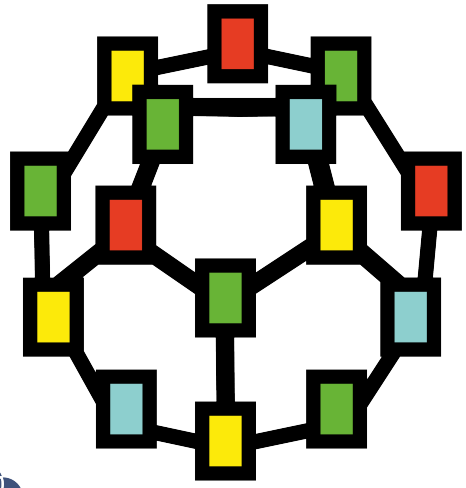
Библиотека paOpcUa



ПромАвтоматика

Руководство пользователя

09.2024
версия 1.2

Содержание

Используемые термины и сокращения

ПК – персональный компьютер.

ПЛК – программируемый логический контроллер.

SQL (**Structured Query Language**) – язык программирования для хранения и обработки информации в реляционной базе данных.

URI (Uniform Resource Identifier) – имя и адрес ресурса в сети.

OPC UA (Open Platform Communications, Unified Architecture) – промышленный протокол. Спецификация OPC UA разработана организацией OPC Foundation.

Введение

Настоящее руководство описывает настройку обмена данными по протоколу **OPC UA** для контроллеров ОВЕН, программируемых в среде **Полигон**. Подразумевается, что читатель обладает базовыми навыками работы с **Полигон**, поэтому общие вопросы (например, создание и загрузка проектов) в данном документе не рассматриваются – они подробно описаны в документах Руководство по программированию. Библиотека paCore и Быстрый старт.

Настройка обмена по протоколу **OPC UA** в среде **Полигон** осуществляется с помощью функциональных блоков из библиотеки ***paOpcUA***.

Примеры в документе актуальны для версии среды **Полигон** – **1994** и для версии библиотеки **paOpcUA** – **922** и выше.

1 Базовые сведения о технологии OPC

OPC (от англ. **Open Platform Communications**) – это набор программных технологий, которые предоставляют единый интерфейс для управления различными устройствами и обмена данными. Спецификации стандарта OPC были разработаны организацией OPC Foundation, которую создали в 1994 году ведущие производители средств промышленной автоматизации. Целью создания OPC было предоставить инженерам универсальный интерфейс для управления различными устройствами.

После стандартизации OPC-клиента и OPC-сервера на уровне программной архитектуры разработчики стандарта избавились от необходимости поддерживать сотни драйверов для различных устройств. Таким образом, технология OPC позволяет поддерживать связь приборов с различными SCADA-системами.

Технология OPC включает несколько стандартов, которые описывают набор функций определенного назначения. **OPC UA (Unified Architecture)** — последняя по времени выпуска спецификация, которая основана не на технологии Microsoft COM, что предоставляет кроссплатформенную совместимость.

2 Библиотека paOpcUA

paOpcUA – библиотека, обеспечивающая обмен по протоколу OPC UA в режиме сервера и/или клиента.

Каждый ПЛК, программируемый в **Полигон**, является OPC UA-сервером, так как ***Отладчик*** подключается к контроллеру как OPC UA-клиент. Поэтому присутствие библиотеки ***paOpcUA*** в проекте обязательно.

Преднастроенный OPC UA-сервер добавляется автоматически при выборе модуля из шаблона ***Модуль с отладчиком для контроллера*** в месте работы ***Фон***, программа ***Debug***.

Особенности реализации протокола OPC UA в **Полигон** приведены в Приложении Поддержка сервисов по спецификации OPC UA.

Для добавления библиотеки ***paOpcUA*** в проект следует:

1. Перейти в меню ***Окна/Проекты***. В появившемся окне отобразится текущий проект и добавленные библиотеки.

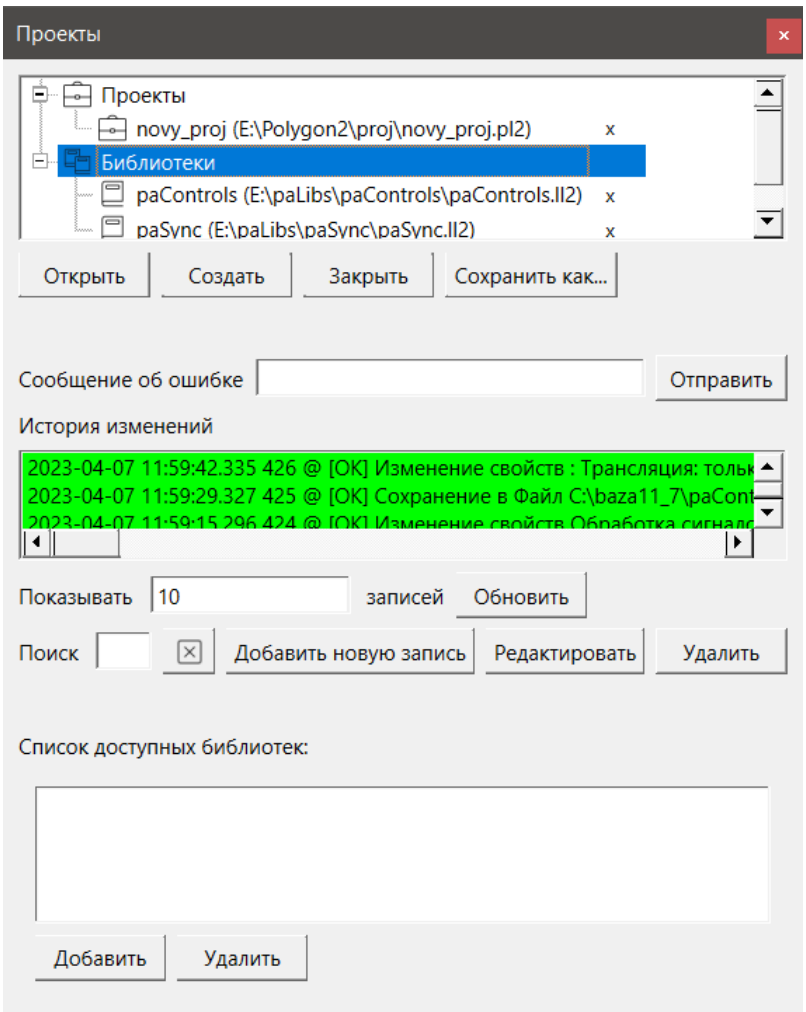


Рисунок 2.1 – Добавление библиотеки paOpcUA в проект

2. Нажать кнопку ***Открыть*** и перейти в папку с файлами библиотеки, которую необходимо добавить. Затем в выпадающем списке выбрать тип файла ***Библиотека Полигон 2 (*.ll2)***.

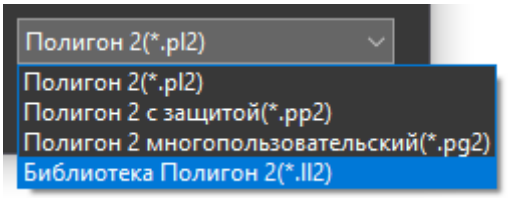


Рисунок 2.2 – Добавление библиотеки paOpcUA в проект

3. В окне появится файл библиотеки с расширением **.ll2**. Следует выбрать его и нажать **Открыть**.

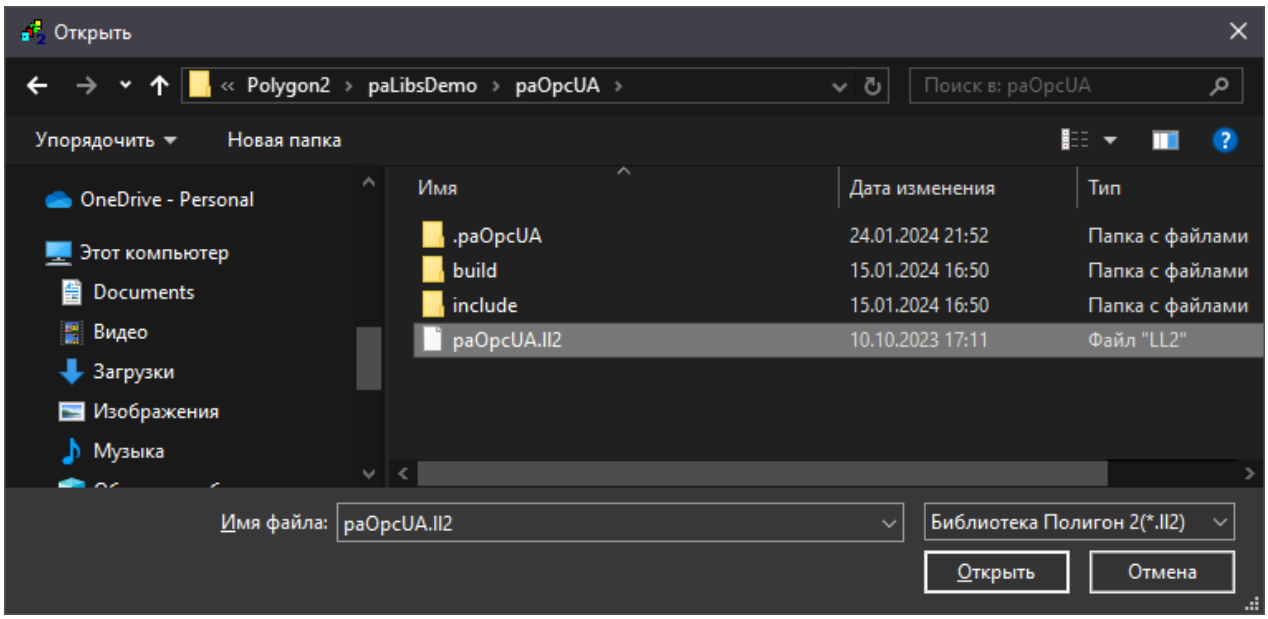


Рисунок 2.3 – Добавление библиотеки paOpcUA в проект

Добавленная библиотека отобразится в окне ***Проекты***.

2.1 OPC UA-сервер (OpcUAServer)

Блок ***OpcUAServer*** обеспечивает реализацию сервера по протоколу **OPC UA**. Присутствие данного блока в проекте обязательно, так как ***Отладчик*** подключается к контроллеру как OPC UA-клиент.

Данный блок можно разместить только в ***Фоне***.

На входы блока **ip** – IP-адрес контроллера и **prt** – локальный порт контроллера рекомендуется подавать SQL-запросы к соответствующим свойствам текущего модуля.

Запрос IP-адреса (prop_ip):
```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос номера порта (prop_debug_port):
```
<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_debug_port"</sql>
```

При добавлении циклических входов **cip** к OPC UA-серверу смогут подключаться только клиенты с указанным IP-адресом (в т. ч. ***Отладчик***).

Таблица 2.1 – Назначение входов и выходов OpcUAServer

Элемент	Описание
Входы	
ip	Локальный IP адрес (константный)
prt	Локальный порт (константный)
sdr	Сетевой стек, для ПЛК ОВЕН **"/"** (константный)
st	Статус сервера в соответствии со спецификацией OPC UA (см. Part 5 – 12.6 ServerState)
sl	Дополнительный статус сервера ServiceLevel в соответствии со спецификацией OPC UA (см. Part 4 – 6.6.2.4.2 ServiceLevel): **255** – ведущий **199** – ведомый
max	Максимально разрешенное количество соединений (константный)
cip	IP разрешенного подключения (циклический и константный)
cpr	Приоритет подключения (циклический и константный)
Выходы	
st	Текущий статус сервера в соответствии со спецификацией OPC UA (см. Part 5 – 12.6 ServerState)
cn	Количество активных соединений (сумма **cnDbg**, **cnPA**, **cnOth**)

Продолжение таблицы 2.1

Элемент	Описание
cnDbg	Количество соединений с ***Отладчиком*** среды разработки
cnPA	Количество соединений с клиентами OpcUAClient
cnOth	Количество соединений с другими клиентами
cst	Статус соединения с разрешенным клиентом (циклический)

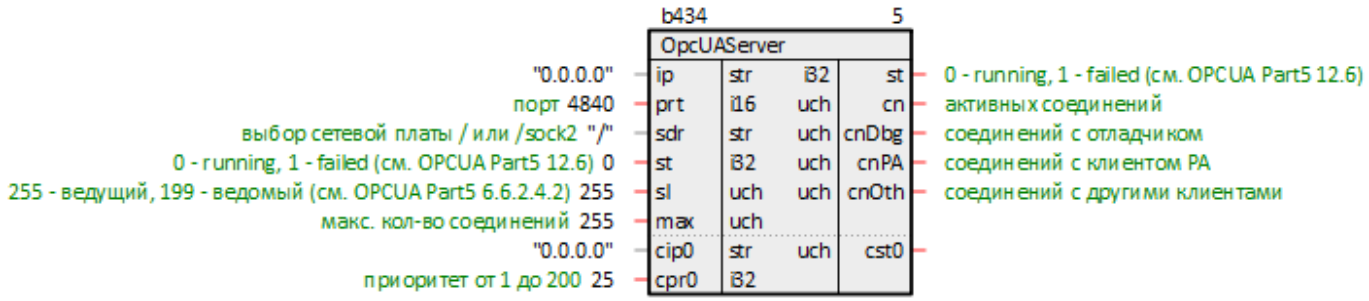


Рисунок 2.4 – OPC UA-сервер (OpcUAServer)

2.2 OPC UA-сервер, таймерная часть (OpcUAServerTimer)

Блок ***OpcUAServerTimer*** обеспечивает возможность сбора данных для подписок OPC UA-сервера в таймере. Для этого необходимо добавить блок в место работы ***Таймер*** с минимальным размером таймерного промежутка. Данный блок можно разместить только в ***Таймере***.

Блок ***OpcUAServerTimer*** должен встречаться в проекте только один раз независимо от того, сколько OPC UA-серверов добавлено в проект. После этого можно настраивать параметры сбора данных в подписке с дискретностью таймерного цикла. В частности, это можно использовать для отладки программы в представлении ***График***. Надо учитывать, что при инициализации подписки клиентом OPC UA с дискретизацией, кратной таймерному промежутку, в контроллере выделяется память для хранения очереди накапливаемых значений.

Назначение входов/выходов:

enb – разрешение на работу блока;

subsN – количество подписок, обрабатываемых в таймере;

dpsN – количество данных

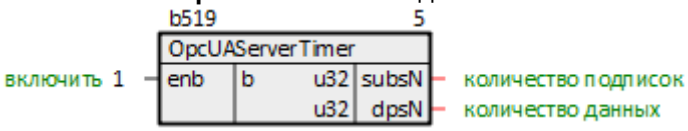


Рисунок 2.5 – OPC UA сервер, таймерная часть (OpcUAServerTimer)

2.3 OPC UA-клиент (OpcUAClient)

Блок ***OpcUAClient*** обеспечивает реализацию одной подписки к серверу по протоколу **OPC UA**. Входы и выходы, которыми необходимо обмениваться с сервером необходимо добавить в раздел ***Данные*** внутри этого блока. Данный блок можно разместить только в ***Фоне***.

На входы блока **Iip** – IP-адрес контроллера и **Iprt** – локальный порт контроллера рекомендуется подавать SQL-запросы к соответствующим свойствам текущего модуля.

Запрос IP-адреса (prop_ip):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос номера порта (prop_debug_port):

