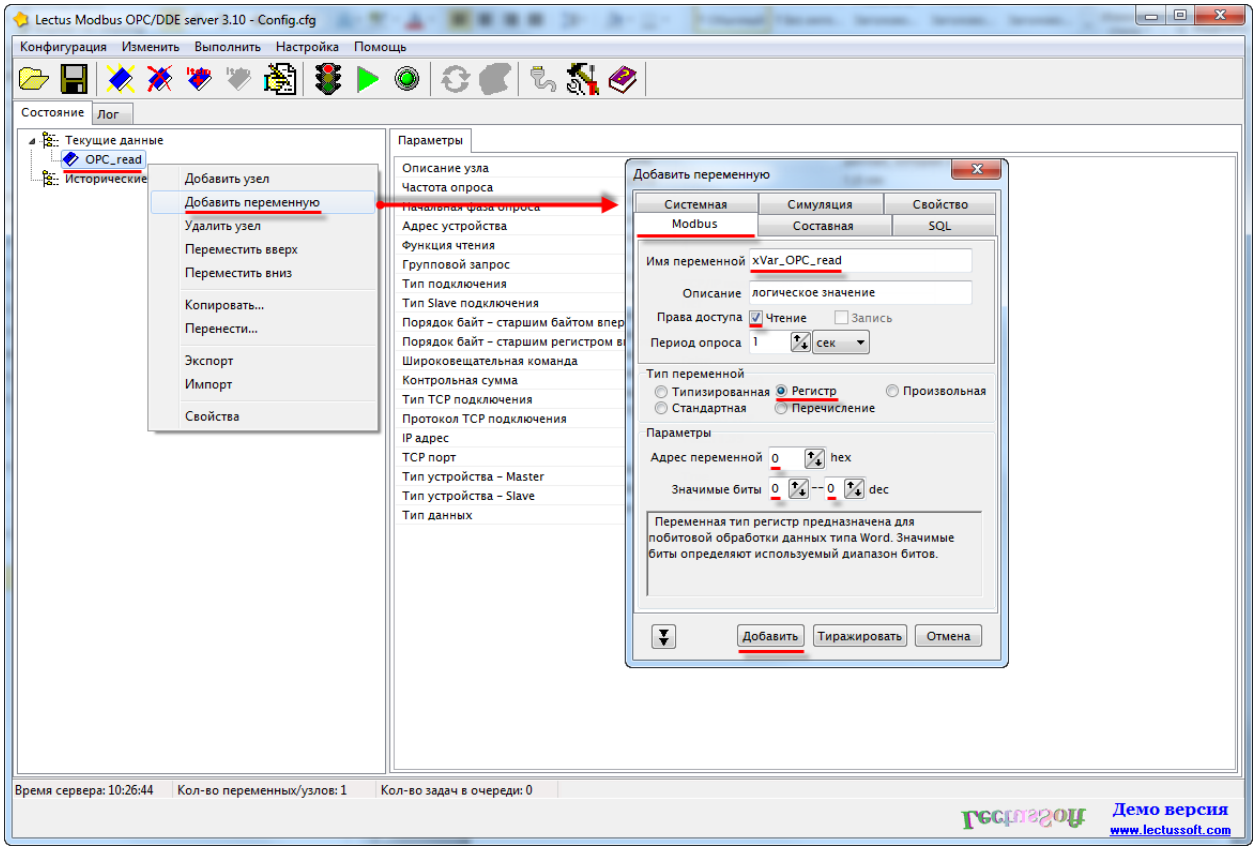


Рис. 4.4.2. Добавление и настройка переменной **xVar_OPC_read**

В данном примере мы работаем с нулевым битом нулевого регистра. При необходимости считать первый бит нулевого регистра необходимо выбрать **значимые биты** 1—1, для второго бита 2—2 и т.д. Напомним, что подобный подход используется по причине того, что **Modbus Slave,** созданный стандартными средствами конфигурирования, на СПК с прошивками **3.9xx** не поддерживает работу с битовыми функциями.

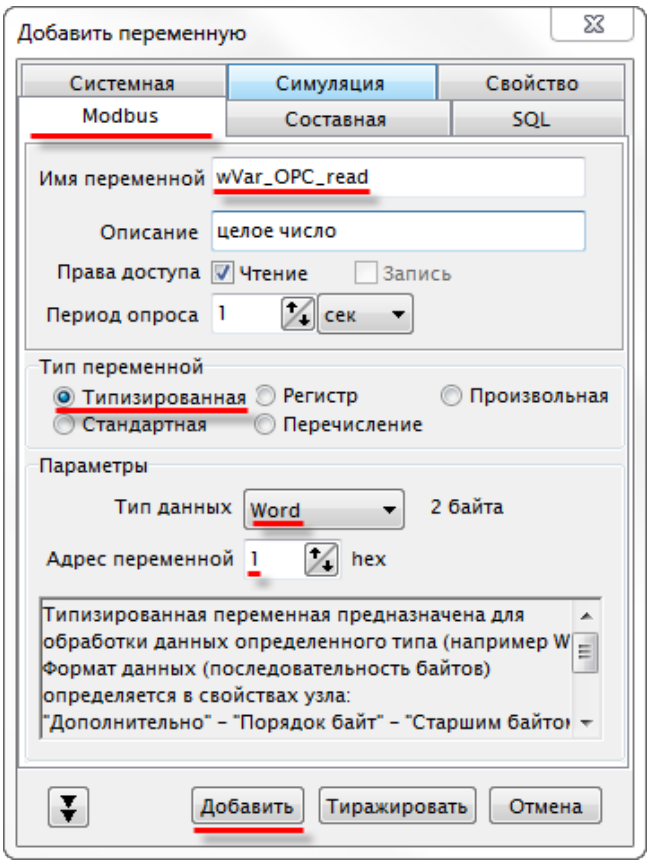


Рис. 4.4.3. Настройки переменной **wVar_OPC_read**

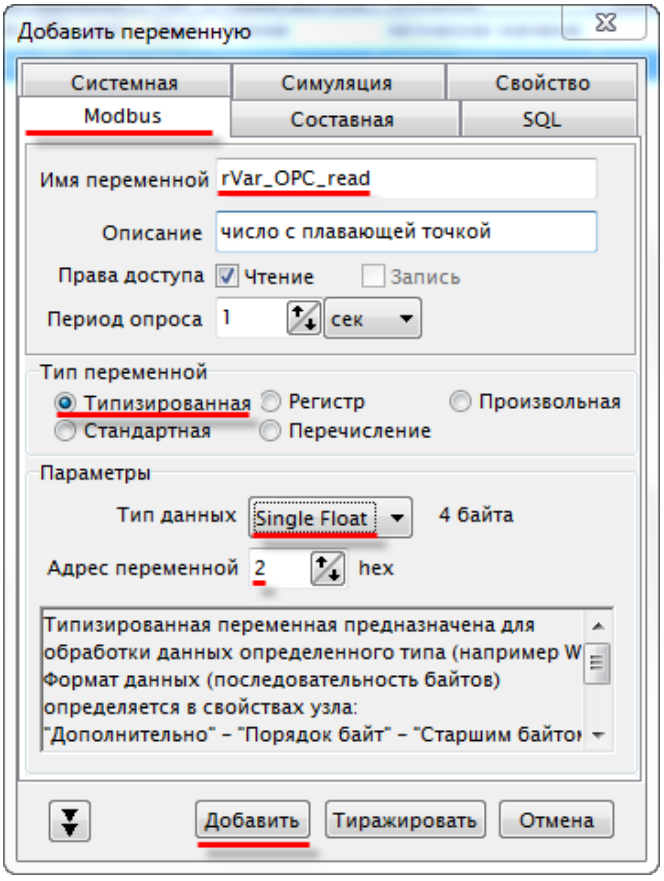


Рис. 4.4.4. Настройки переменной **rVar_OPC_read**

4. Нажмите **ПКМ** на вкладку **Текущие данные** и добавьте узел с названием **OPC_write**. В его настройках укажите используемый тип подключения (**TCP клиент**), протокол (**Modbus TCP**), IP-адрес контроллера (в соответствии с п. 4.3.1 пп. 5), порт и адрес устройства (в соответствии с п. 4.3.1 пп. 6), а также функции **Modbus**, используемые для чтения данных – **03** (**Read Holding Registers**) и записи данных – **16** (**Preset Multiple Registers**).

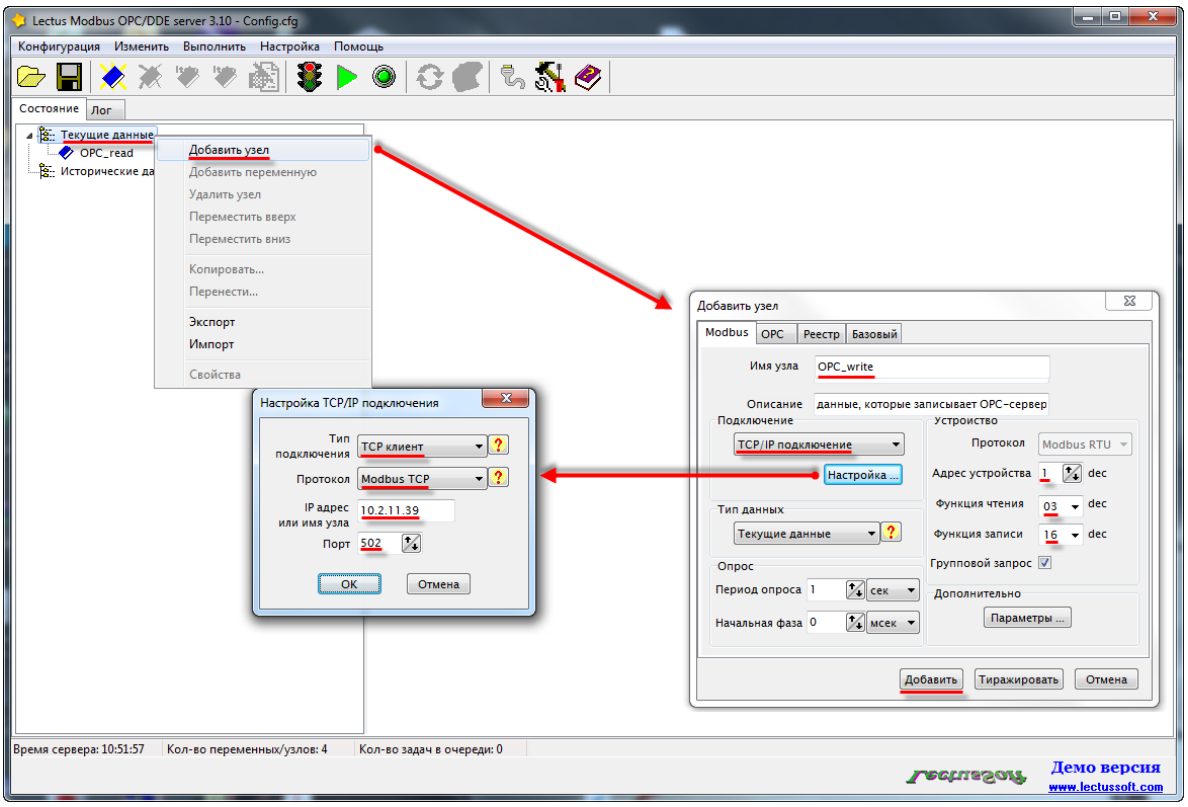


Рис. 4.4.5. Добавление узла **OPC_write**

5. Нажмите **ПКМ** на узел **OPC_write** и добавьте три переменные. Настройки переменных приведены ниже:

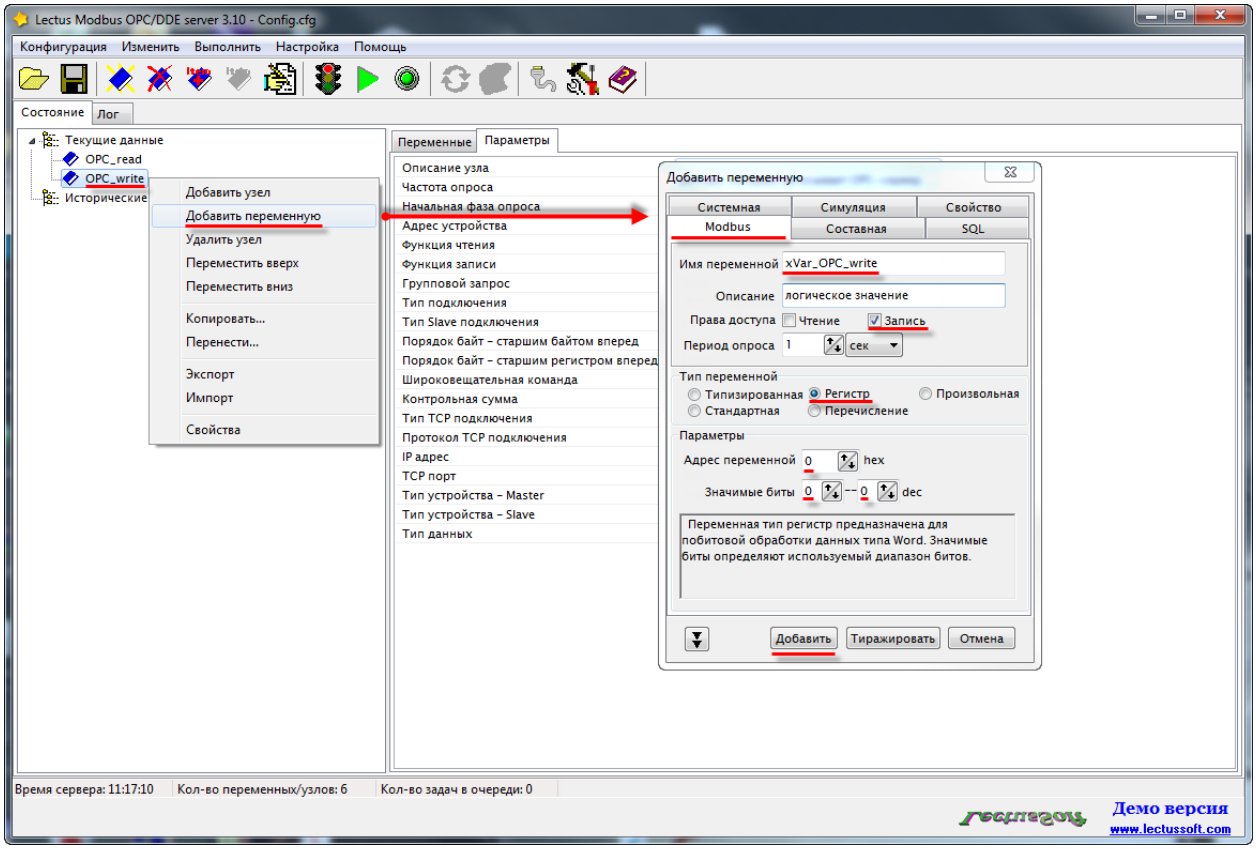


Рис. 4.4.6. Добавление и настройки переменной **xVar_OPC_write**

В данном примере мы работаем с нулевым битом нулевого регистра. При необходимости записать первый бит нулевого регистра необходимо выбрать значимые биты 1—1, для второго бита 2—2 и т.д. Напомним, что подобный подход используется по причине того, что **Modbus Slave,** созданный стандартными средствами конфигурирования, на СПК с прошивками **3.9xx** не поддерживает работу с битовыми функциями.

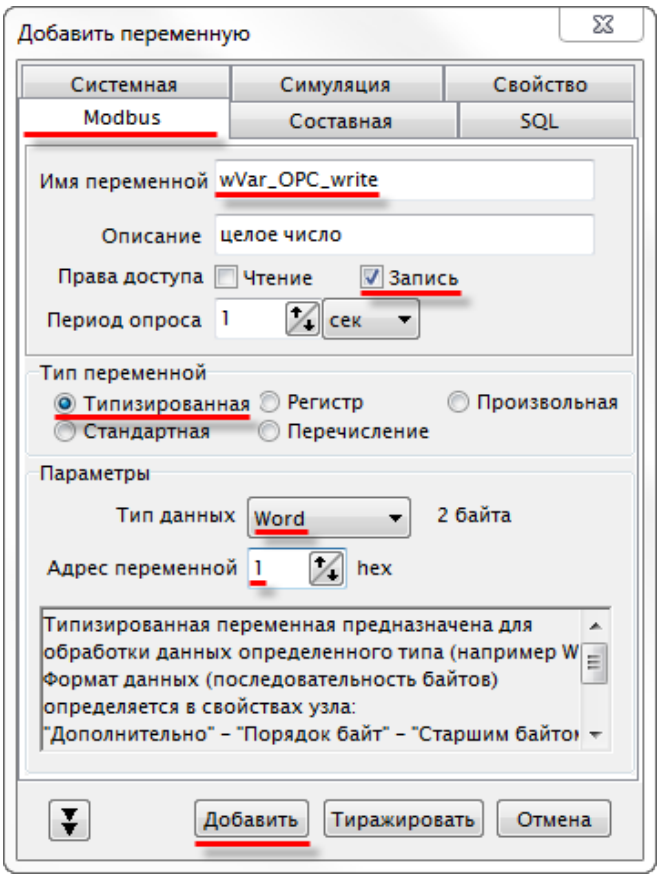


Рис. 4.4.7. Настройки переменной **wVar_OPC_write**

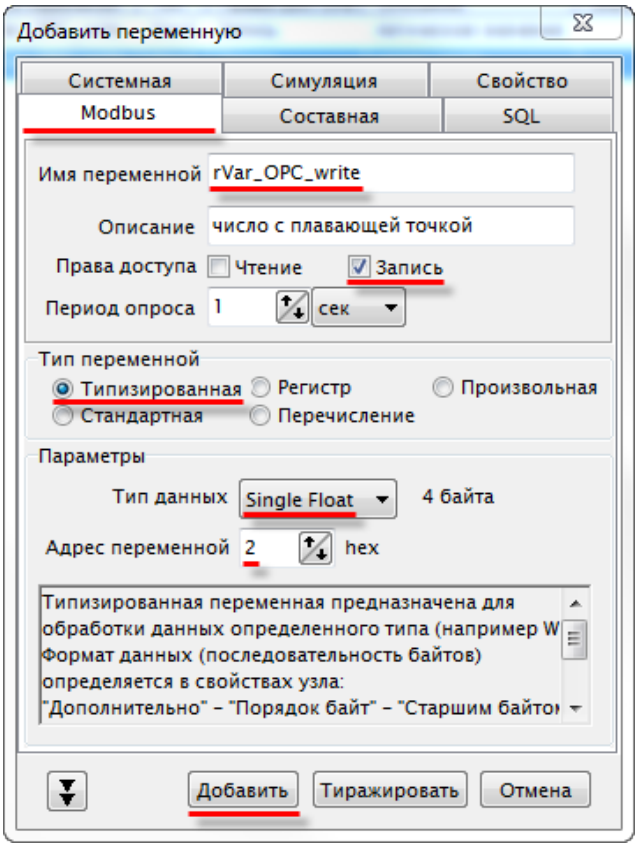


Рис. 4.4.8. Настройки переменной **rVar_OPC_write**

5. После добавления и настройки переменных сохраните конфигурацию OPC-сервера:

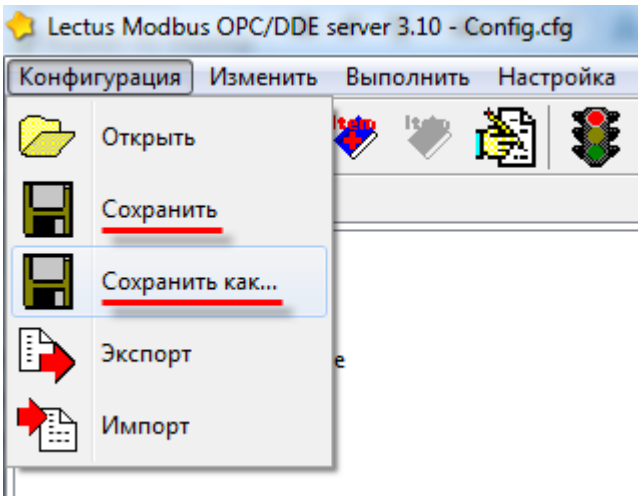


Рис. 4.4.9. Сохранение конфигурации OPC-сервера

Закройте OPC-сервер. Запускать его не требуется, поскольку SCADA-система производит этот процесс автоматически.

Теперь вам необходимо загрузить проект, созданный в п. 4.3.1 в СПК и убедиться, что контроллер находится в одной локальной сети с OPC-сервером. После этого можно переходить к п. 4.6.

Созданная в пункте конфигурация доступна для скачивания: Example_MasterOPC_LectusOPC_OwenOPC.zip

4.5. ОВЕН OPC (новый)

4.5.1. Настройка СПК

1. Создайте проект согласно п. 4.3.1.

4.5.2. Настройка OPC-сервера

1. Установите и запустите ОВЕН OPC (новый).

2. Нажмите **ПКМ** на узел **Сервер** и добавьте узел с названием по умолчанию (**Узел1**). В его настройках укажите используемый протокол (**Modbus TCP**).

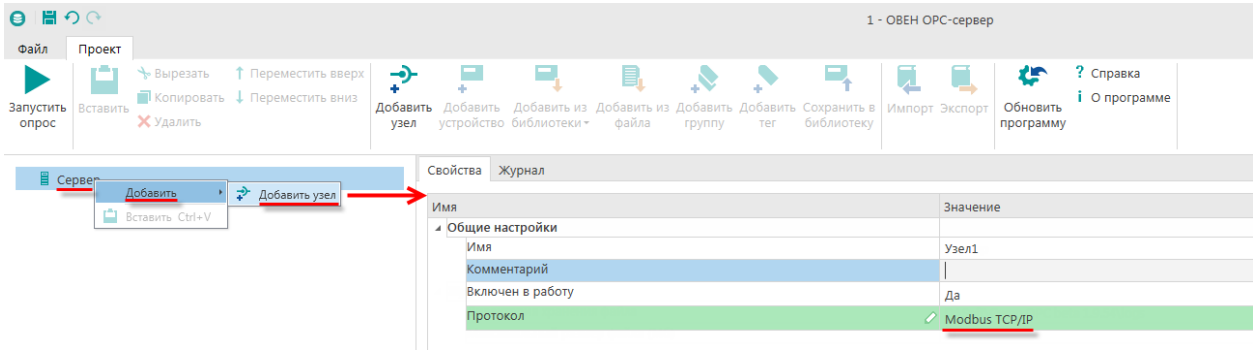


Рис. 4.5.1. Добавление и настройка узла

3. Нажмите **ПКМ** на узел **Узел1** и устройство с названием **СПК**. В его настройках укажите сетевые параметры (**IP-адрес** и **порт**). Сетевые настройки должны соответствовать настройкам СПК (см. п. 4.3.1, пп. 4-5).