```
<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_debug_port"</sql>
```

Таблица 2.2 – Назначение входов и выходов OpcUAClient

Элемент	Описание
Входы	
enb	Разрешение на работу блока: **0** – выключен; **1** – включен; **2** – в резерве
wait	Таймаут ожидания ответа от сервера, мс (константный)
lip	Локальный IP-адрес (константный)
lprt	Локальный порт (константный)
sdr	Сетевой стек, для ПЛК ОВЕН **"/"** (константный)
rip	IP адрес сервера (константный)
rprt	Порт сервера (константный)
usr	Логин для доступа к серверу (константный)
psw	Пароль для доступа к серверу (константный)
pbl	Интервал работы подписки в мс (PublishingInterval) – периодичность работы подписки внутри сервера. Если **pbl = 0**, то сервер обрабатывает подписку каждый фоновый цикл (константный)
lfc	Время жизни подписки в циклах (LifetimeCount) (константный)
kac	Максимальное значение keep-alive счетчика в циклах (MaxKeepAliveCount) (константный)
wcon	Отправлять значения выходов при установке соединения
routs	Читать выходы
cfg	Дополнительная конфигурация работы **ПРИМЕЧАНИЕ** Начиная с версии **901** библиотеки ***paOpcUA*** в блоке ***OpcUAClient*** отключен разрыв соединения в случае отсутствия некоторых данных в сервере. Для возвращения старого поведения следует установить бит **0** на входе **cfg** в **1**.
buf	Подключение блоков ***BufSupEx*** из библиотеки paCore (циклический)
Выходы	
sts	Статус работы: **0** – нет обмена; **1** – обмен данными; **2** – в резерве, соединение установлено, значения выходов **sst** и **ssl** актуальные, обмена данными нет; **3** – в процессе установки соединения; **4** – в процессе разрыва соединения; **5** – сервер вернул ошибку (более подробную информацию программа печатает в консоль); **6** – внутренняя ошибка, перезапуск соединения; **–1** – системная ошибка, невозможно выделить локальный сокет для установки соединения; **–2** – системная ошибка, невозможно использовать локальный порт для установки соединения
sst	Статус сервера в соответствии со спецификацией OPC UA (см. Part 5 – 12.6 ServerState)
ssl	Дополнительный статус сервера ServiceLevel в соответствии со спецификацией OPC UA (см. Part 4 – 6.6.2.4.2 ServiceLevel): **255** – ведущий; **199** – ведомый
sid	ID подписки
ssn	Номер уведомления подписки
rcnt	Количество принятых пакетов
wcnt	Количество отправленных пакетов

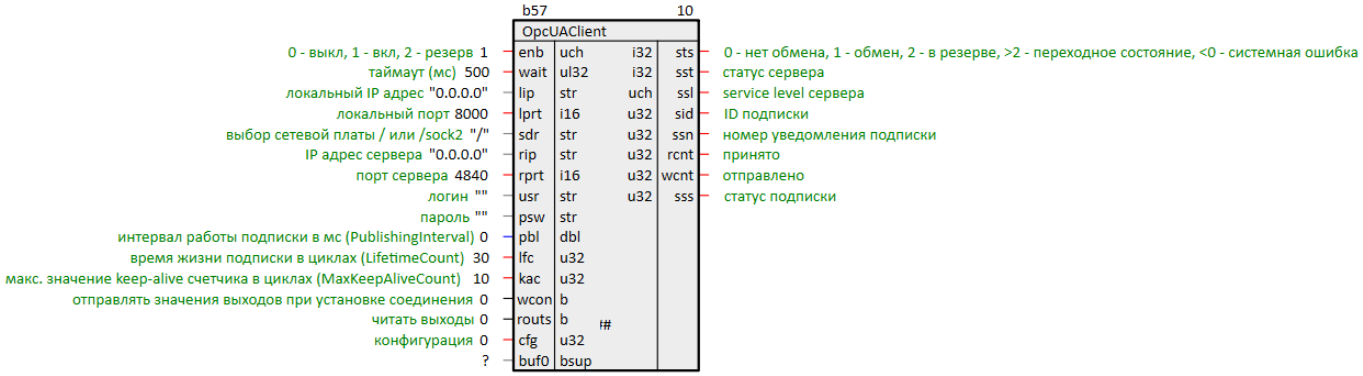


Рисунок 2.6 – OPC UA клиент (OpcUAClient)

2.4 Управление блоками BufSup (UABufSups)

Блок ***UABufSups*** предназначен для подключения блоков ***BufSupEx*** из библиотеки ***paCore*** и записи/чтения данных. Для записи данного в файл ***BufSupEx*** клиенту необходимо записать значение во вход или выход **dan**. Прочитать текущее значение параметра из файла клиент может с выхода **dan**. Данный блок можно разместить только в ***Фоне***.

Назначение входов/выходов:

enb – разрешение на работу блока;

buf – вход для подключения блоков ***BufSupEx*** из библиотеки paCore (циклический);

itr – выход для подключения блоков ***BufSupEx***;

cnt – записано

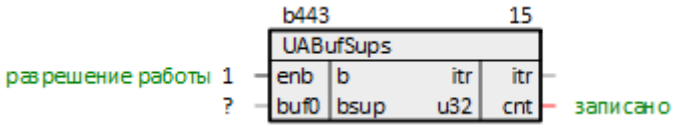


Рисунок 2.7 – Управление блоками BufSup (UABufSups)

3 OPC UA-сервер

Каждый ПЛК, программируемый в **Полигон**, является OPC UA-сервером, так как ***Отладчик*** подключается к контроллеру как OPC UA-клиент.

Преднастроенный OPC UA-сервер добавляется автоматически при создании модуля из шаблона ***Модуль с отладчиком для контроллера*** в месте работы ***Фон***, программа ***Debug***.

При настройке блока OpcUAServer удобно использовать некоторые свойства модуля. Для этого можно использовать технологию SQL-запросов. Такие запросы автоматически добавляются на входы **ip** и **prt** при создании модуля из шаблона ***Модуль с отладчиком для контроллера***. Это позволяет изменять IP-адрес и порт в одном месте, и использовать эти значения в разных частях проекта.

Запрос IP-адреса (prop_ip):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос номера порта (prop_debug_port):

```
<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_debug_port"</sql>
```

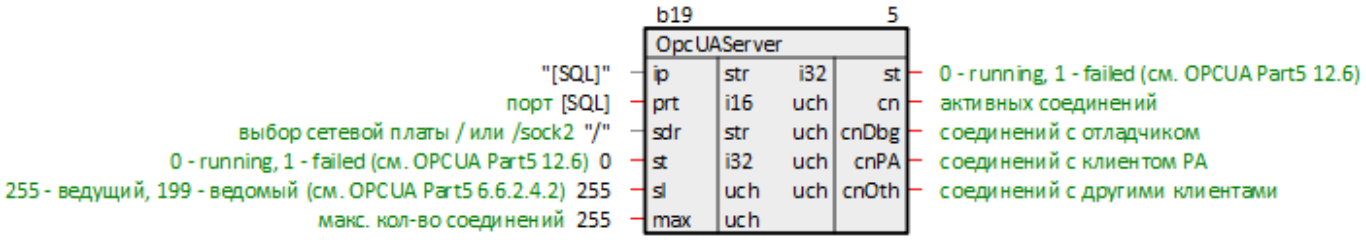


Рисунок 3.1 – Преднастроенный OPC UA-сервер

3.1 Адресация данных

С точки зрения OPC UA-клиента исполняемая программа выглядит как дерево, в котором модуль представлен так же, как в представлении ***Дерево*** в среде разработки (за исключением страниц, они не отображаются как отдельные объекты).

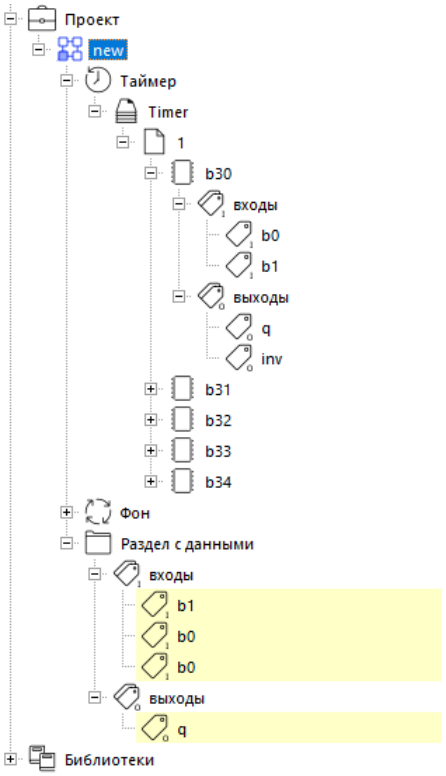


Рисунок 3.2 – Дерево проекта

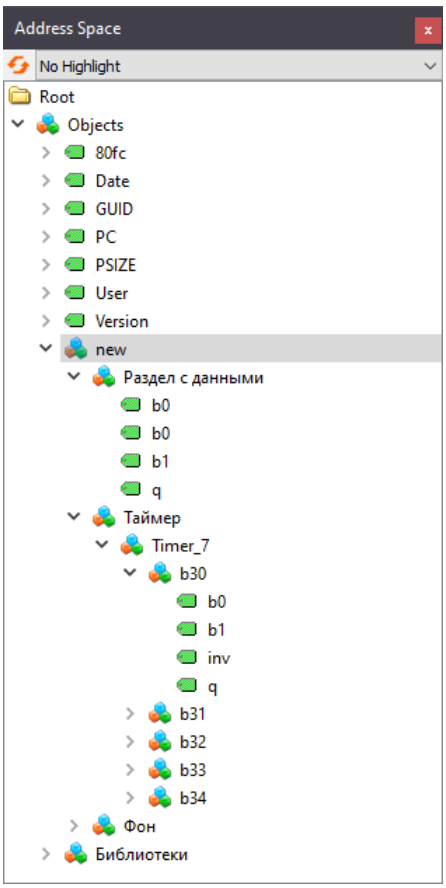


Рисунок 3.3 – Дерево проекта при чтении OPC UA-клиентом (UaExpert)

Узлы дерева представлены двумя типами:

- для модуля, места работы, программы, раздела и функционального блока ***NodeClass = Object***;
- для входов, выходов и их свойств ***NodeClass = Variable***.

Отношение между основными узлами (***Reference***) имеет тип ***HasComponent***. Отношение между входом/выходом и его свойством имеет тип ***HasProperty***.

Тип адреса используется числовой (***NodeId.IdentifierType = Numeric***) и имеет два параметра:

- ***ns*** (NamespaceIndex);
- ***i*** (Identifier).

Для модуля, места работы, программы, раздела и функционального блока ***i*** равен индексу – свойство компонента ***Индекс***.

Для модуля, места работы, программы, раздела и функционального блока ***ns*** равен **0x8000** (**32768** в десятичной системе счисления).

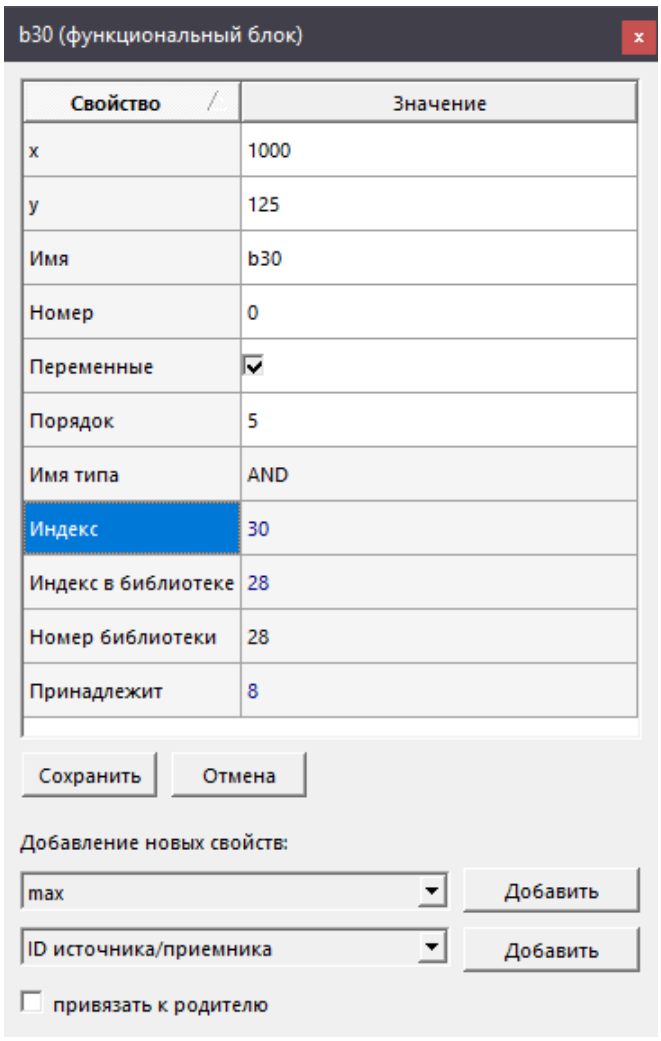


Рисунок 3.4 – Индекс блока в программе

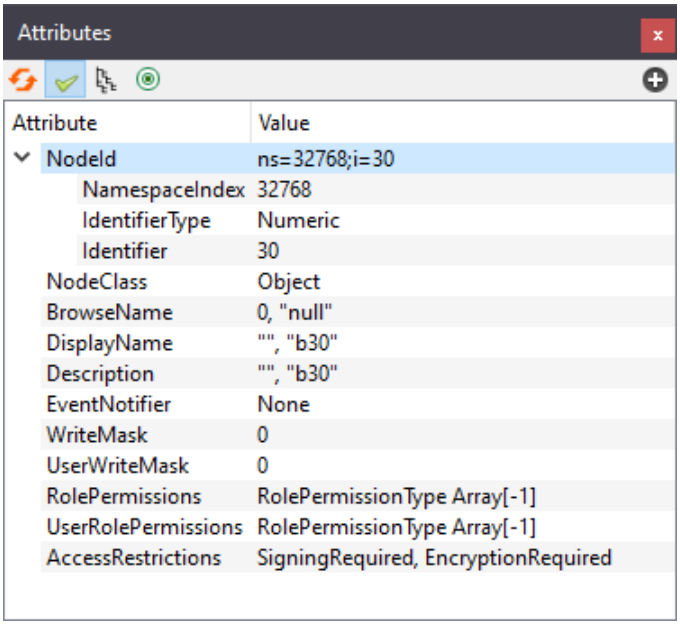


Рисунок 3.5 – NodeId блока при чтении OPC UA-клиентом (UaExpert)

Для входов: ***i*** равен свойству ***Индекс блока***, ***ns*** равен свойству ***Порядок***.

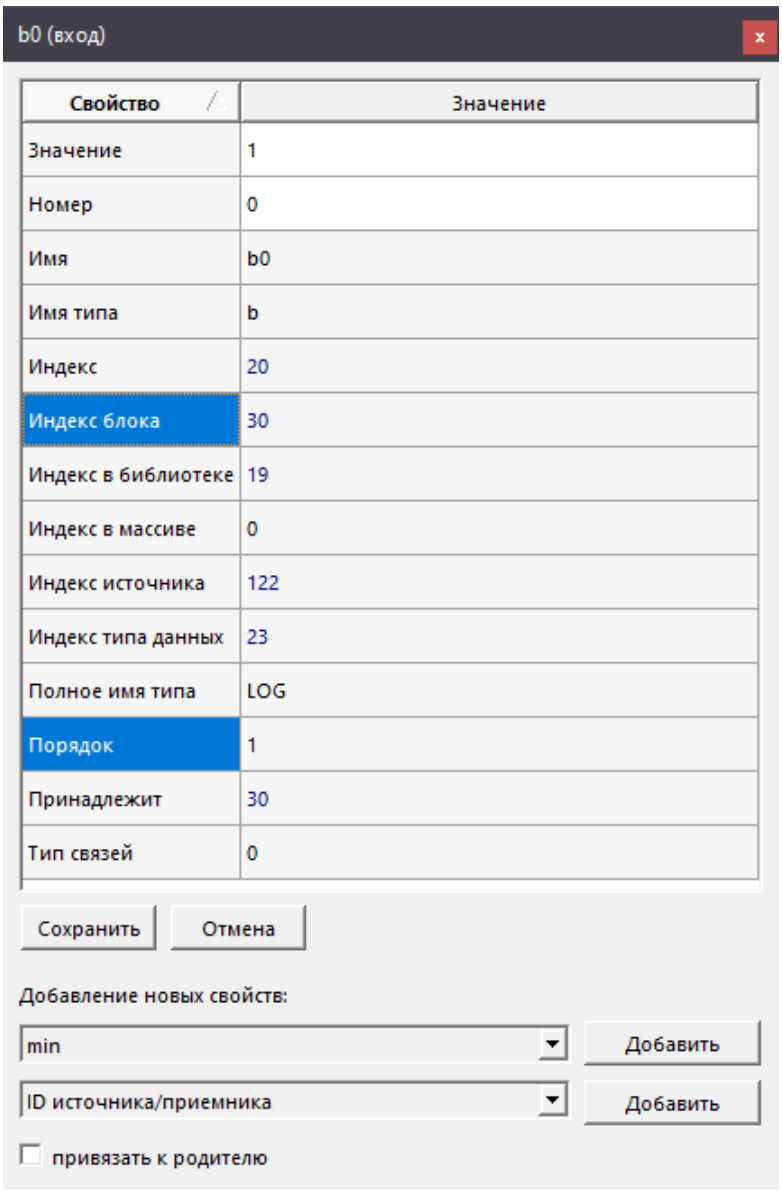


Рисунок 3.6 – Свойства входа в программе

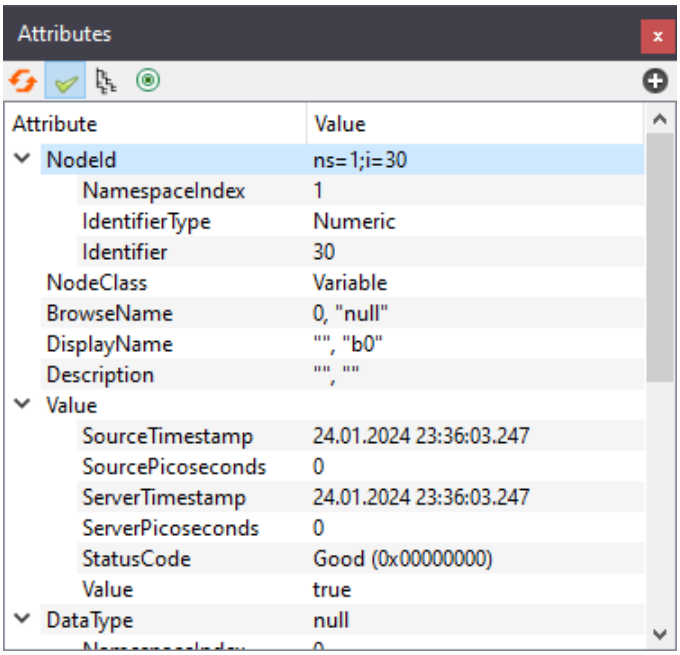


Рисунок 3.7 – Свойства входа при чтении OPC UA-клиентом (UaExpert)

Для выходов: ***i*** равен свойству ***Индекс блока***, ***ns*** равен свойству ***Порядок*** при побитовом сложении с **0x8000**.

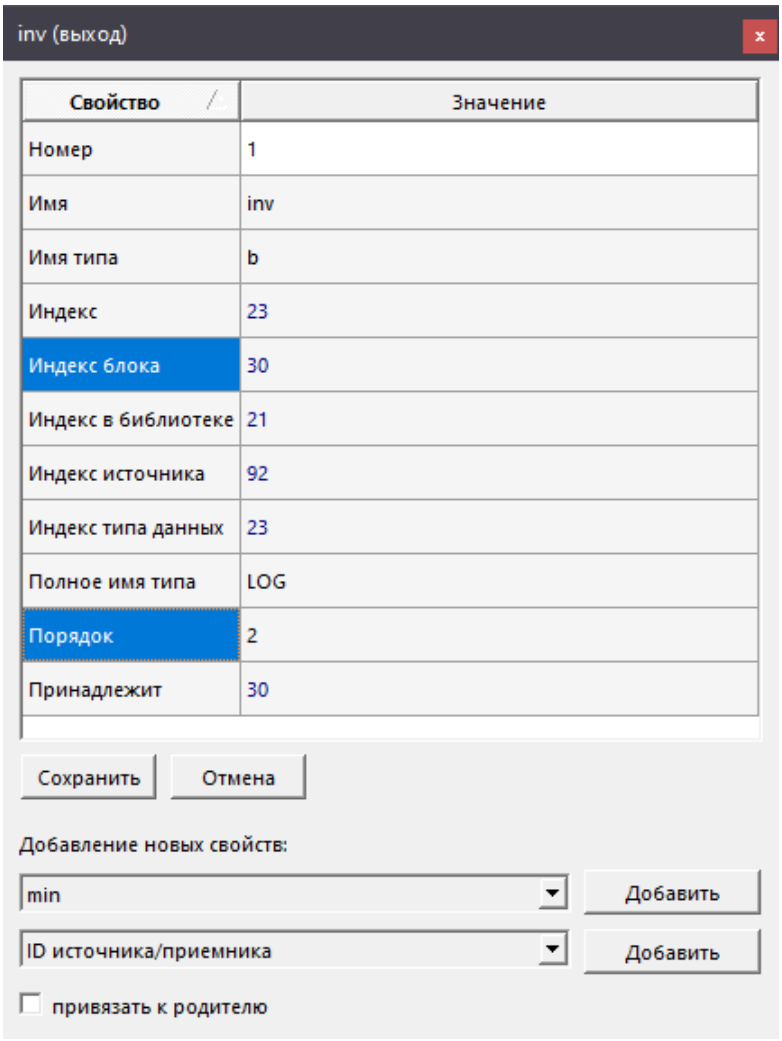


Рисунок 3.8 – Свойства выхода в программе

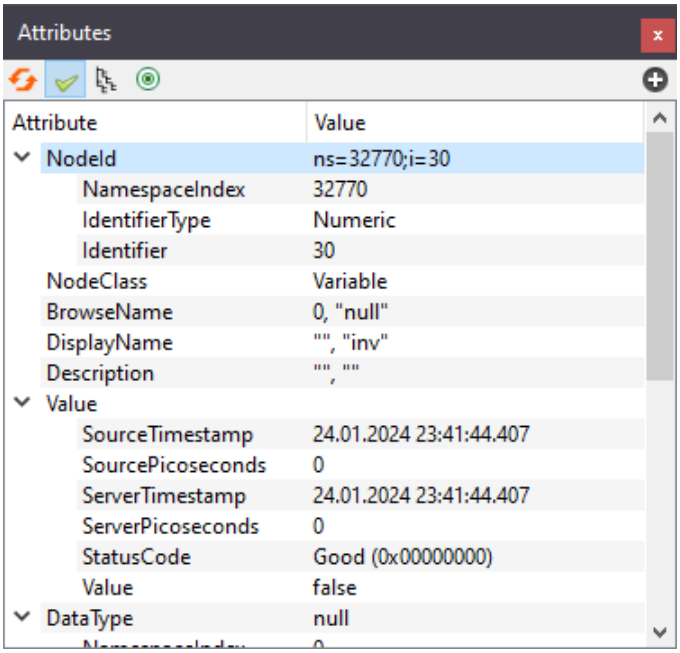


Рисунок 3.9 – Свойства выхода при чтении OPC UA-клиентом (UaExpert)

3.2 Сбор данных для подписок в Фоне и в Таймере

Сбор данных для подписок OPC UA-сервера производится в фоновом потоке. Для того, чтобы собирать данные подписок в таймерном прерывании, необходимо добавить добавить блок OpcUaServerTimer на любую страницу места работы ***Таймер***.

Блок ***OpcUAServerTimer*** должен встречаться в проекте только один раз независимо от того, сколько OPC UA-серверов добавлено в проект. После этого можно настраивать параметры сбора данных в подписке с дискретностью таймерного цикла.

В частности, это можно использовать для отладки программы в представлении ***График***. График подключается к OPC UA-серверу контроллера в качестве клиента с добавленными в него данными. Помимо добавления блока ***OpcUAServerTimer*** для сбора данных графика в таймере перед его запуском в настройках следует установить флаг ***Делать отсчеты в таймере***.

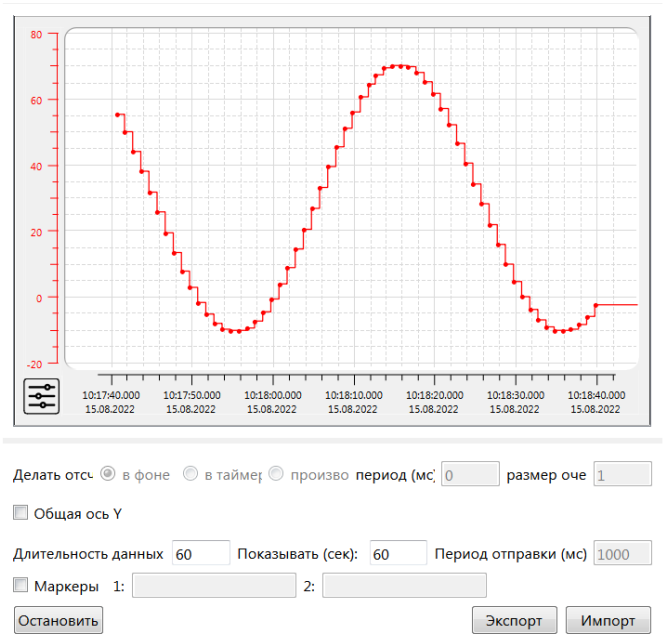


Рисунок 3.10 – Пример работы графика в фоне

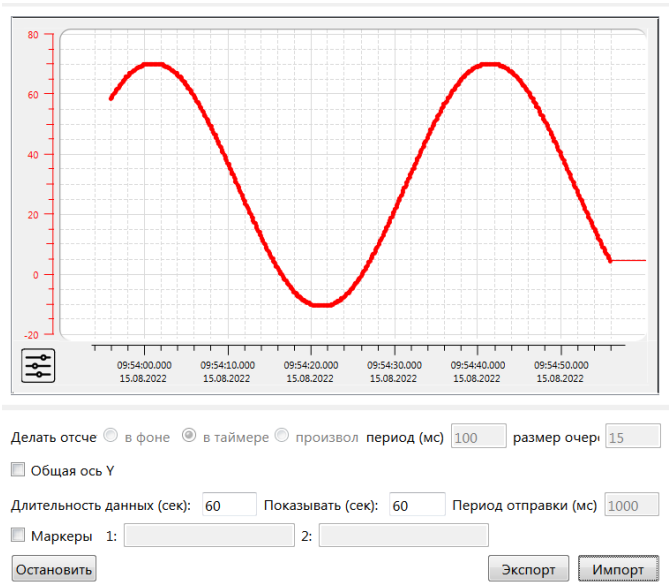


Рисунок 3.11 – Пример работы графика в таймере

Надо учитывать, что при инициализации подписки OPC UA-клиентом с дискретизацией, кратной таймерному промежутку, в контроллере выделяется память для хранения очереди накапливаемых значений.

ПРИМЕЧАНИЕ
Если сбор данных в таймере используется только для целей отладки, перед вводом ПЛК в работу блок ***OpcUAServerTimer*** рекомендуется отключать (вход **enb** = 0), чтобы не нагружать место работы ***Таймер***.

3.3 Авторизация

В **Полигон** поддерживается два вида авторизации для доступа к OPC UA-серверу: анонимный доступ и доступ по логину и паролю.

Все блоки OpcUAServer, добавленные в модуль, обеспечивают по умолчанию доступ ко всем входам/выходам проекта в том случае, если клиент использует для авторизации имя пользователя ***admin*** (свойство модуля ***Пароль admin***). Этот способ соединения также использует и отладчик среды разработки.

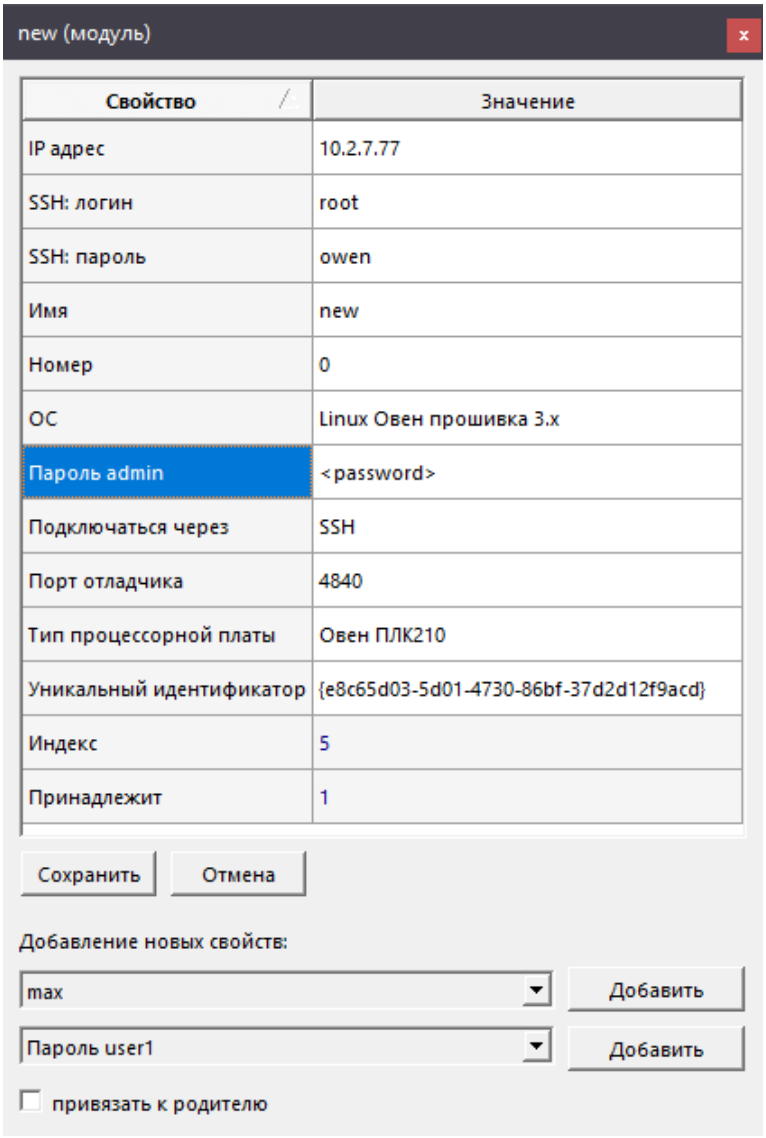


Рисунок 3.12 – Пароль admin

Чтобы ограничить видимость частей проекта для клиента, следует использовать другие учетные записи.

Анонимный доступ по умолчанию предоставляет доступ только к диагностической информации: дате трансляции, идентификатору модуля, ПК, с которого транслировалась программа, пользователь ПК, версия проекта.

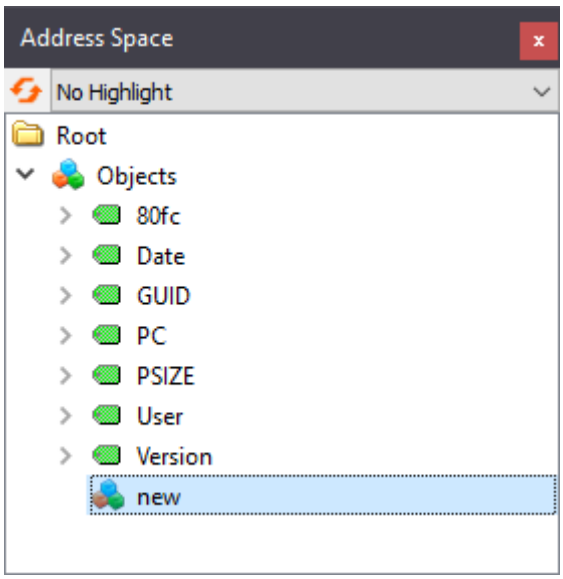


Рисунок 3.13 – Чтение OPC UA-клиентом (UaExpert) с анонимным доступом