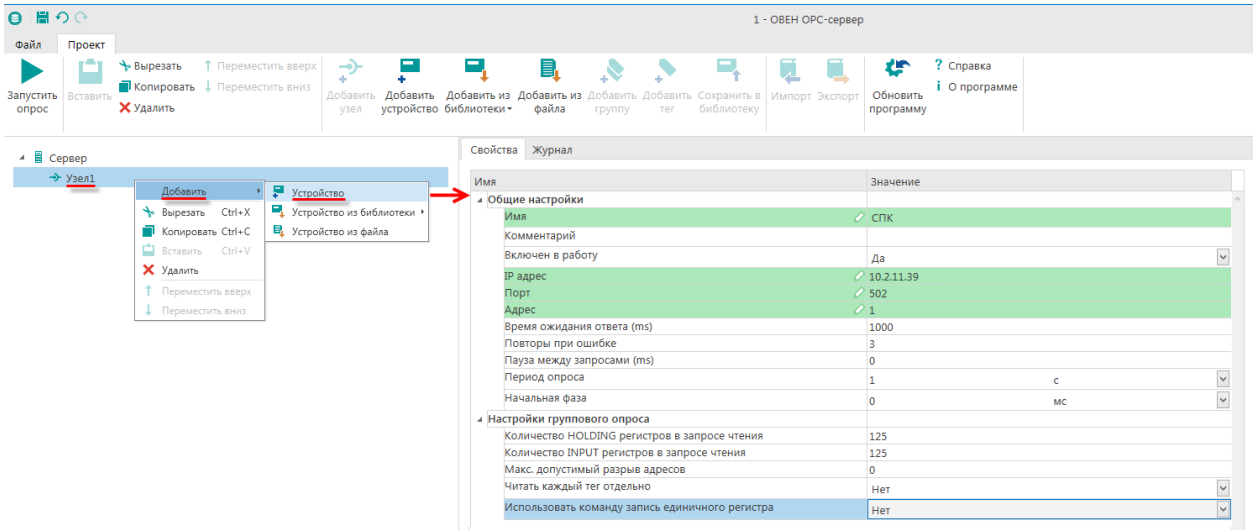


Рис. 4.5.2. Добавление коммуникационного узла

4. Нажмите **ПКМ** на устройство СПК и добавьте 6 тегов. Число тегов соответствует числу переменных, считываемых/записываемых в СПК. Настройки тегов приведены ниже.