Помимо пользователя ***admin*** в среде есть возможность подключения еще 7 пользователями – ***user1***, …, ***user7***. Пароль для ***user1***, …, ***user7*** устанавливается свойствами модуля ***Пароль user1***, …, ***Пароль user7***. По умолчанию для пользователей ***user1***, …,***user7*** также доступна только диагностическая информация.

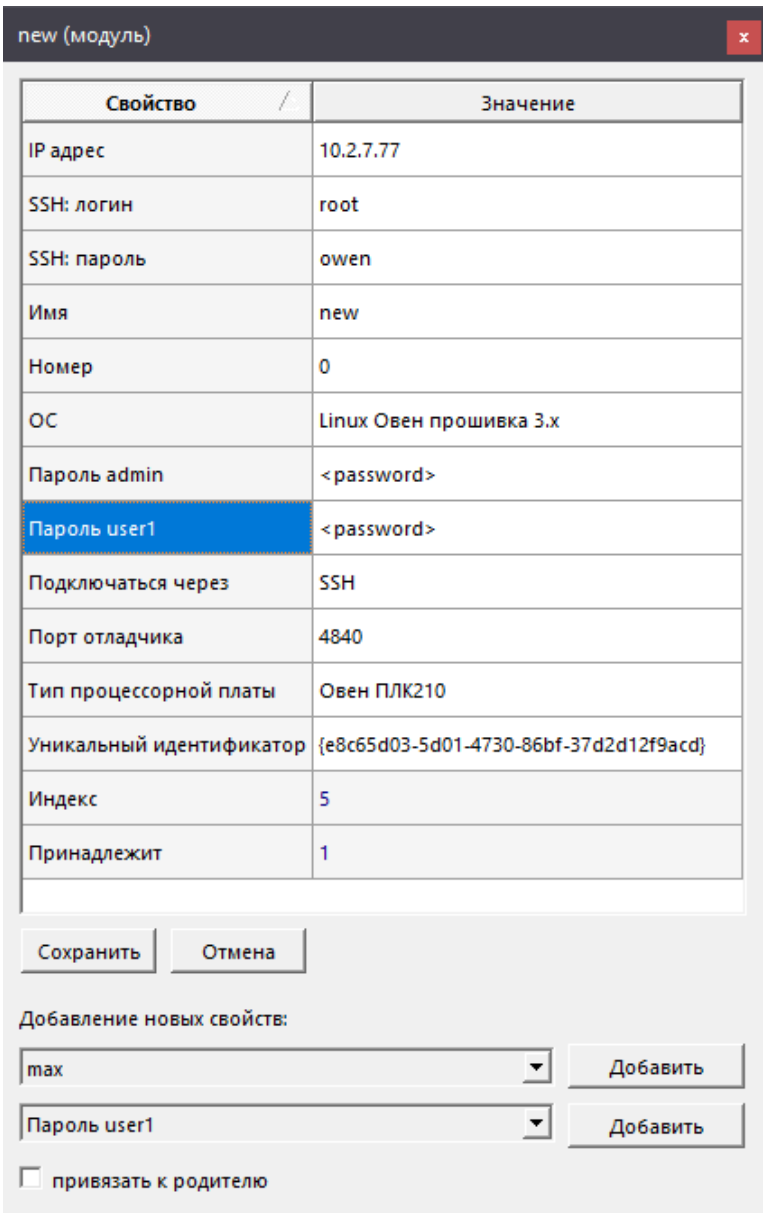


Рисунок 3.14 – Пароль user1

Для того, чтобы разрешить анонимный доступ или доступ пользователями **user1**, …, **user7** к частям проекта, следует добавить узлам проекта свойство **Разрешить доступ**.

Для разрешения анонимного доступа необходимо установить ***Разрешить доступ = user0*** для пользователей ***user1***, …,***user7*** – ***Разрешить доступ = user1***, …,***user7***.

Свойство ***Разрешить доступ*** можно добавить для разделов и программ.

ВНИМАНИЕ
Чтобы разрешить в программе доступ к входам/выходам блоков, все узлы дерева проекта, находящиеся выше данной программы, также должны иметь такое же свойство **Разрешить доступ**.

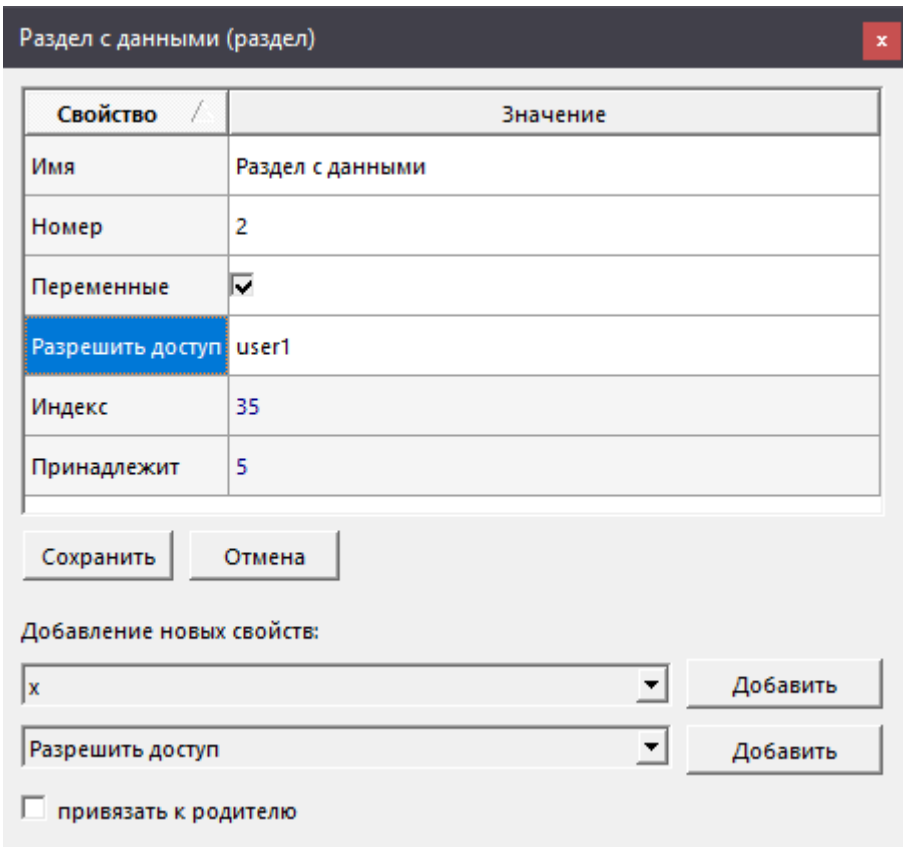


Рисунок 3.15 – Свойство Разрешить доступ = user1

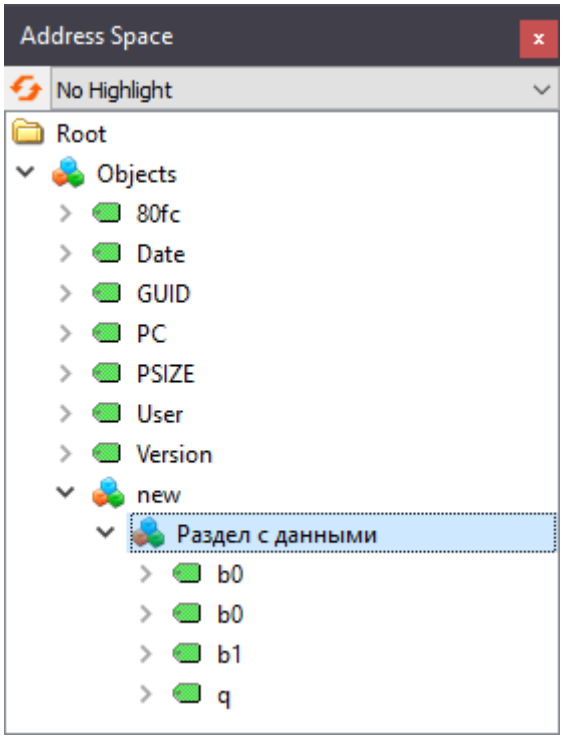


Рисунок 3.16 – Чтение OPC UA-клиентом (UaExpert) с user1

Все указанные выше пароли вступают в силу только после трансляции модуля.

3.4 Ограничение подключений к серверу

По умолчанию к блоку OpcUAServer возможно подключение клиентами с любыми IP адресами при условии отсутствия ограничения межсетевым экраном.

В блоке ***OpcUAServer*** также есть возможность ограничить доступ – для этого необходимо создать у блока дополнительные группы входов ***m_allowed_clients***, по одной на каждый разрешенный IP-адрес.

Входы **cip** позволяют ограничить подключения только определенными IP-адресами. Входы **cpr** задают разные приоритеты для адресов.

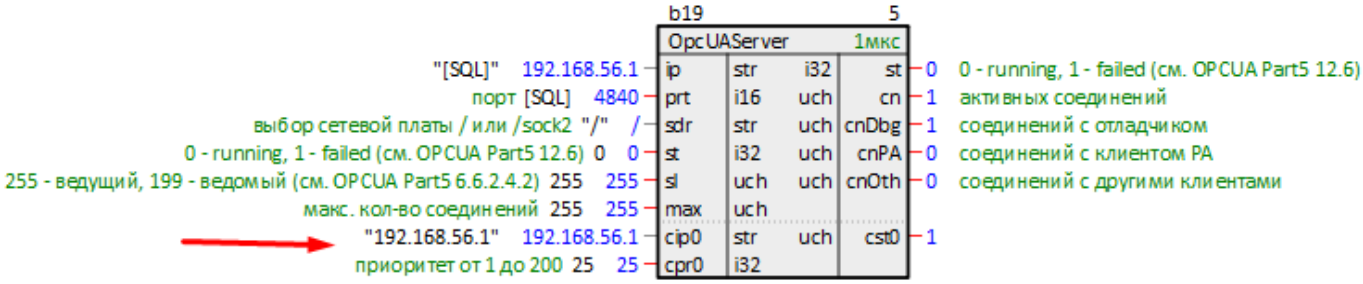


Рисунок 3.17 – OpcUAServer с разрешенным IP-адресом

При использовании **cip** и **cpr** OPC UA-сервер не будет принимать подключений ни с каких IP-адресов, кроме заданных.

3.5 Свойства входов/выходов данных, влияющих на обмен с клиентом

В среде **Полигон** для входов/выходов, участвующих в обмене по **OPC UA**, можно задать свойства, влияющие на обмен с клиентом.

Свойство ***Зона нечувствительности*** (prop_deadzone) позволяет задать абсолютную зону нечувствительности на стороне сервера. Если значение изменилось больше, чем величина зоны нечувствительности, то оно передается клиенту.

Если задать **Зона нечувствительности = 0**, значение будет передаваться при изменении.

Согласно спецификации **OPC UA**, данный параметр должен устанавливаться на стороне клиента, поэтому возможность установить его на стороне сервера является дополнительной. Она предусмотрена для особых случаев, когда клиент не поддерживает этот параметр.

При соединении с клиентом OpcUAClient свойство **Зона нечувствительности** следует задавать на стороне клиента. Заданная на стороне сервера зона нечувствительности игнорируется.

Свойство ***Интервал принудительной отправки*** (в мс) (prop_reftime) позволяет организовать принудительную отправку данного в подписке независимо от изменения значения и настроек зоны нечувствительности. Заданный интервал отсчитывается от момента последней отправки значения. При использовании в качестве клиента ***OpcUAClient*** данный параметр можно (и желательно) задавать на стороне клиента, при использовании других клиентов – только на стороне сервера.

ВНИМАНИЕ
Для корректной работы данных свойств входов/выходов необходимо добавить их в любой раздел внутри модуля и установить свойство модуля ***Трансляция: включить свойства входов/выходов*** = ***Только из разделов***.

4 OPC UA-клиент

Один блок OpcUAClient соответствует одной подписке в OPC UA-сервере (Subscription в спецификации OPC UA). Кроме этого, клиент позволяет записывать данные в сервер.

Входы **lip**, **lprt**, **sdr**, **rip**, **rprt**, **usr** и **psw** задают настройки сетевого соединения с сервером.

Входы **pbl**, **lfc** и **kac** задают параметры подписки согласно спецификации OPC UA.

Входы **wcon**, **routs** определяют работу с выходами, добавленными у клиента.

При настройке блока ***OpcUAClient*** удобно использовать некоторые свойства модуля. Для этого можно использовать технологию SQL-запросов. Это позволяет изменять IP-адрес и порт в одном месте, и использовать эти значения в разных частях проекта.

Запрос IP адреса (prop_ip):
```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос номера порта (prop_debug_port):
```
<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_debug_port"</sql>
```

Запрос пользовательского свойства ***Пользовательское свойство 00*** (prop_0):
```
<sql SELECT value FROM blocks_prXop WHERE indx=:module AND type="prop_0"</sql>
```

(обычно используется на входах **rip** и **rprt**).

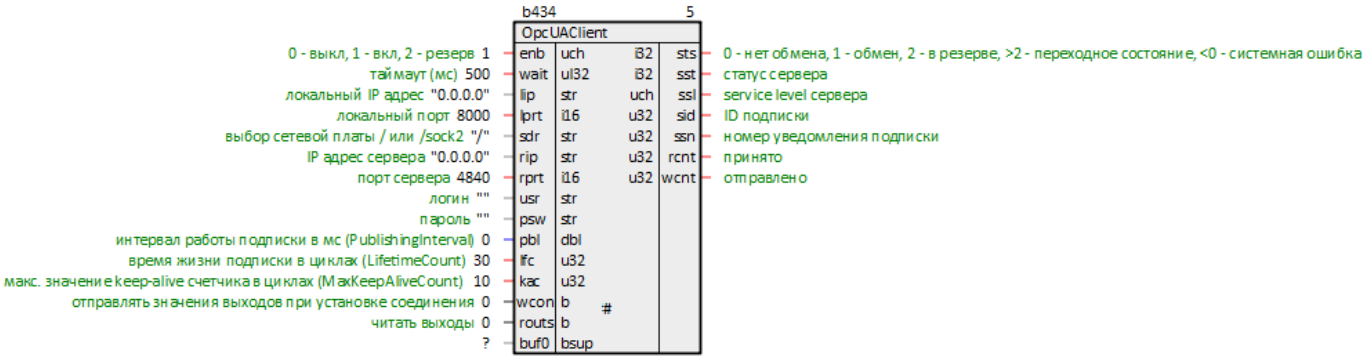


Рисунок 4.1 – OPC UA клиент (OpcUAClient)

4.1 Конфигурирование данных

Клиент читает и пишет данные (входы и выходы других функциональных блоков), добавленные внутрь раздела ***Данные***.

Добавить данное в раздел можно одним из следующих способов:

1. Открыть на одной странице блок OpcUAClient, на другой странице блок с входом/выходом, который необходимо добавить. Выделить вход/выход и с нажатым **Ctrl** перетащить его на блок ***OpcUAClient***. Отпустить, выбрать команду ***Добавить***.
2. Открыть блок ***OpcUAClient*** в дереве (со страницы это проще всего сделать командой ***Показать в дереве***), раскрыть его. Вход/выход перетащить в раздел ***Данные***, выбрать команду ***Добавить***.

Чтобы назначить данному адрес входа/выхода сервера, необходимо перетащить его со страницы с нажатым **Ctrl** на выход/вход в разделе ***Данные*** и в выпадающем меню выбрать команду ***Назначить источником***.

Также можно перетащить вход/выход из дерева сервера на нужный выход/вход в разделе ***Данные***.

После этого среда автоматически добавит в свойства данного из раздела свойство ***ID источника/приемника***, равное адресу назначенного входа/выхода сервера. Первое значение свойства не участвует в адресации, второе значение свойства определяет Id адреса входа/выхода сервера, третье – ns.

При подключении ***OpcUAClient*** к стороннему OPC UA-серверу свойство ***ID источника/приемника*** требуется добавлять и задавать вручную. При этом сервер должен поддерживать идентификаторы числового типа (Numeric).

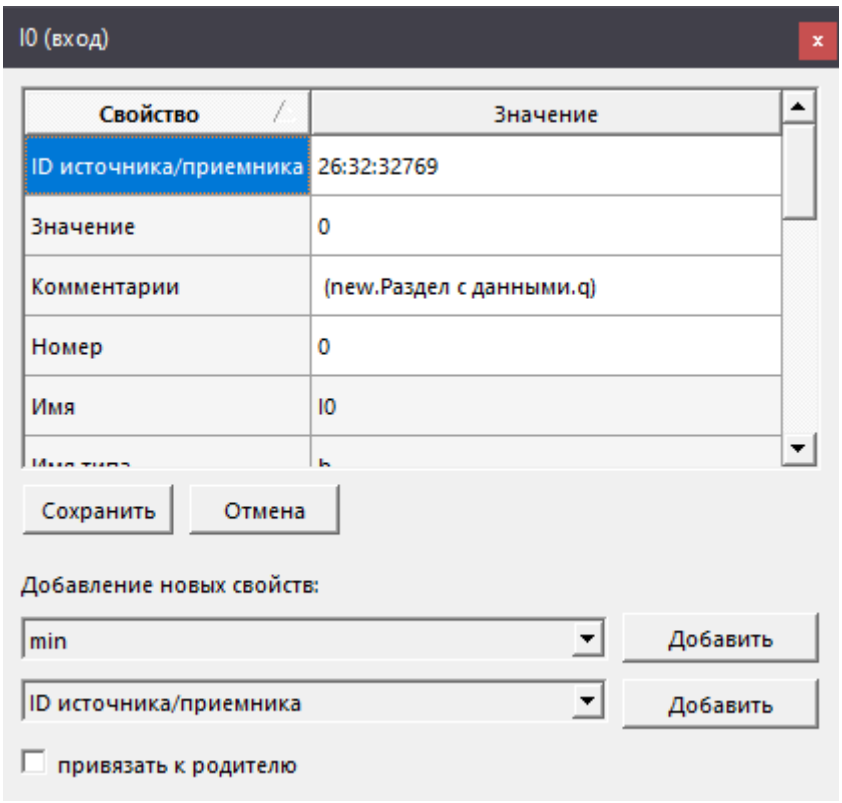


Рисунок 4.2 – Свойство ID источника/приемника

Входы и выходы блоков сервера можно назначать источниками как для входов, так и для выходов клиента из раздела ***Данные***.

Входы, добавленные в раздел ***Данные***, клиент регистрирует в сервере для чтения через подписку (MonitoredItem по спецификации OPC UA).

Выходы, добавленные в раздел ***Данные***, клиент записывает при помощи сервисов записи в OPC UA-сервер. Их поведение определяется входами **wcon** и **routs** блока ***OpcUAClient*** и свойством ***Зона нечувствительности***. Если клиент осуществляет запись в выходы блоков сервера, то соответствующие блоки сервера выключаются. Это может быть полезно, например, при отладке проекта с использованием внешней программы-модели.

4.2 Режимы работы клиента

Значение на входе **enb** определяет режим работы клиента OpcUAClient:

- **0** – соединение с сервером разорвано, обмена нет;
- **1** – соединение с сервером установлено, обмен работает;
- **2** (в резерве) – соединение с сервером установлено, обмена нет. В этом режиме клиент читает от сервера OpcUAServer только значения статусов (выходы **sst** и **ssl**), такой вариант можно использовать для выбора одного из нескольких серверов с наибольшим **ssl**.

Режим **2** можно использовать для подключения к резервированным серверам.

4.3 Свойства входов/выходов данных, влияющих на обмен с сервером

Параметры объекта данных подписки задается при помощи следующих свойств входов из раздела ***Данные***.

Свойство ***Зона нечувствительности*** (prop_deadzone) позволяет задать абсолютную зону нечувствительности. Если значение в сервере изменилось больше, чем величина зоны нечувствительности, то оно передается клиенту.

Если задать ***Зона нечувствительности*** = **0** (или при отсутствии свойства), значение будет передаваться при изменении.

Свойство ***Период опроса*** (мс) (prop_sinter) задает период опроса (samplingInterval), если данное необходимо собирать в таймерном потоке.

Значение **0** (или отсутствие свойства) означает опрос каждый цикл работы подписки в фоне.

Значения **>0** должны быть кратны таймерному промежутку. Если значение не кратно, то оно принудительно «округляется» вниз. Тогда значение анализируется в каждом таймерном цикле и добавляется в очередь, если проходит проверку на зону нечувствительности. Для такой работы в проекте обязательно должен быть добавлен блок OpcUAServerTimer.

Свойство ***Размер очереди*** (prop_qsize) задает размер очереди (queueSize) данного в подписке. Применяется для сбора данных в таймере (см. свойство ***Период опроса***).

Свойство ***Интервал принудительной отправки (мс)*** (prop_reftime) позволяет организовать принудительную отправку данного в подписке независимо от изменения значения и настроек зоны нечувствительности. Заданный интервал отсчитывается от момента последней отправки значения. Это нестандартная функция, поэтому работает, только если в качестве сервера выступает блок OpcUAServer.

Свойство I***D источника/приемника*** (prop_srcid) - идентификатор объекта данных, который необходимо читать из сервера и записывать во вход раздела ***Данные***. Это свойство добавляется автоматически при выполнении команды ***Назначить источником***. Если свойство не добавлено, то клиент читает из сервера значение такого же входа (данный случай используется для синхронизации данных между дублированными модулями, подробнее см. документацию Синхронизация проектов и реализация резервирования. Библиотека paSync).