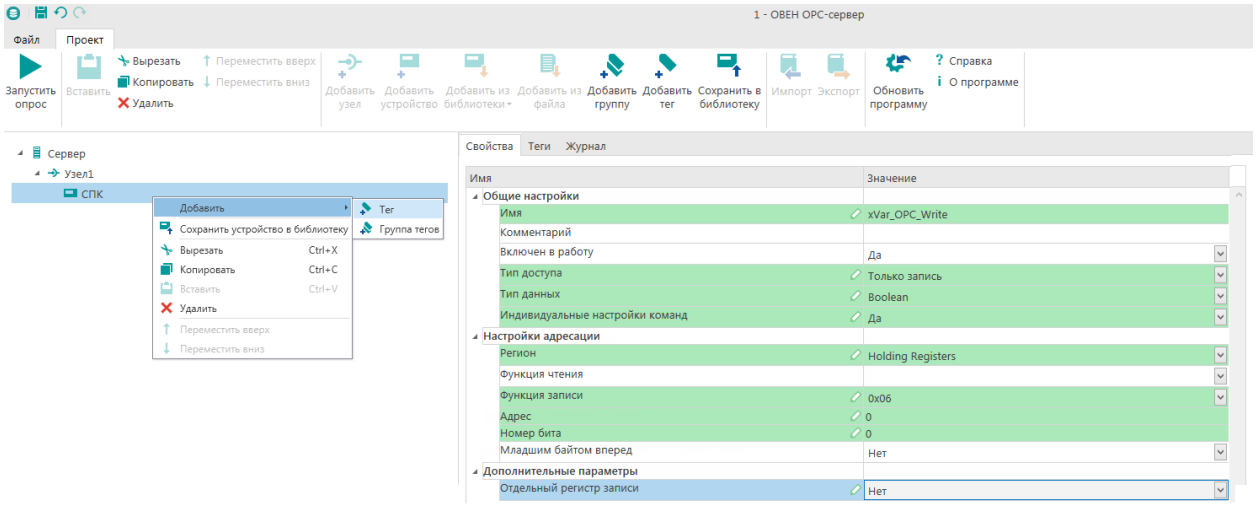


Рис. 4.5.3. Добавление и настройка тега **xVar_OPC_write**

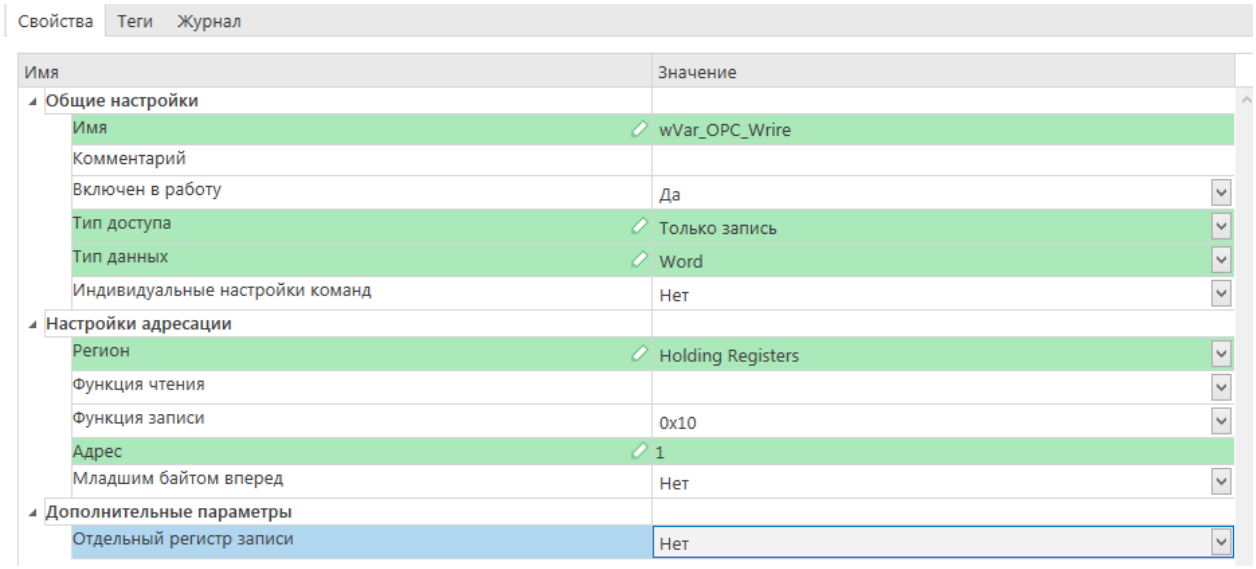


Рис. 4.5.4. Настройки тега **wVar_OPC_write**

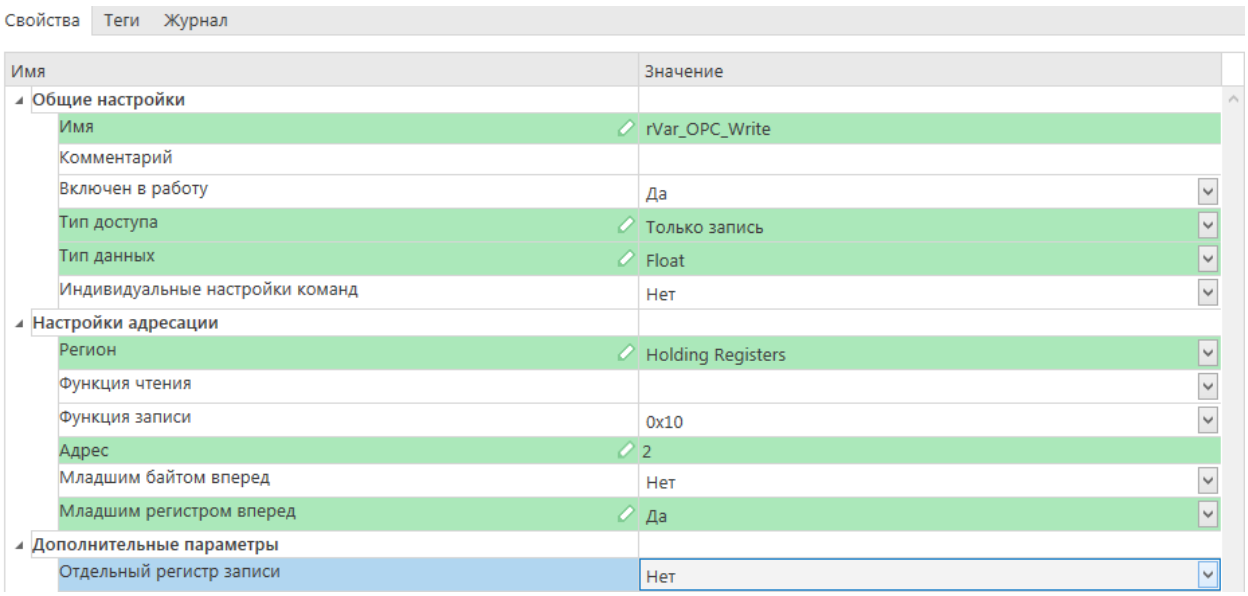


Рис. 4.5.5. Настройки тега **rVar_OPC_write**

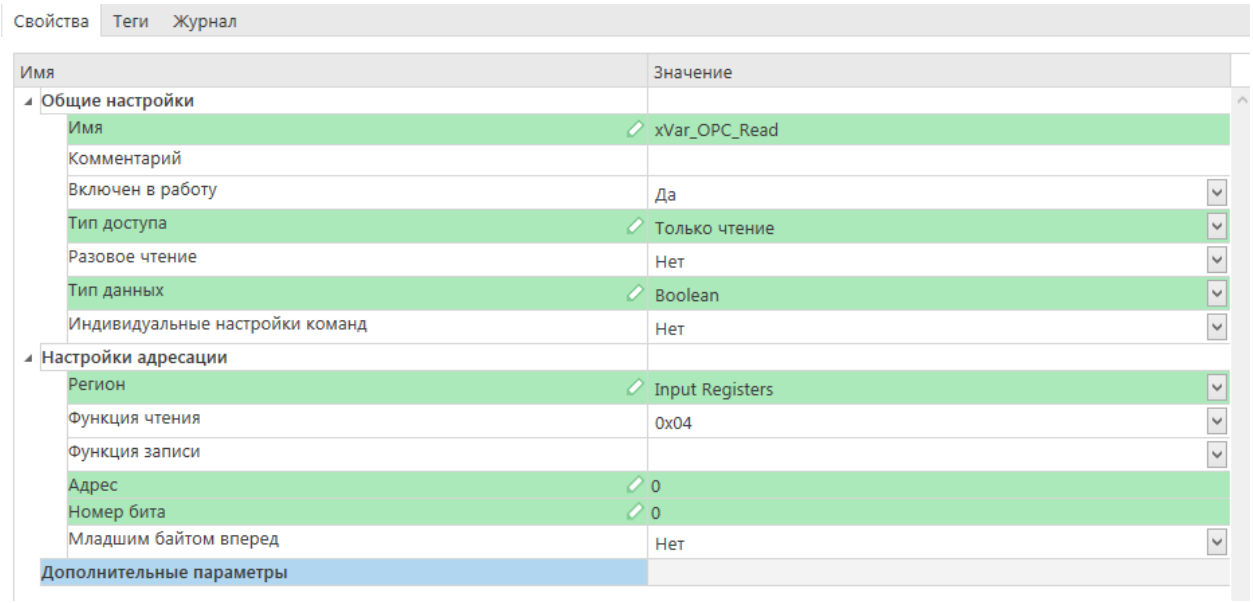


Рис. 4.5.6. Настройки тега **xVar_OPC_read**

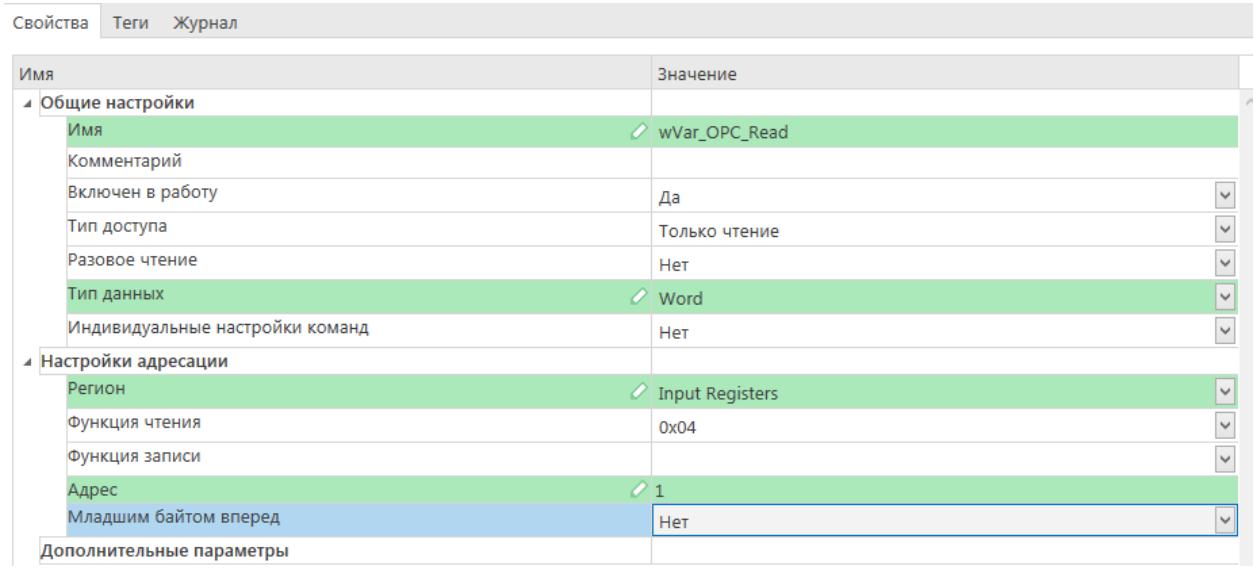


Рис. 4.5.7. Настройки тега **wVar_OPC_read**

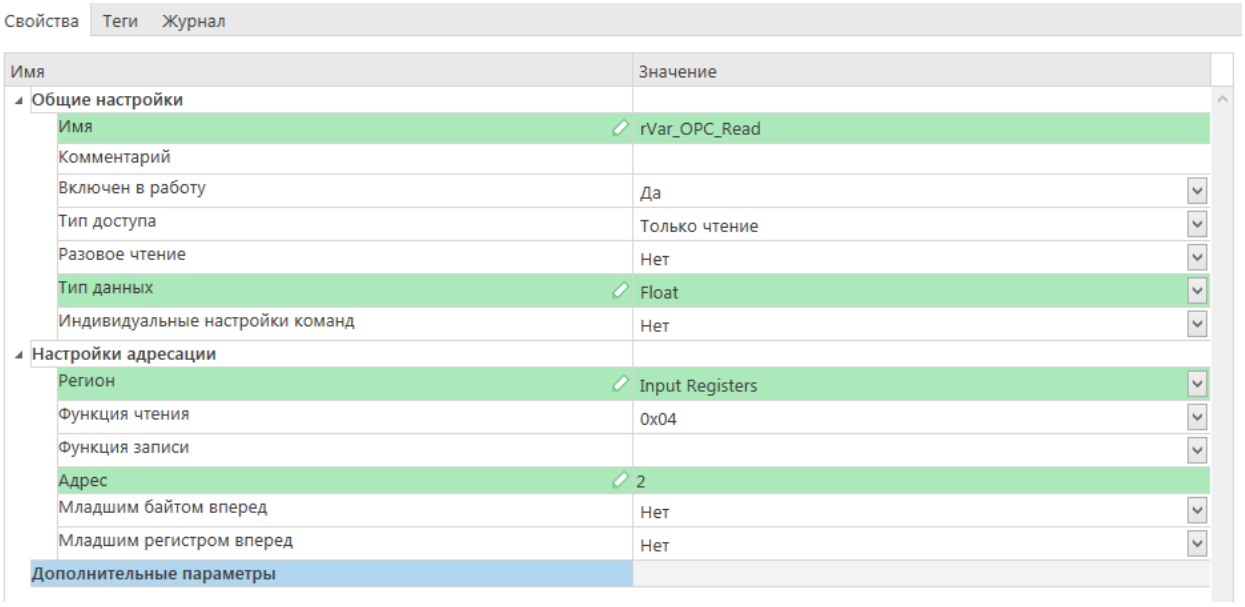


Рис. 4.5.8. Настройки тега **rVar_OPC_read**

5. После добавления и настройки тегов сохраните конфигурацию OPC-сервера:

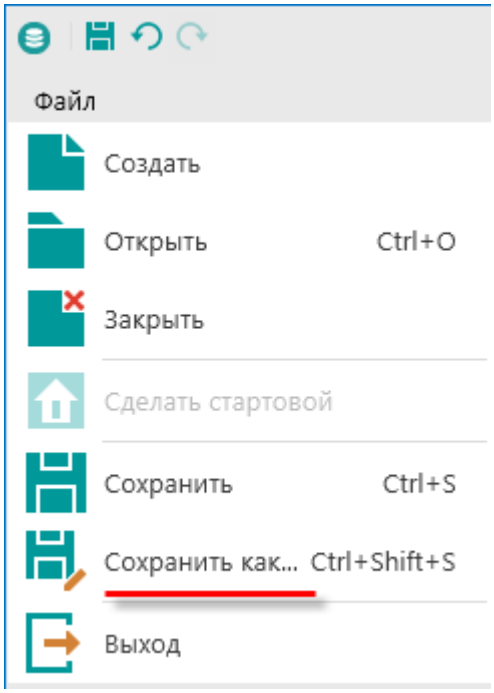


Рис. 4.5.9. Сохранение конфигурации OPC-сервера

Закройте OPC-сервер. Запускать его не требуется, поскольку SCADA-система производит этот процесс автоматически.

Теперь вам необходимо загрузить проект, созданный в п. 4.3.1 в СПК и убедиться, что контроллер находится в одной локальной сети с OPC-сервером. После этого можно переходить к п. 4.6.

Созданная в пункте конфигурация доступна для скачивания: Example_MasterOPC_LectusOPC_OwenOPC.zip

4.6. Подключение OPC-сервера к SCADA-системе

После настройки OPC-сервера необходимо подключить его к SCADA-системе. Покажем, как это сделать на примере системы MasterSCADA.

Запустите **MasterSCADA**. Нажмите **ПКМ** на узел **Система** и добавьте узел **Компьютер**:

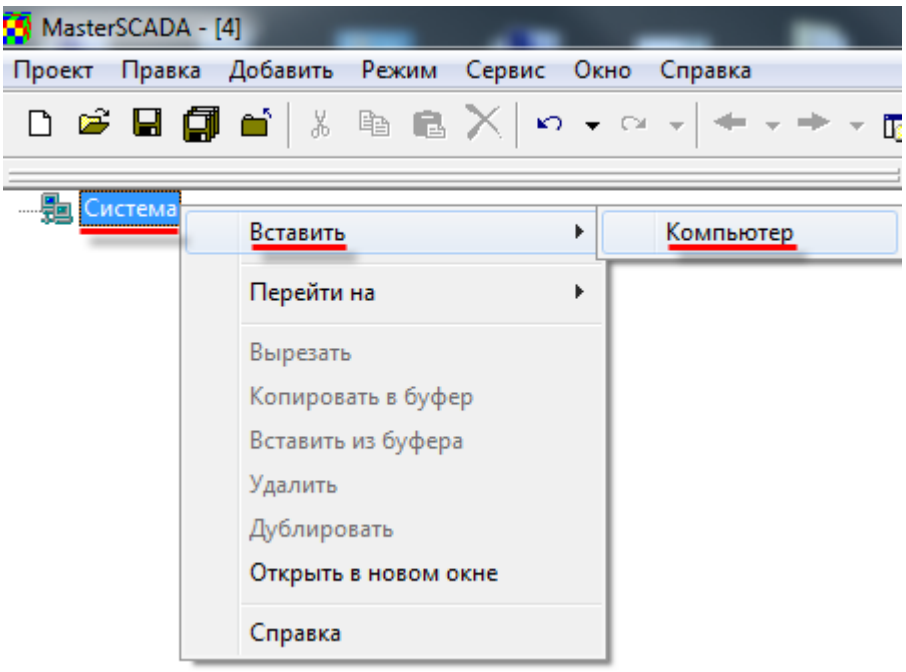


Рис. 4.6.1. Добавление узла **Компьютер**

Нажмите **ПКМ** на узел **Компьютер** и выберите команду **Вставить OPC-сервер**:

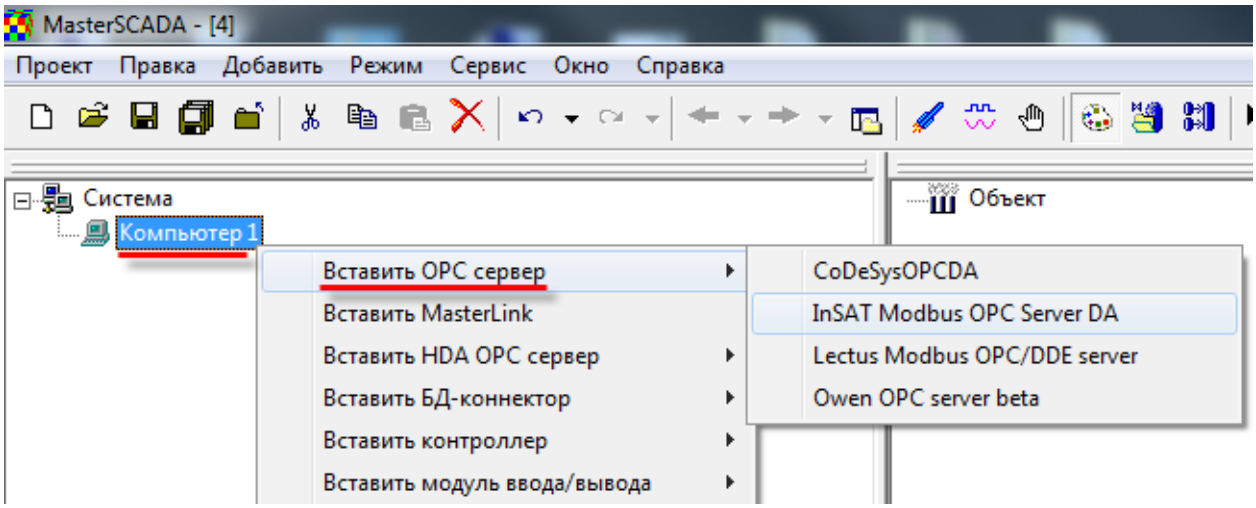


Рис. 4.6.2. Выбор OPC-сервера

Если нужный OPC-сервер не отображается в списке, тогда необходимо выполнить команду **Поиск OPC DA серверов**:

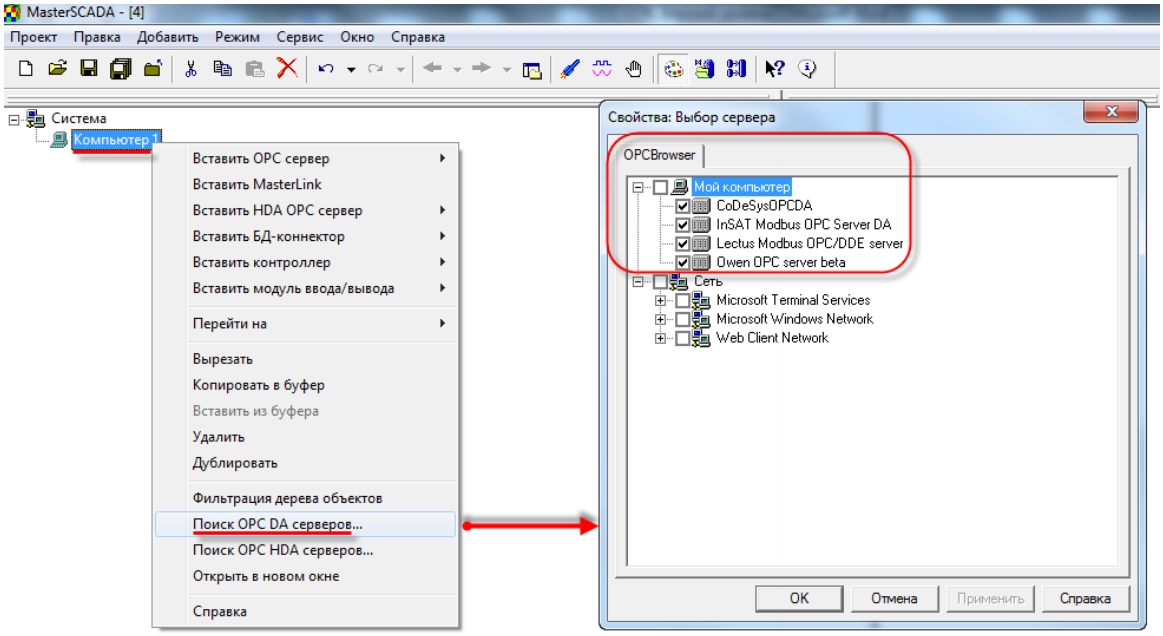


Рис. 4.6.3. Поиск установленных OPC-серверов

После добавления OPC-сервера, нажмите **ПКМ** на его название и выберите команду **Вставить OPC переменные**. В появившемся диалоговом окне необходимо пометить галочками нужные переменные. Можно также выбрать папку – в этом случае, в проект будут добавлены все переменные данной папки.

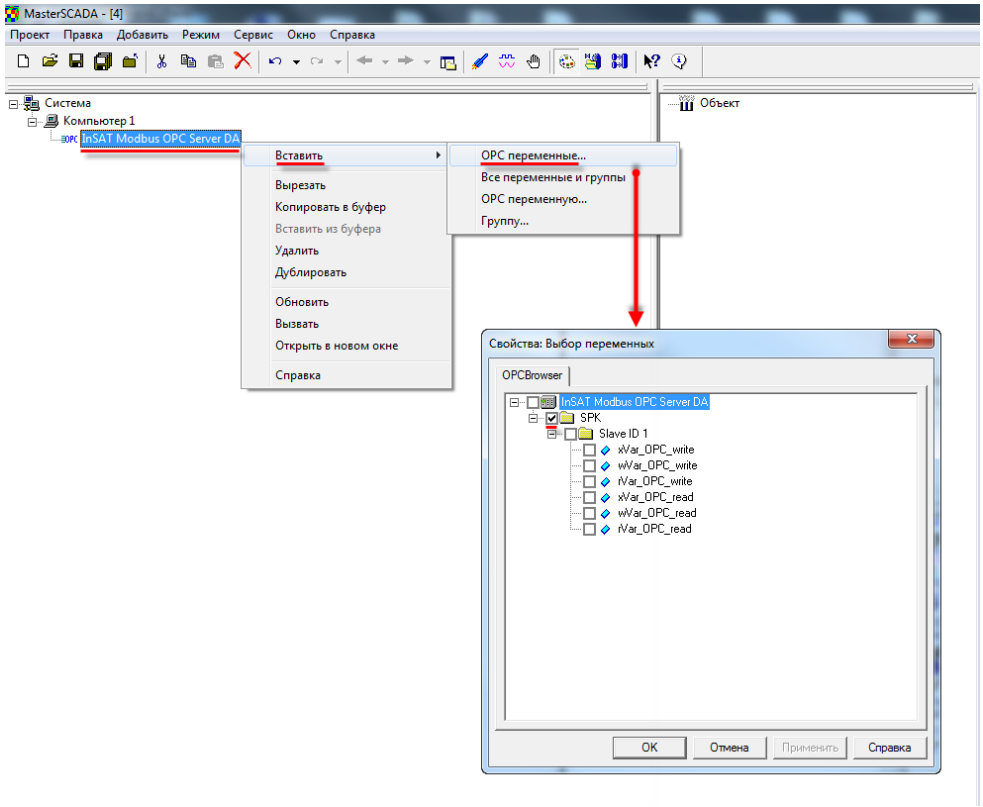


Рис. 4.6.4. Добавление переменных OPC-сервера

Запустите проект на исполнение. Подразумевается, что в СПК загружен и запущен нужный проект, при этом настроена связь между СПК и ПК, на котором установлен OPC-сервер.

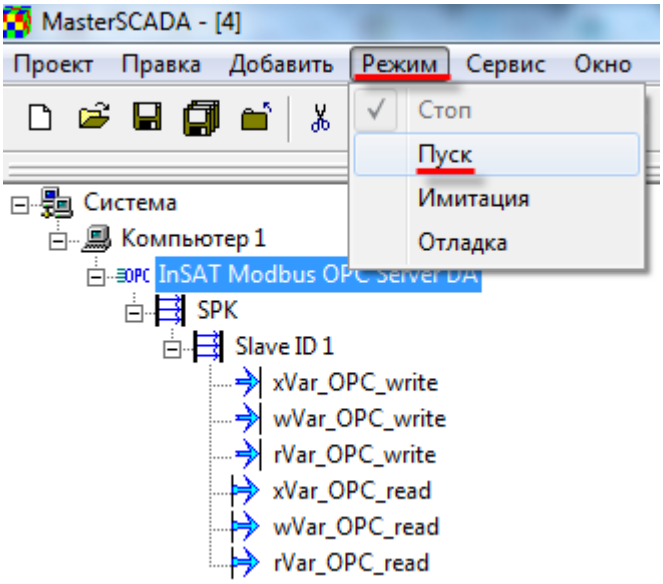


Рис. 4.6.5. Запуск проекта **MasterSCADA** на исполнение

В редакторе **CODEYS** изменяйте значения **OPC_read** переменных и наблюдайте соответствующие изменения в **MasterSCADA**.

PLC_PRG

Device.Application.PLC_PRG

Выражение	Тип	Значение	Подготовленное ...	Адрес	Комментарий
xVar_OPC_read	BOOL	TRUE			логическое значение
wVar_OPC_read	WORD	11			целое число
_rVar_OPC_read	Real_Word				число с плавающей точкой
rRealValue	REAL	11.11			
awModbusReal	ARRAY [0..1] OF WORD				
xVar_OPC_write	BOOL	FALSE			логическое значение
wVar_OPC_write	WORD	0			целое число
_rVar_OPC_write	Real_Word				число с плавающей точкой

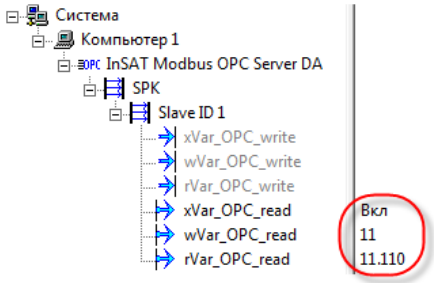


Рис. 4.6.6. Считывание данных из контроллера в **MasterSCADA**

В **MasterSCADA** изменяйте значения **OPC_write** переменных и наблюдайте соответствующие изменения в редакторе **CODESYS.**