Выходы, добавленные в раздел ***Данные***, клиент записывает в сервер. Периодичность записи зависит от значения свойства ***Зона нечувствительности*** у выхода:

- Свойство отсутствует – запись производится каждый цикл работы клиента;
- **0** – запись производится при любом изменении значения;
- **>0** – запись производится, если значение изменилось больше, чем размер зоны нечувствительности.

ВНИМАНИЕ
Для корректной работы данных свойств входов/выходов необходимо добавить их в любой раздел внутри модуля и установить свойство модуля ***Трансляция: включить свойства входов/выходов*** = ***Только из разделов***.

5 Запись уставок с использованием OPC UA (BufSupEx)

5.1 Запись уставок OPC UA-сервера с удаленного клиента

Для записи уставок в блоки ***BufSupEx*** из библиотеки ***paCore*** с удаленного OPC UA-клиента используется блок UABufSups.

Блоки ***BufSupEx*** в проекте контроллера-сервера подключаются выходами **pkt** к входам **buf** блока ***UABufSups***.

Вход **inter** блока ***BufSupEx*** можно при необходимости подключить к другому блоку протокола (например, к блокам ***Modbus Slave***).

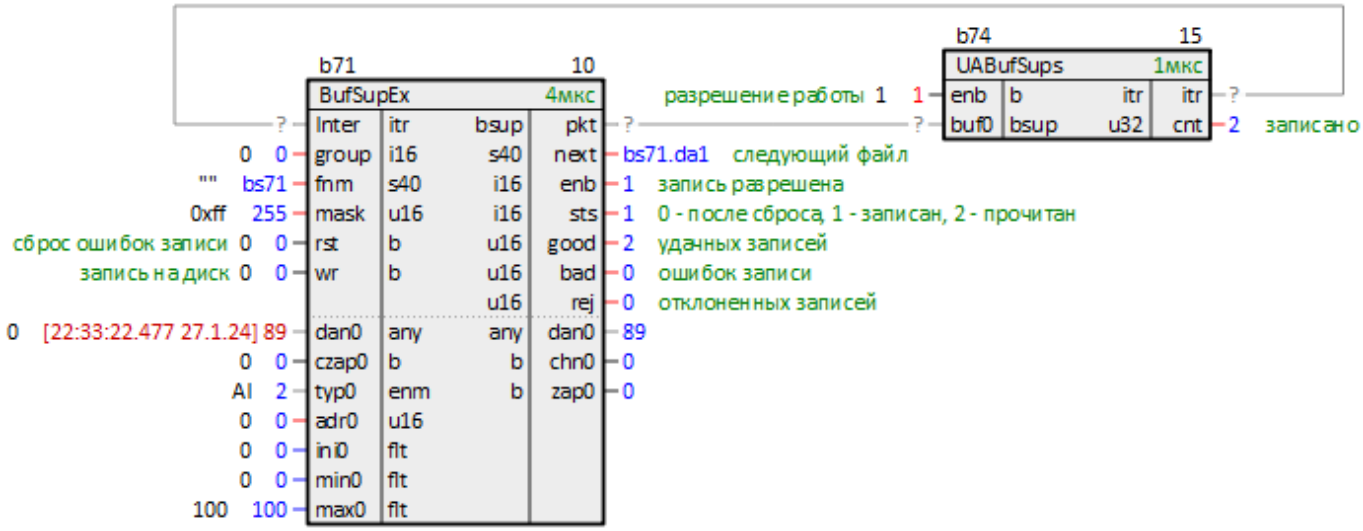


Рисунок 5.1 – Запись уставок с OPC UA-клиента

При отключении блока ***UABufSups*** (или при его отсутствии) для записи уставки с OPC UA-клиента потребуется подать импульс на входы **czap**.

Параметры на диске сохраняются в бинарных файлах с расширениями **.da1** и **.da2**.

ВНИМАНИЕ
При изменении числа входов блока ***BufSupEx*** файлы на диске перезаписываются.

Подробнее о возможностях и работе блока ***BufSupEx*** в документации Архивирование и сохранение уставок.

5.2 Синхронизация записи уставок между контроллерами

Для синхронизации записи уставок блоков ***BufSupEx*** из библиотеки ***paCore*** между двумя контроллерами требуется дублировать во второй контроллер программу (или целиком место работы), в которой добавлен блок ***BufSupEx***.

Для этого следует перетащить на модуль второго контроллера (оба модуля должны быть в одном проекте) требуемую программу, в выпадающем меню выбрать ***Добавить***.

Обе программы подсветятся желтым. Теперь все изменения на страницах данной программы будут одинаково применены в обоих модулях.

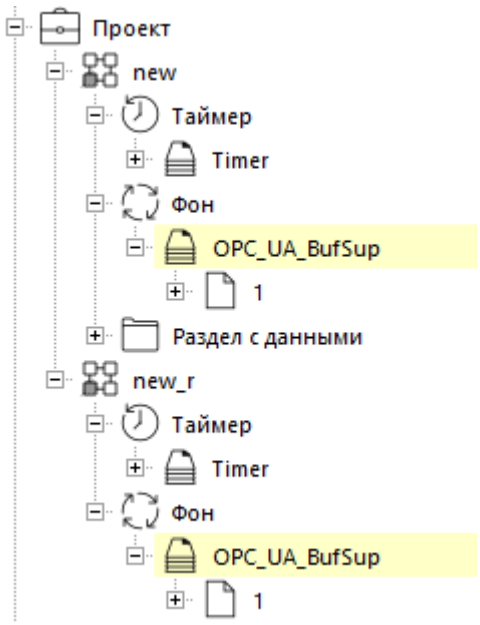


Рисунок 5.2 – Дублированные программы

Блоки ***BufSupEx*** в данной программе следует соединить выходами **pkt** с входами **buf** блока OpcUAClient.

Входы ***OpcUAClient***, отвечающие за настройку обмена (IP-адреса и порты) следует задавать с помощью SQL-запросов к соответствующим свойствам модулей. Для задания параметров «соседнего» контроллера рекомендуется использовать пользовательские свойства. Примеры SQL-запросов к свойствам модуля приведены в разделе 4.

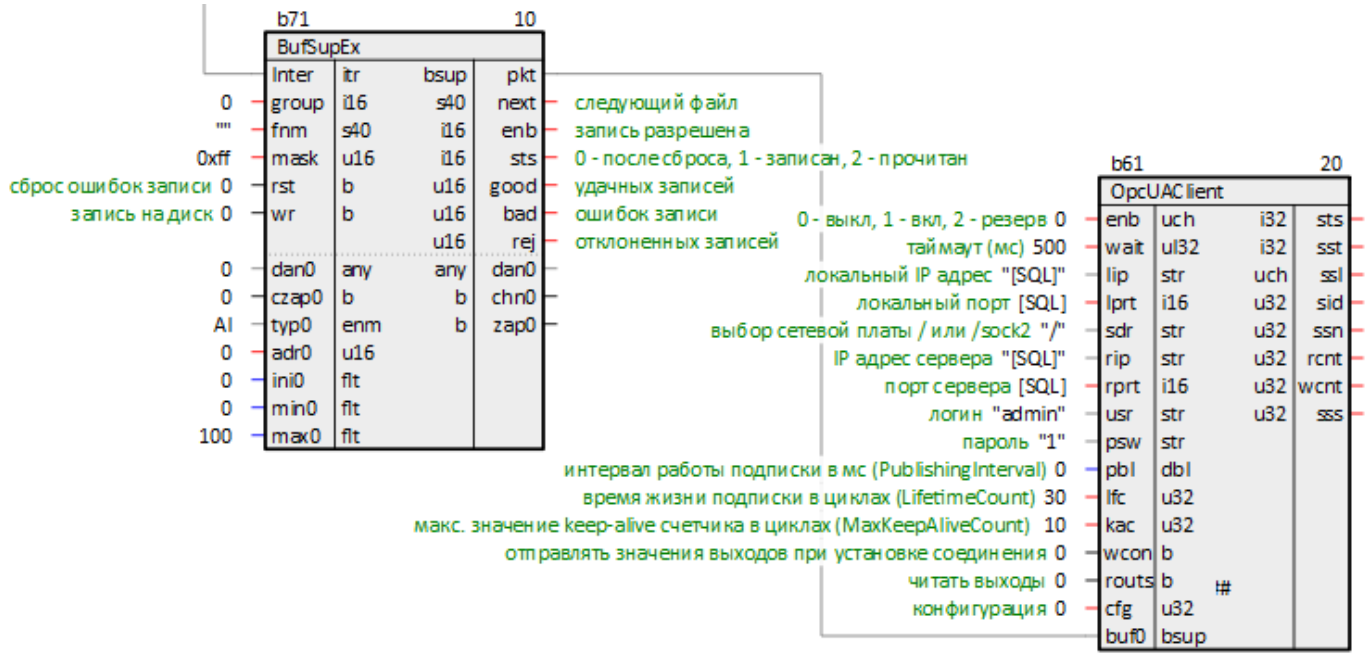


Рисунок 5.3 – Подключение BufSupEx к OpcUAClient

После запуска программ на обоих контроллерах один из них следует назначить «ведущим» – отключить ***OpcUAClient*** (**enb** = 0), а второй «ведомым» – включить ***OpcUAClient*** (**enb** = 1).

Теперь при изменении уставок ведущего контроллера изменения будут дублироваться в ведомый посредством чтения их OPC UA-клиентом.

При попытке изменить уставки ведомого контроллера изменения в ведущем контроллере не применятся.

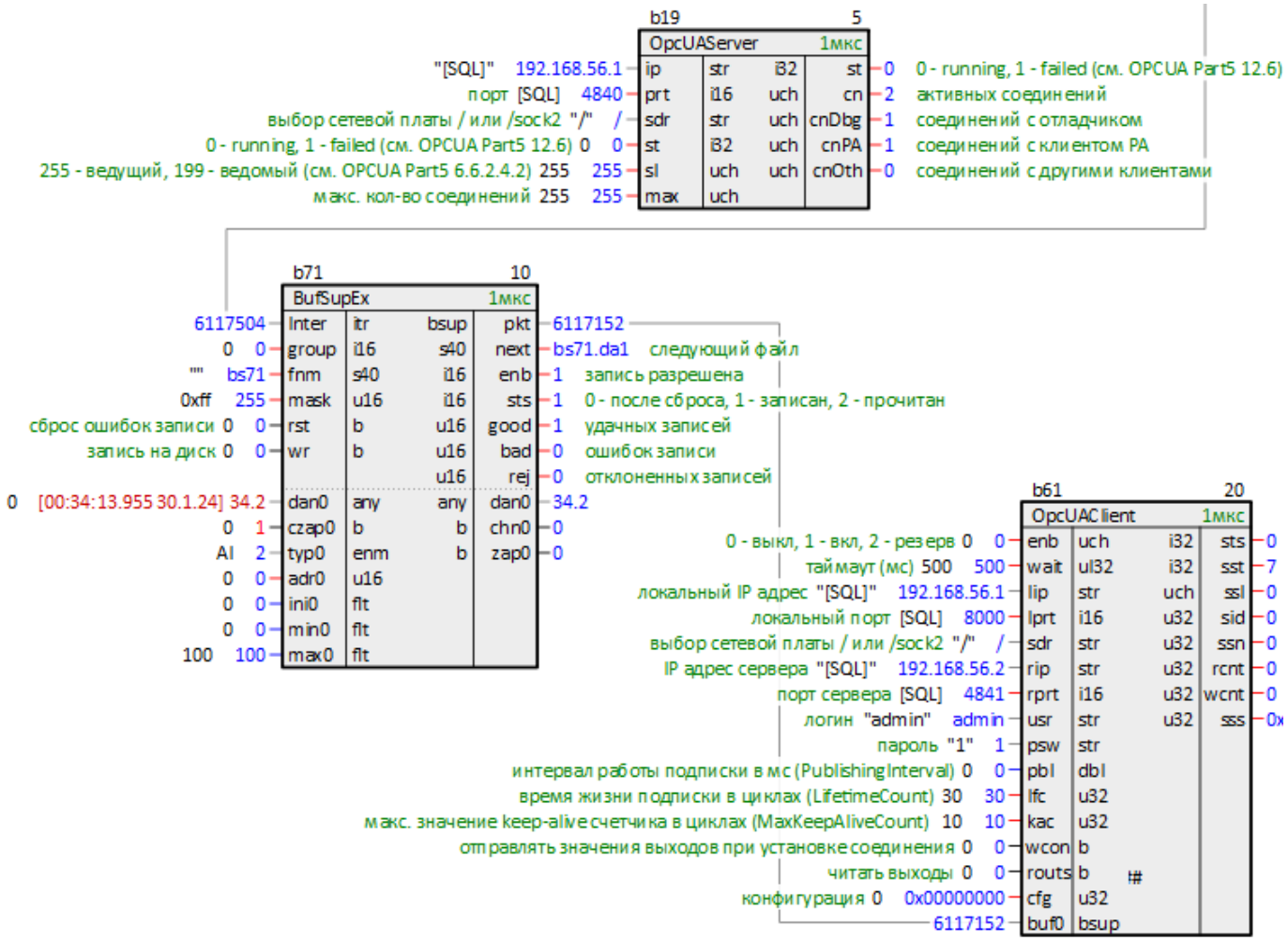


Рисунок 5.4 – Изменение уставки с ведущего контроллера

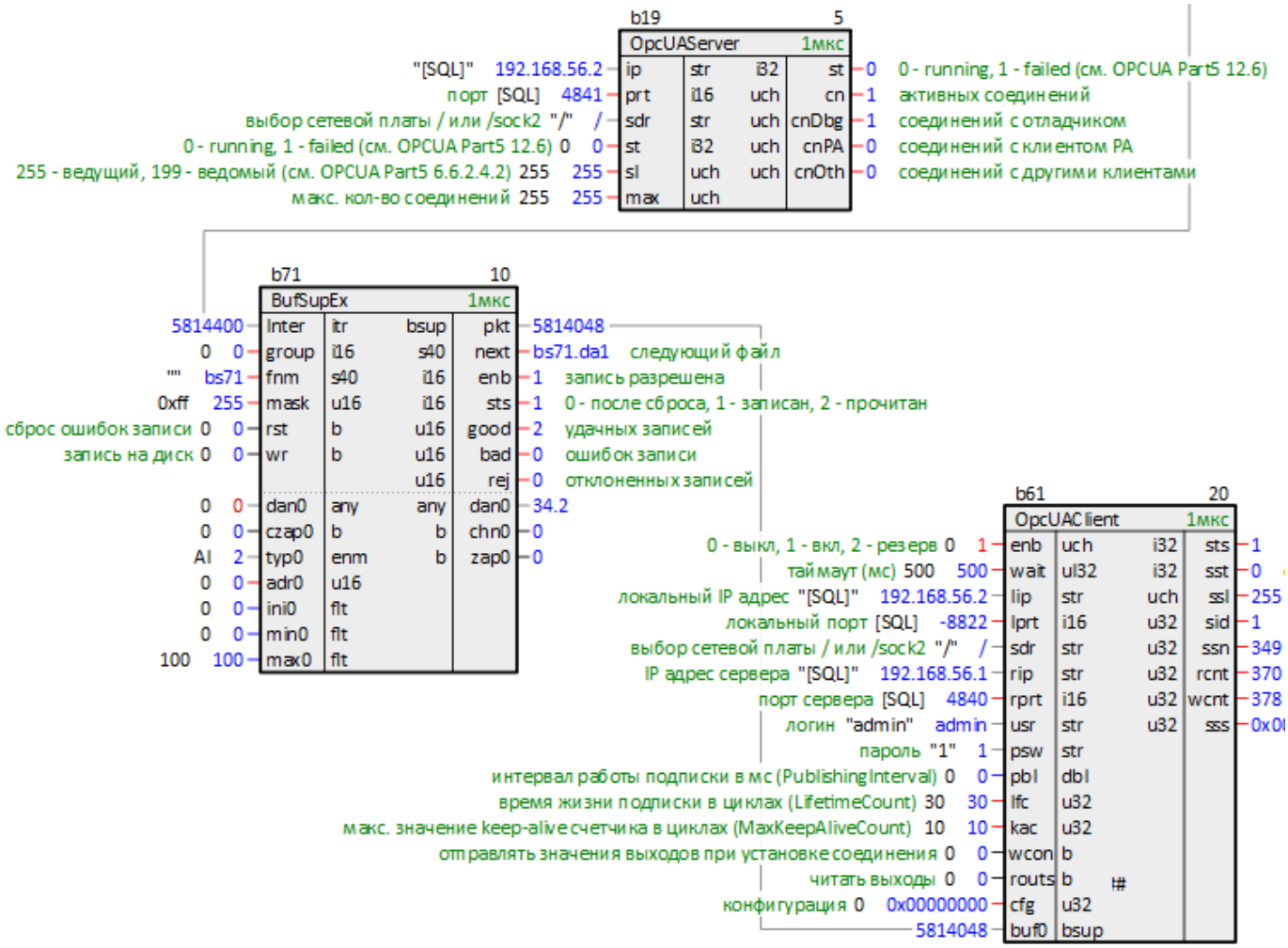


Рисунок 5.5 – Ведомый контроллер

Параметры на диске сохраняются в бинарных файлах с расширениями **.da1** и **.da2**.

ВНИМАНИЕ
При изменении числа входов блока ***BufSupEx*** файлы на диске перезаписываются.

Подробнее о возможностях и работе блока ***BufSupEx*** в документации Архивирование и сохранение уставок.

6 Настройка обмена с использованием технологии OPC

6.1 OPC UA-сервер. Пример подключения к MasterSCADA 4D

В данном примере будет рассмотрено подключение к OPC UA-серверу ПЛК210 клиентом MasterSCADA 4D (версия 1.3.2.32723).

Для настройки обмена следует:

1. Добавить в проект блок OpcUAServer, настроить его входы.

   В примере используется блок OPC UA-сервера, автоматически добавляемый в новый проект для подключения ***Отладчиком***.

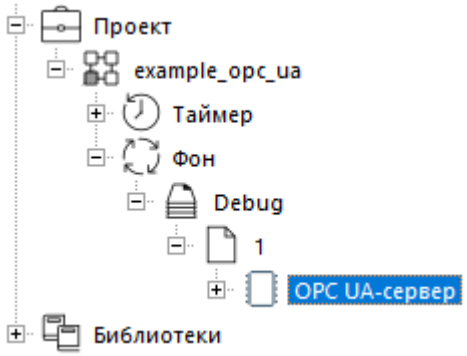


Рисунок 6.1 – Дерево проекта

На входы блока **ip** – IP-адрес контроллера и **prt** – локальный порт контроллера поданы SQL-запросы к соответствующим свойствам текущего модуля.

Запрос IP адреса (prop_ip):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Запрос номера порта (prop_debug_port):

```
<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_debug_port"</sql>
```

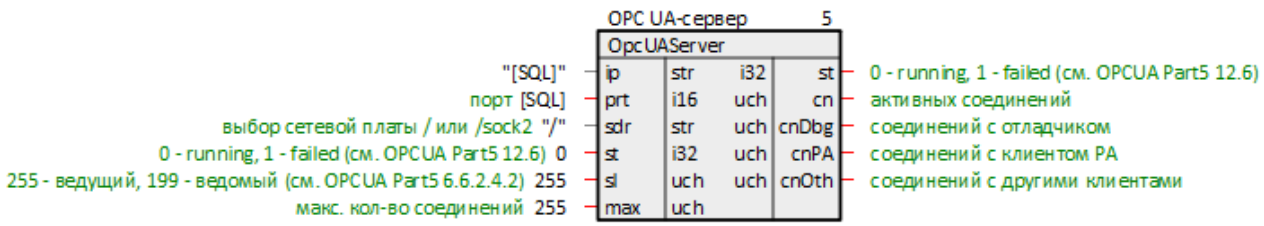


Рисунок 6.2 – Блок OPC UA-сервера

В данной конфигурации OPC UA-клиент будет видеть все входы/выходы блоков в проекте. Для удобства выбора переменных для обмена можно создать ***Раздел*** и добавить в него необходимые входы/выходы блоков в проекте.

Для этого необходимо создать в модуле ***Раздел*** с именем *Данные OPC UA-сервер* (или любым другим).

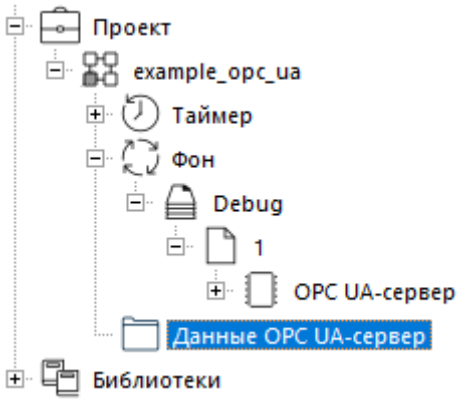


Рисунок 6.3 – Дерево проекта

2. Добавить в раздел необходимые входы/выходы блоков в проекте. Для этого необходимо выделить интересующий вход/выход блока, с нажатым **Ctrl** перенести данный вход/выход в созданный раздел **ДанныеOPC UA-сервер**, и в появившемся меню выбрать ***Добавить***.

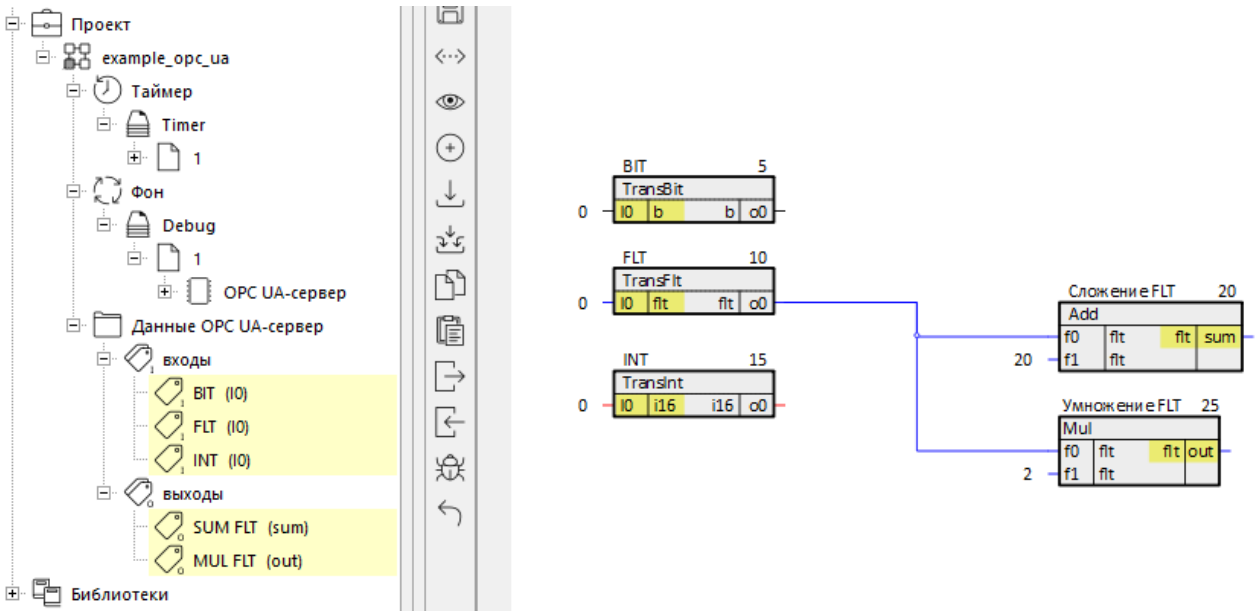


Рисунок 6.4 – Раздел Данные OPC UA-сервер

3. Транслировать модуль и запустить программу на ПЛК. Перейти к настройке OPC UA-клиента MasterSCADA 4D.
4. Создать новый пустой проект MasterSCADA 4D.
5. В узле ***Система*** добавить узел ***АРМ***.

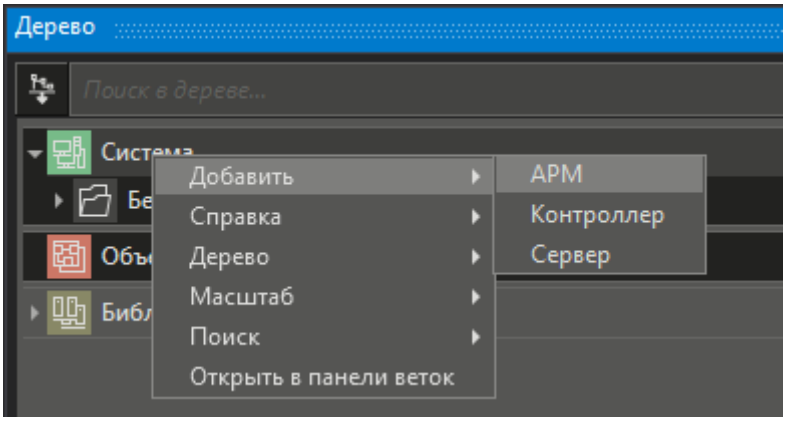


Рисунок 6.5 – Узел АРМ

6. В созданном узле ***АРМ1*** добавить протокол ***OPC UA***.

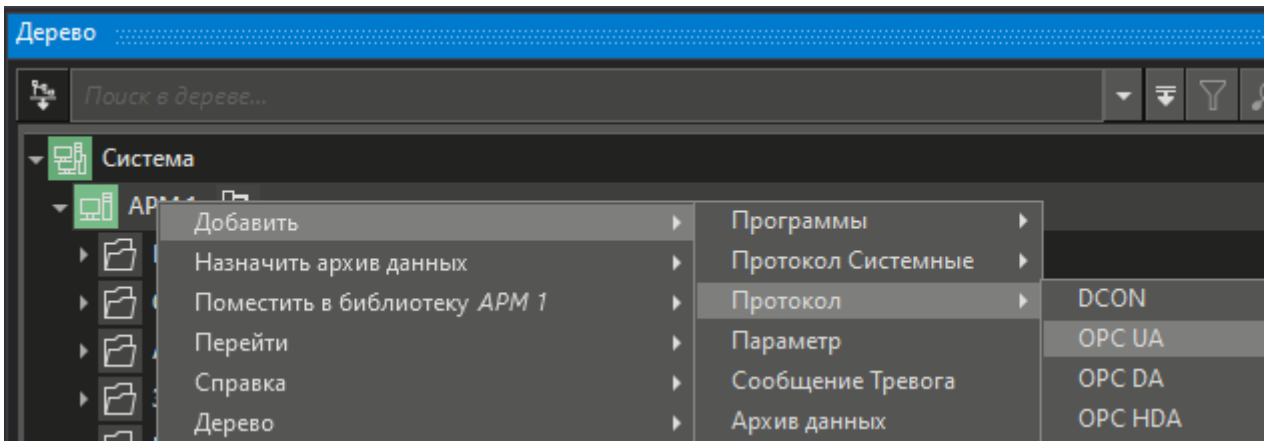


Рисунок 6.6 – Узел АРМ, протокол OPC UA

7. В свойствах ***OPC UA*** задать адрес ***URI*** OPC UA-сервера в соответствии с блоком OpcUAServer, ***Имя пользователя*** и ***Пароль*** в соответствии со свойствами модуля (в примере модуль *example_opc_ua*).