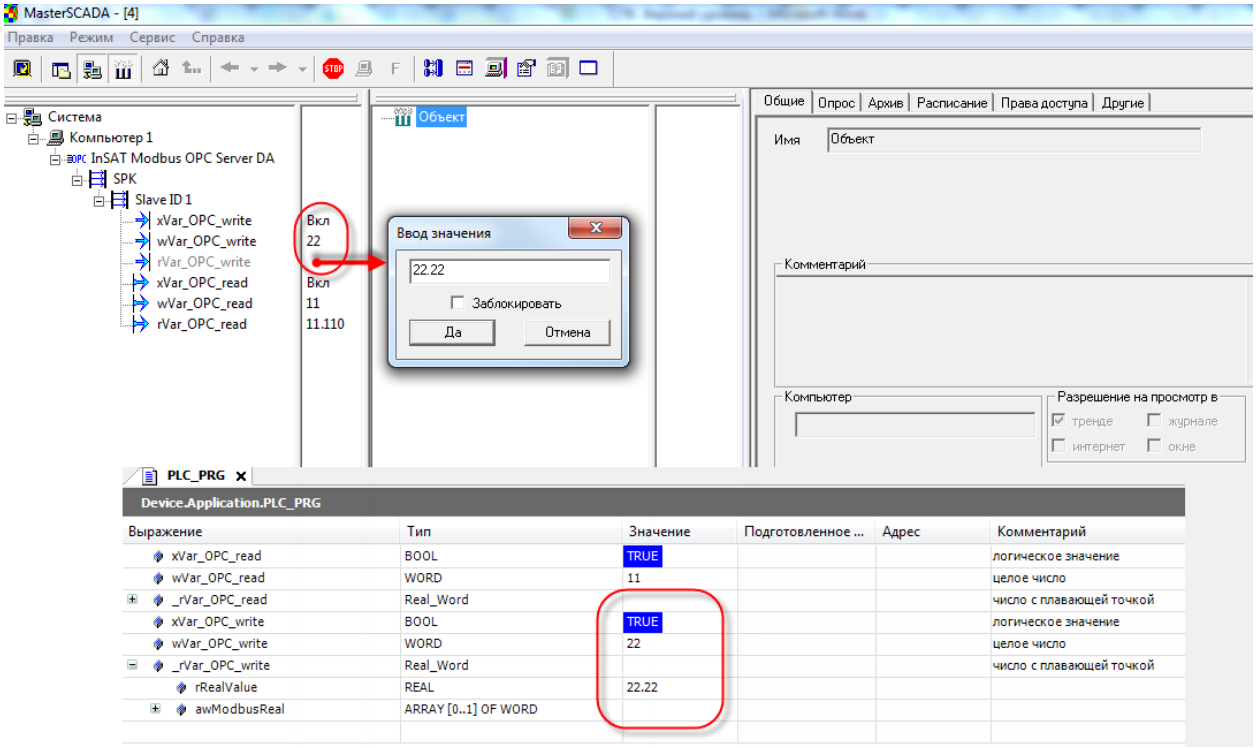


Рис. 4.6.7. Запись данных из **MasterSCADA** в контроллер

5. Облачный сервис OwenCloud

Облачный сервис OwenCloud применяется для удаленного мониторинга, управления и хранения архивов данных приборов, используемых в системах автоматизации. Подключение приборов к сервису осуществляется по интерфейсам **RS-485** (с помощью специальных сетевых шлюзов) или **Ethernet** (в этом случае требуется подключение приборов к сети с доступом к Интернету).

Контроллеры **СПК1xx** могут быть подключены к сервису **OwenCloud** через сетевые шлюзы линейки Пх210. На контроллере в этом случае достаточно создать и настроить обычный **Modbus RTU Slave**.

Для подключения контроллеров **СПК207** к сервису **OwenCloud** не требуется наличие сетевых шлюзов линейки Пх210. Доступ к облачному сервису осуществляется через подключение контроллера к локальной сети с доступом в Интернет. Для передачи данных используется протокол **Modbus TCP**.

Функционал доступен начиная с версии встроенного ПО контроллера **5.480** и требует установки дополнительного компонента в **CODESYS V3.5.**

Встроенное ПО и инструкции по его обновлению доступны на сайте ОВЕН в разделе CODESYS v.3/Сервисное ПО для СПК2xx. Компонент связи с OwenCloud для CODESYS 3.5 доступен в разделе Codesys v.3/Библиотеки CODESYS.

1. В **CODESYS 3.5** откройте **Менеджер пакетов** (вкладка **Инструменты** на панели управления) и установите компонент **OwenCloud TCP Slave Device**.

2. Создайте проект для СП207. В программе **PLC_PRG** объявите следующие переменные:

PLC_PRG

```
1  PROGRAM PLC_PRG
2  VAR
3      wInputReg:      WORD;
4      wHoldingReg:    WORD;
5  END_VAR
```

Рис. 5.1. Объявление переменных в программе **PLC_PRG**

3. Добавьте в проект компонент **Ethernet** версии **3.4.2.0**.

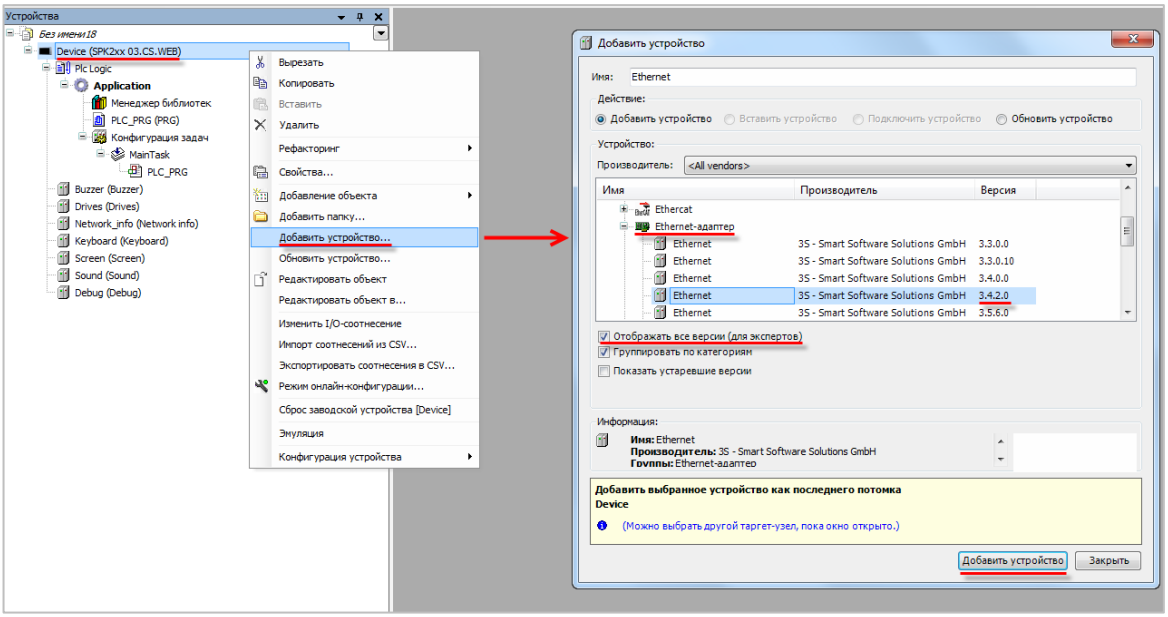


Рис. 5.2. Добавление компонента **Ethernet**

В настройках компонента на вкладке **Конфигурация Ethernet** укажите сетевые параметры вашего контроллера:

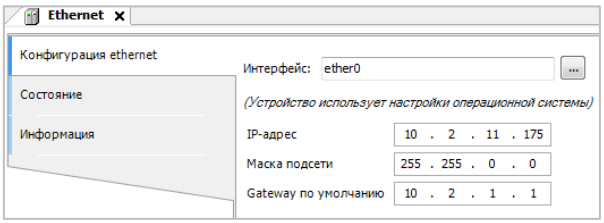


Рис. 5.3. Настройки компонента **Ethernet**

В компонент **Ethernet** добавьте устройство **OwenCloud TCP Slave Device**:

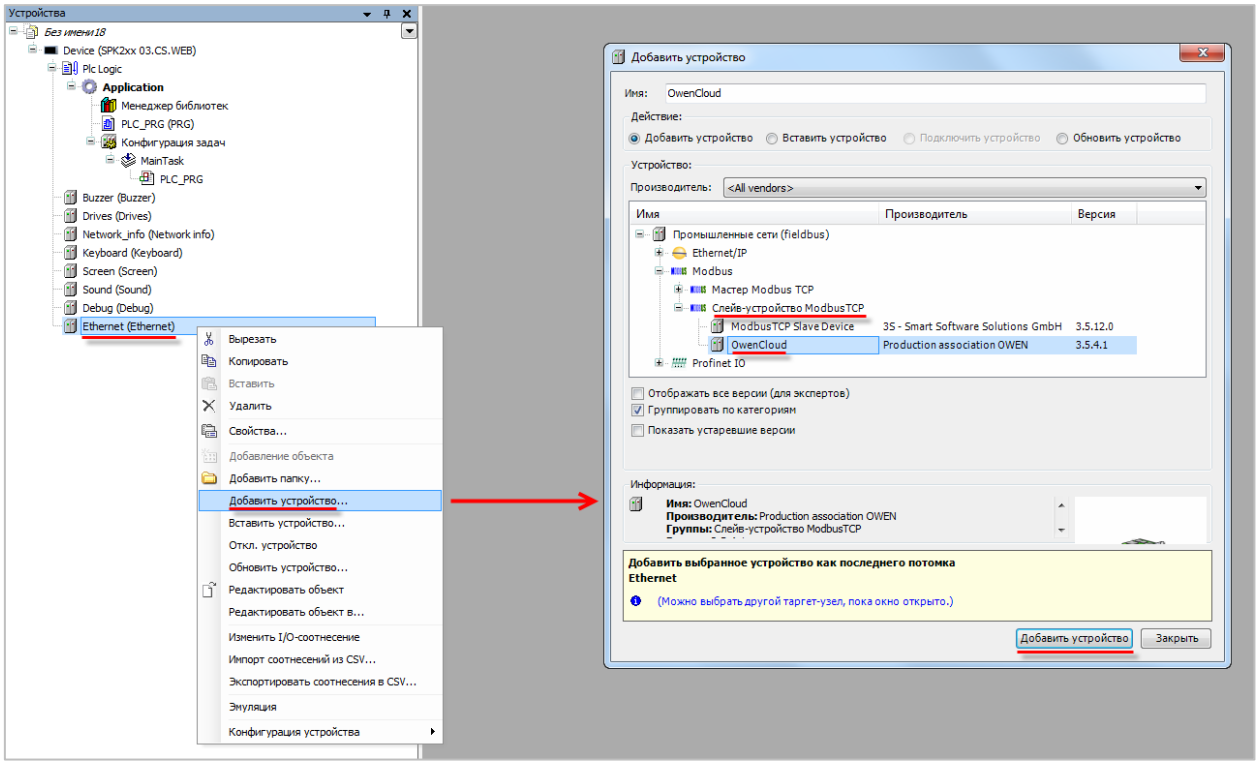


Рис. 5.4. Добавление компонента **OwenCloud TCP Slave Device**

В настройках компонента на вкладке **Страница конфигурации** снимите галочку **Таймаут** и укажите TCP-порт контроллера, который будет использоваться для связи с облачным сервисом (например, **1502**). Кроме того, можно указать количество доступных input- и holding-регистров Modbus.

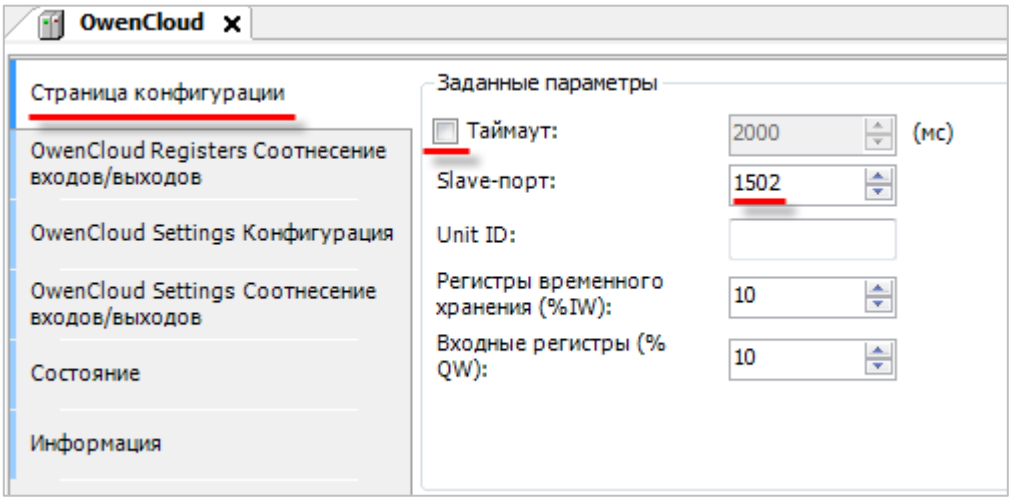


Рис. 5.5. Настройки компонента **OwenCloud TCP Slave Device,** вкладка **Страница конфигурации**

На вкладке **OwenCloud Settings Конфигурация** необходимо повторно указать порт и ввести токен прибора, генерируемый при добавлении прибора в сервис **OwenCloud**. На данном этапе токен отсутствует – он будет получен в пп. 6.

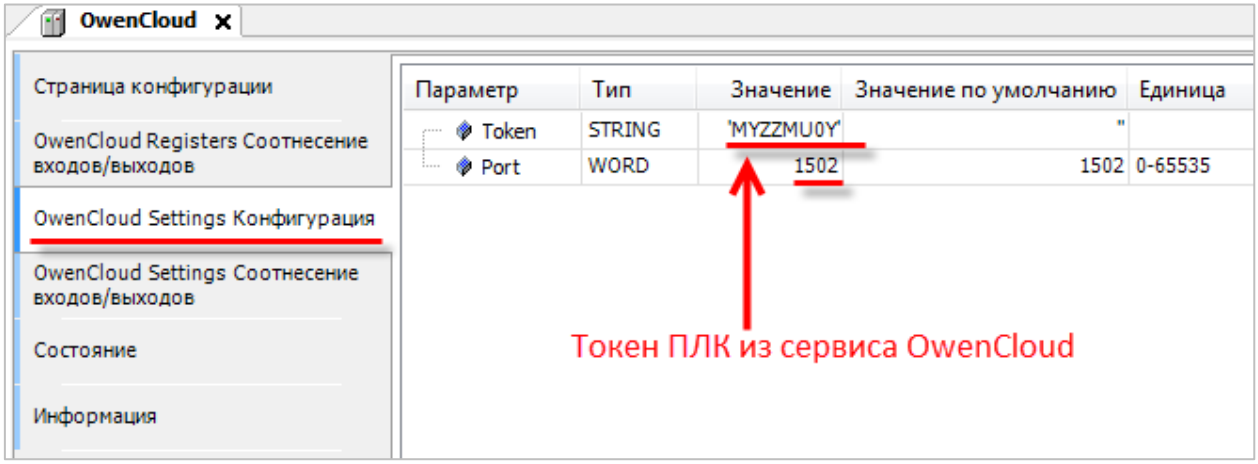


Рис. 5.6. Настройки компонента **OwenCloud TCP Slave Device,** вкладка **OwenCloud Settings Конфигурация**

На вкладке **OwenCloud Registers Соотнесение входов/выходов** привяжите переменные программы к регистрам Modbus. Канал **Inputs** содержит holding-регистры, канал **Outputs** – input-регистры. Адресация для каждой области памяти Modbus является независимой и ведется с нулевого регистра.

Таким образом, в контроллере будет сформирована следующая карта регистров:

Табл. 5.1. Карта регистров для СПК207

Имя переменной	Тип	Область памяти	Адрес регистра (назначается автоматически)
wHoldingReg	WORD	Holding-регистры	0
wInputReg	WORD	Input-регистры	0

Для параметра **Всегда обновлять переменные** следует поставить значение **Вкл. 2 (Всегда в задаче цикла шины)**.

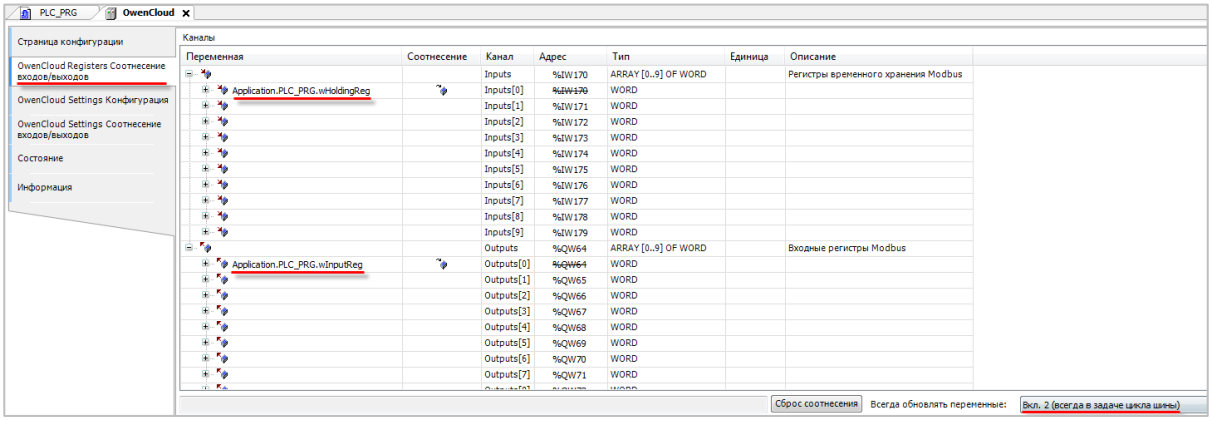
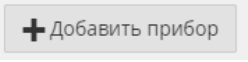


Рис. 5.7. Настройки компонента **OwenCloud TCP Slave Device,** вкладка **OwenCloud Registers Соотнесение входов/выходов**

Обратите внимание на следующие моменты:

- работа с битами (функции 1, 2, 5, 15) не поддерживается;
- holding-регистры не могут быть изменены из программы контроллера – записать их значение может только Master-устройство;
- каналы Slave-устройства в CODESYS имеют тип **WORD**. При необходимости передачи данных других типов (например, **REAL**) необходимо преобразовать их в последовательность регистров типа WORD – например, с помощью объединений.

4. Зайдите на главную страницу сервиса **OwenCloud**. Если вы еще не зарегистрированы в сервисе – необходимо пройти процедуру регистрации.

5. Перейдите на страницу Администрирование, откройте вкладку **Приборы** и нажмите кнопку **Добавить прибор** (

).

Укажите следующие настройки:

- **Идентификатор** – введите MAC-адрес ПЛК (указан на корпусе ПЛК);
- **Тип прибора** – выберите тип **Произвольное устройство Modbus**;
- **Заводской номер** – укажите заводской номер прибора (заполнять необязательно);
- **Название прибора** – введите название прибора;
- **Категории** – выберите категории, к которым будет принадлежать прибор;
- **Часовой пояс** – укажите часовой пояс, в котором находится прибор.

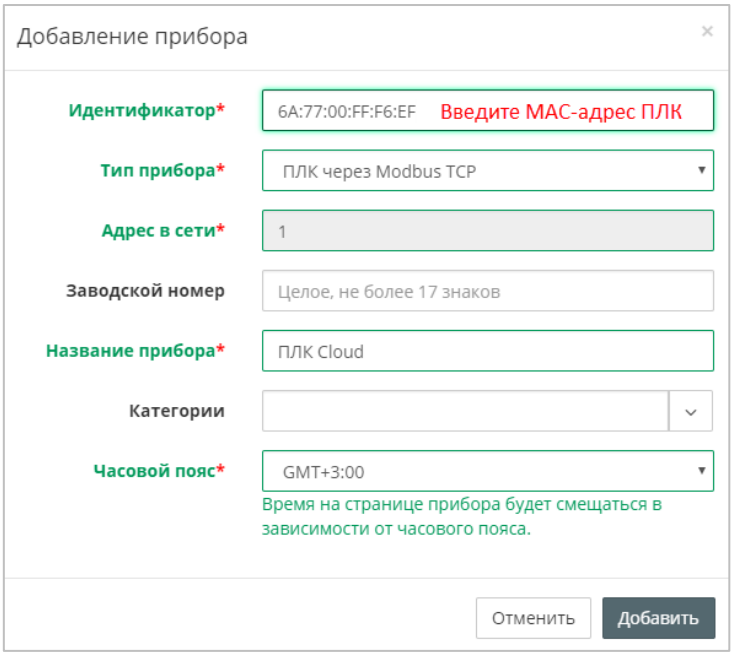


Рис. 5.8. Окно добавления прибора

Нажмите кнопку **Добавить**.

6. На вкладке **Общие/Общие настройки** будет отображаться токен ПЛК. Скопируйте его и введите в **CODESYS 3.5** в настройках компонента **OwenCloud TCP Slave Device** на вкладке **OwenCloud Setting Конфигурация:**

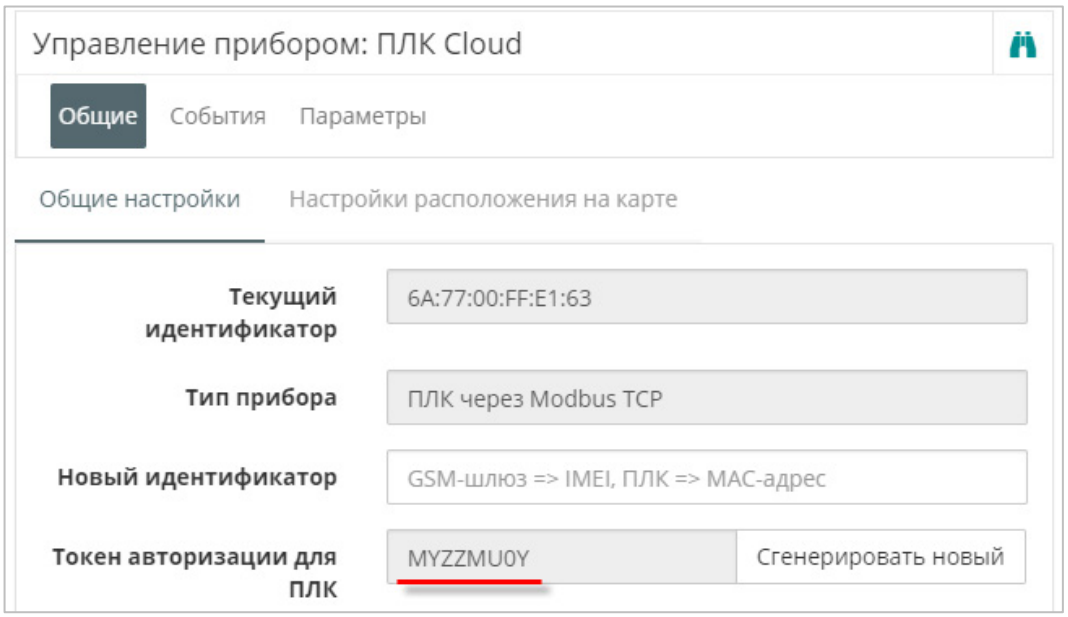


Рис. 5.9. Копирование токена из **OwenCloud**

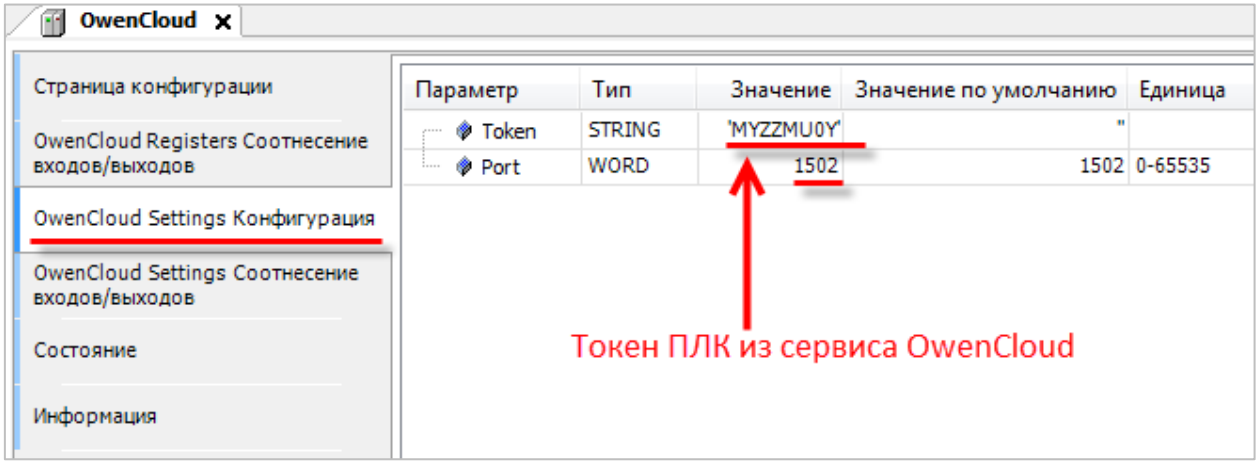


Рис. 5.10. Ввод сетевых настроек прибора в **OwenCloud**

7. На вкладке **Параметры/Настройки параметров Modbus** добавьте параметры в соответствии с рис. 5.11.

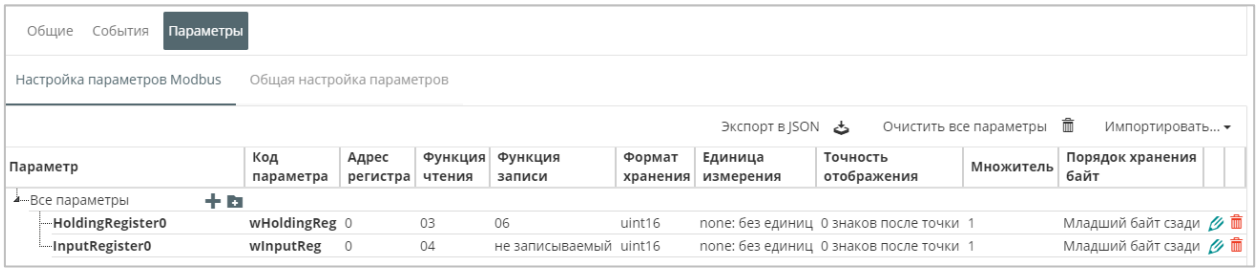


Рис. 5.10.11. Настройка параметров Modbus

8. Нажмите на пиктограмму  , чтобы перейти к просмотру значений параметров прибора. Измените значения переменных в CODESYS и наблюдайте соответствующие изменения в **OwenCloud**. При необходимости изменения значений из облачного сервиса перейдите на вкладку Запись параметров.

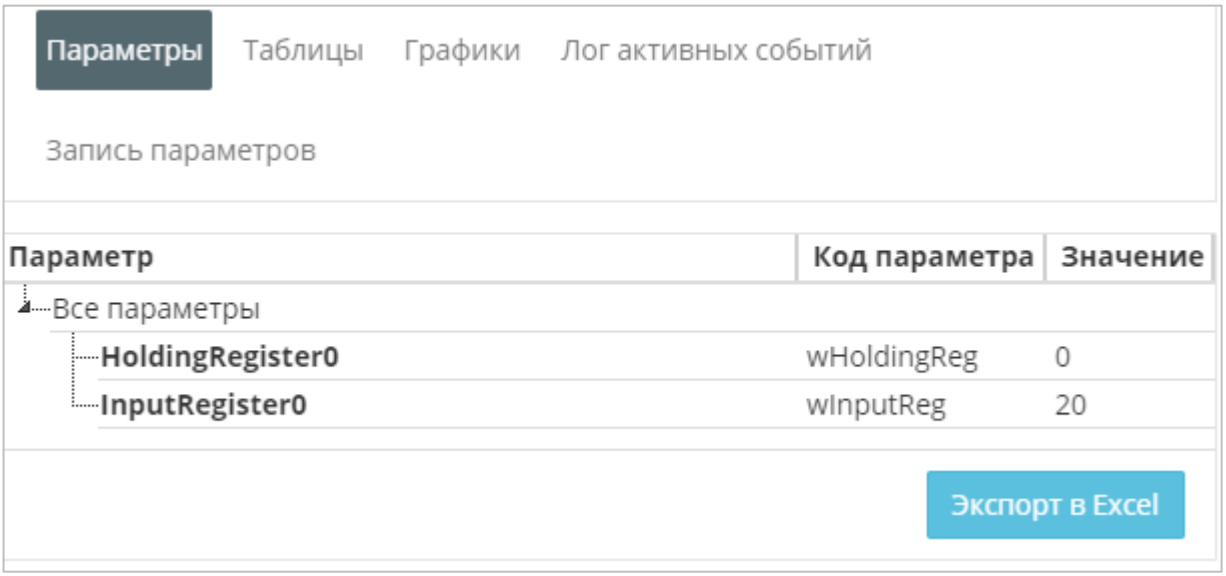


Рис. 5.12. Просмотр параметров прибора

Приложение

A. Использование объединений (UNION)

Стандарт **Modbus** предусматривает только два типа данных, участвующих в обмене – **BOOL** и **WORD**. Достаточно часто возникает потребность передать данные других типов, например, **REAL** и **STRING**. В этом случае на устройстве, которое отравляет данные, необходимо преобразовать их в последовательность **WORD** регистров. Соответственно, на устройстве, получающем данные, должно быть выполнено обратное преобразование. Наиболее простой способ сделать это в **CODESYS 3.5** – использовать объединения.

Объединение (UNION) представляет собой пользовательский тип данных, все переменные которого расположены в одной области памяти. Таким образом, переменные различных типов будут представлять различную интерпретацию одних и тех же данных. Для конвертации достаточно записать значение в одну из переменных объединения и считать его из другой.

Рассмотрим конвертацию значения с плавающей точкой, хранящегося в двух **WORD**, в переменную типа **REAL**:

1. Нажмите **ПКМ** на приложение **Application** и добавьте объект **DUT** типа **объединение** с названием **Real_Word**:

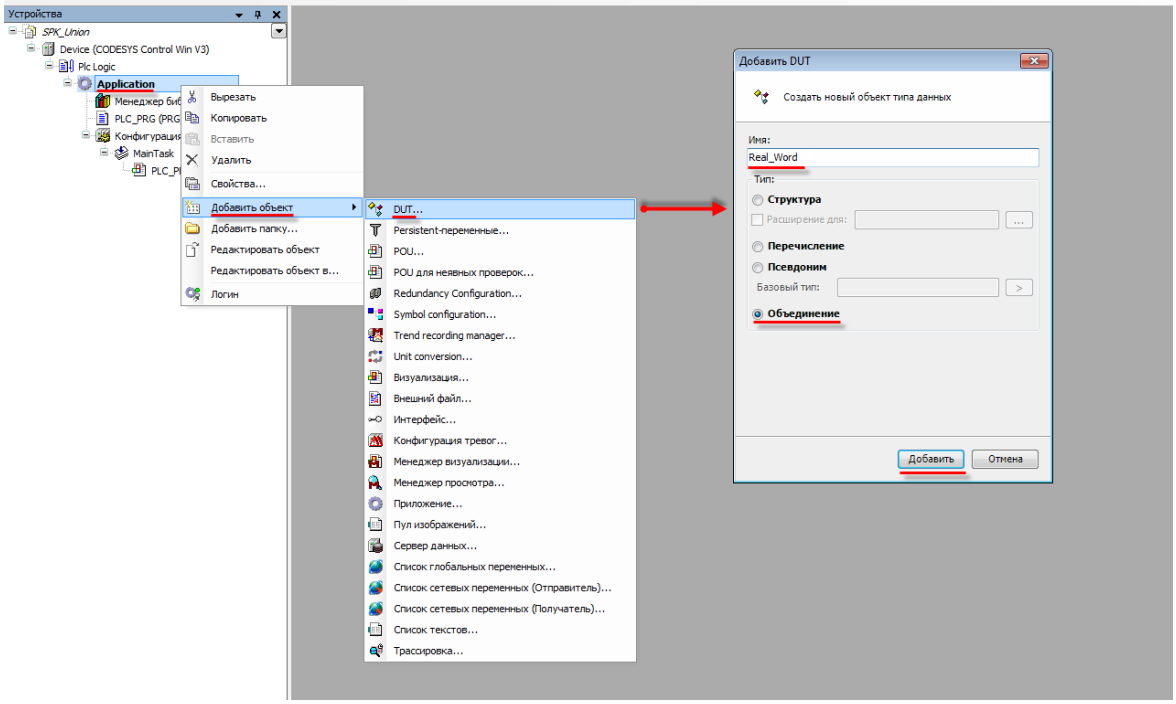


Рис. А.1. Добавление в проект объединения

2. В объединении объявите переменную **rRealValue** типа **REAL** и массив **awModbusReal** типа **WORD**, содержащий два элемента:

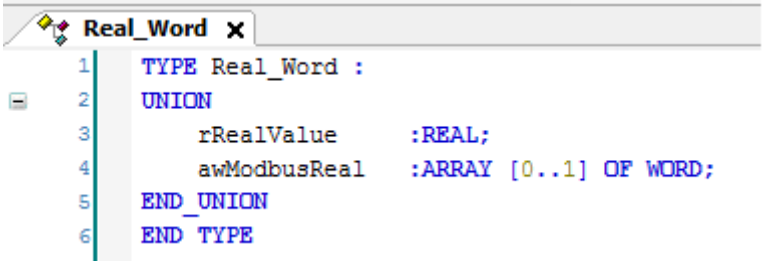


Рис. А.2. Объявление переменных объединения

3. В программе объявим экземпляр объединения **Real_Word** с названием **_2WORD_TO_REAL**:

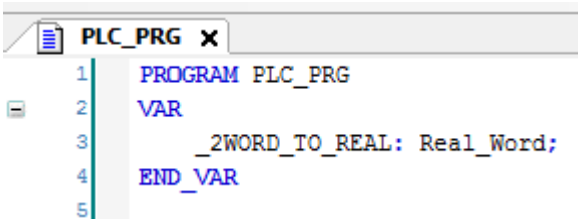


Рис. А.3. Объявление экземпляра объединения в программе

Для использования переменных объединения в нужном месте программы введите имя экземпляра объединения и нажмите точку, после чего выберите из списка нужную переменную:

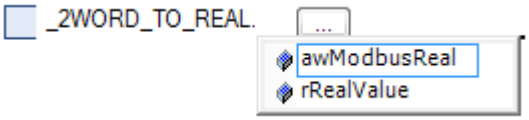


Рис. А.4. Работа с переменными объединения в программе

4. Переменные массива **awModbusReal** будут привязаны к регистрам при настройке **Modbus**, а переменная **rRealValue** будет использоваться в программе для работы со значением с плавающей точкой.