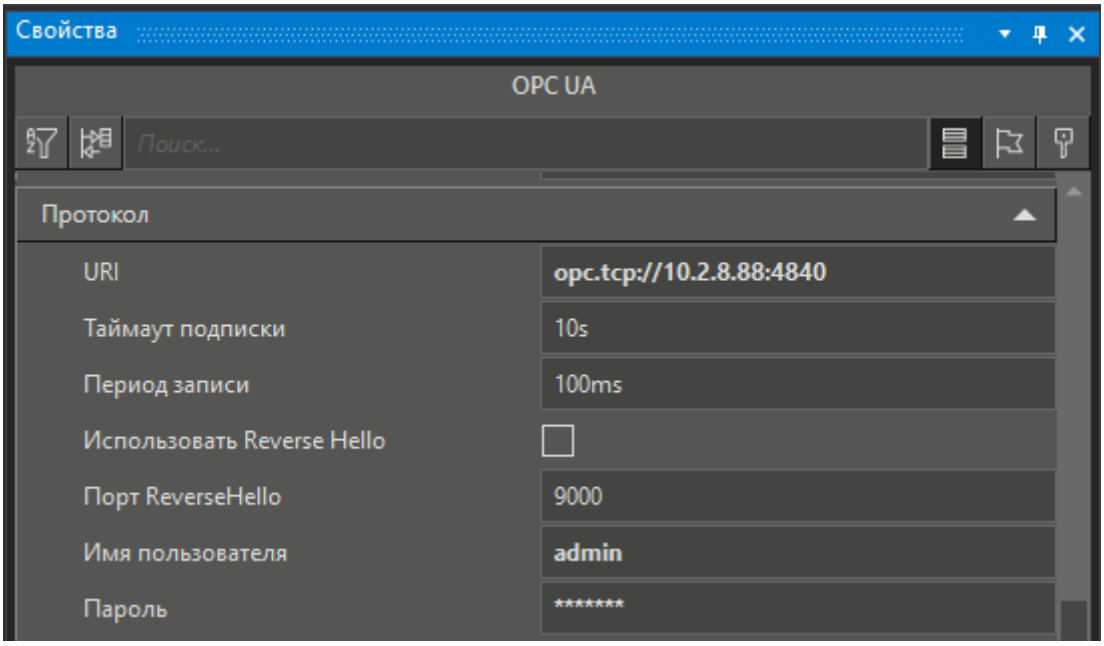


Рисунок 6.7 – Свойства OPC UA

8. В окне ***OPC UA*** нажать ***Подключиться без загрузки***. В окне отобразятся все переменные дерева проекта контроллера. Далее необходимо выбрать интересующие переменные и нажать ***Применить***. В узле ***OPC UA*** появится раздел с добавленными переменными.

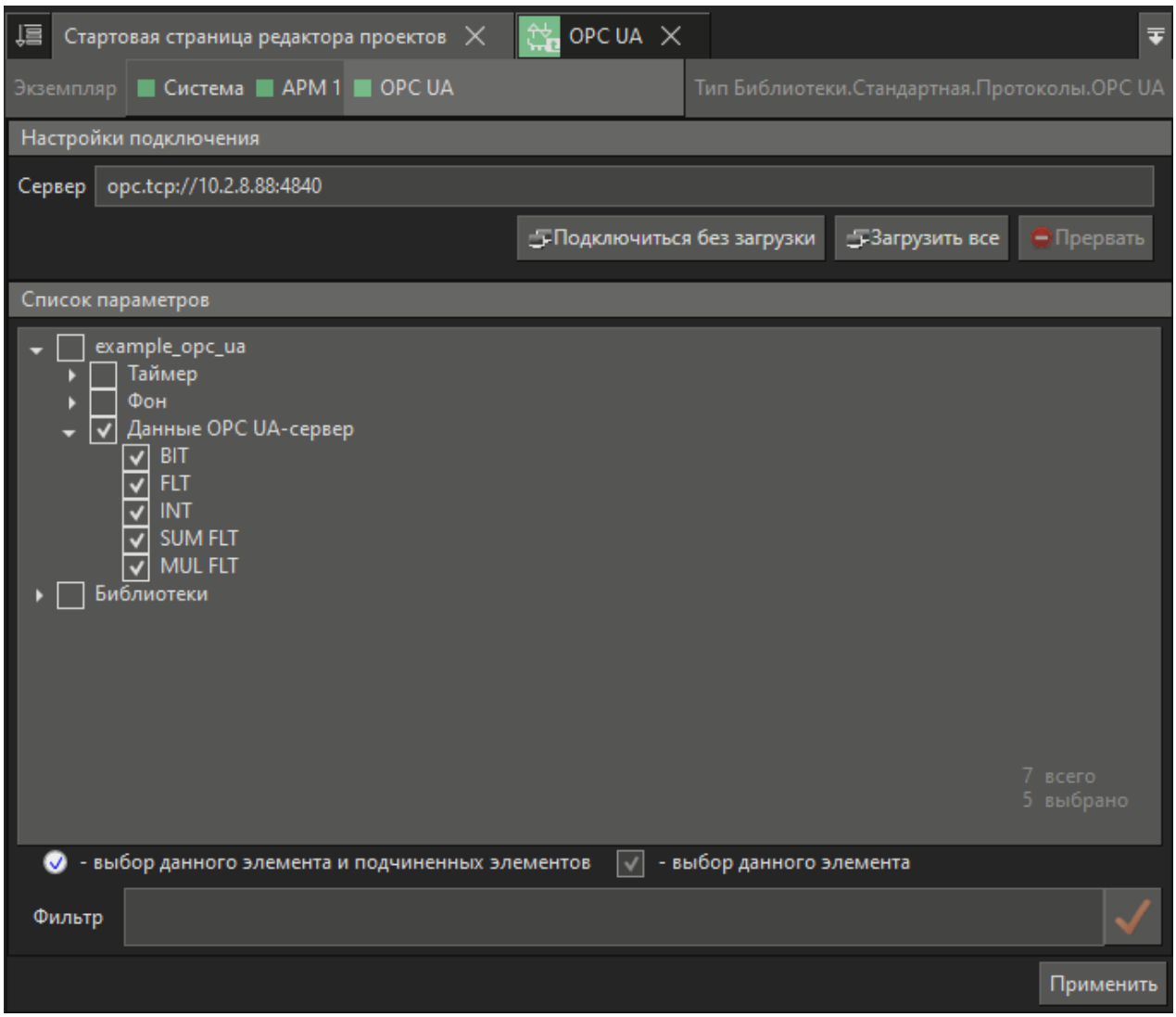


Рисунок 6.8 – Окно OPC UA, выбор переменных

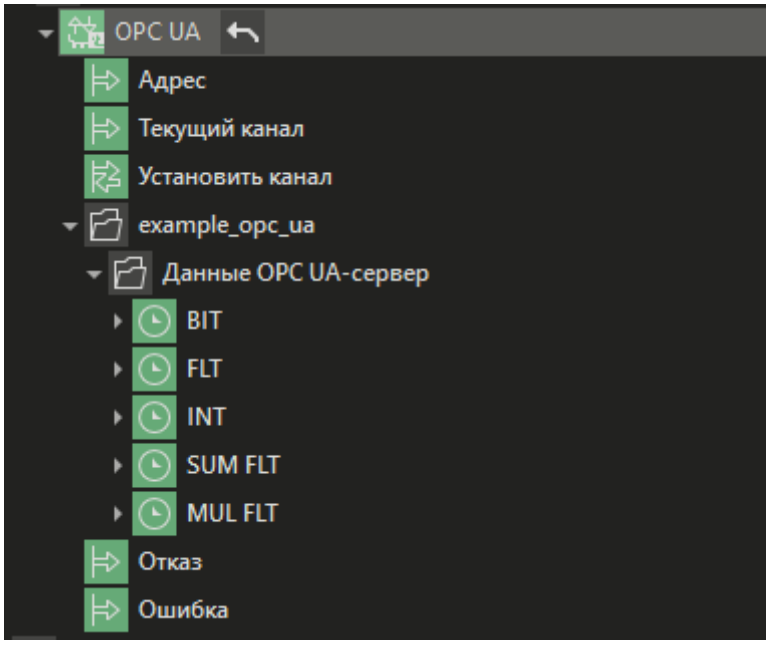


Рисунок 6.9 – Узел OPC UA

9. Подключить узел ***АРМ1***, наблюдать корректный обмен данными.

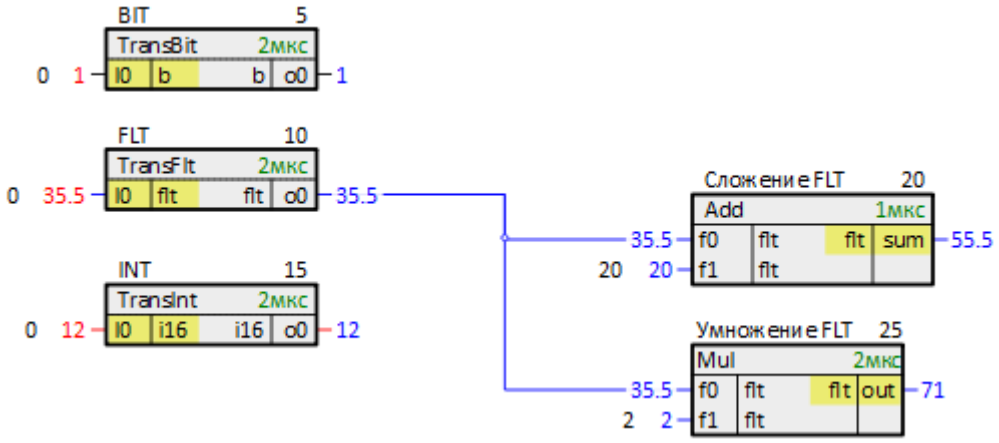


Рисунок 6.10 – Изменение данных в программе ПЛК

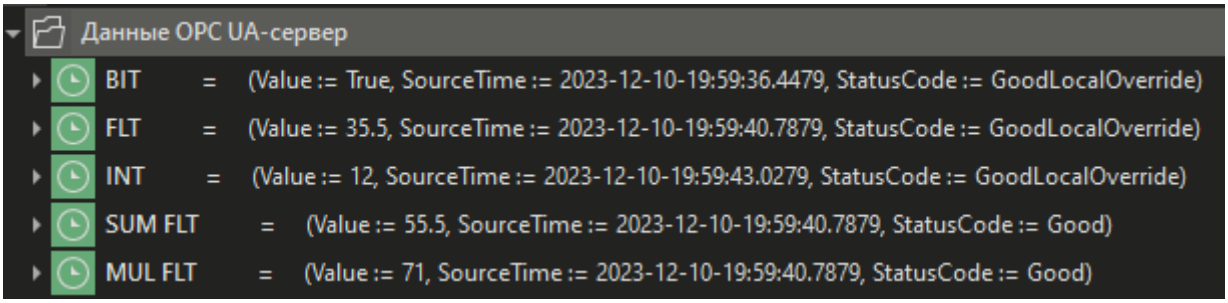


Рисунок 6.11 – Чтение данных OPC UA-клиентом

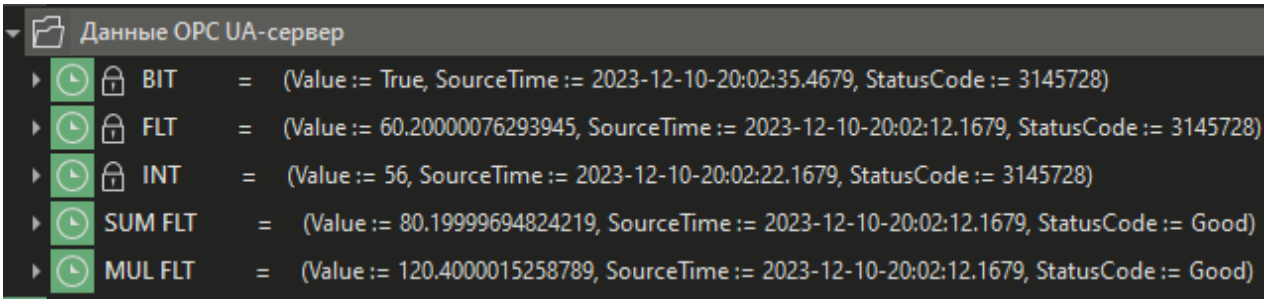


Рисунок 6.12 – Запись данных OPC UA-клиентом

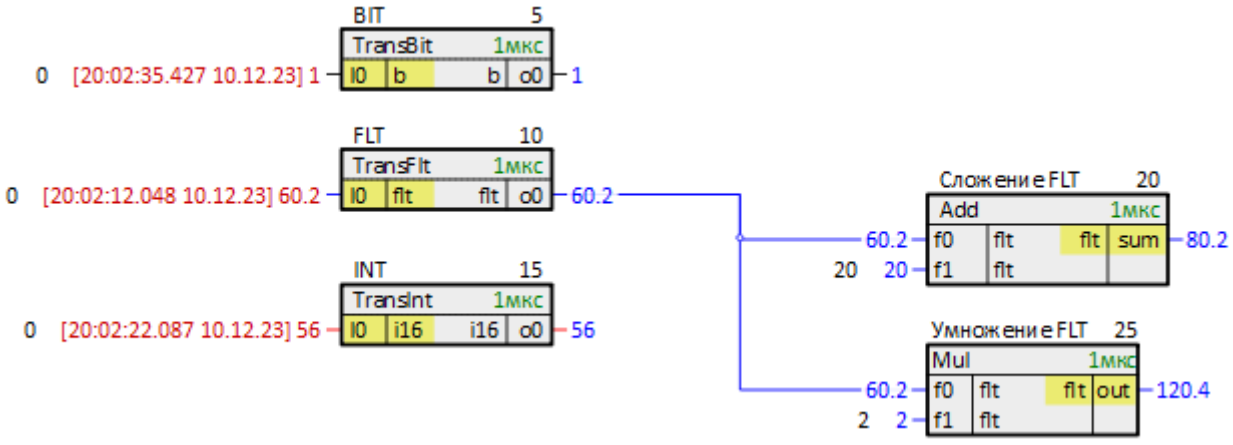


Рисунок 6.13 – Изменение данных в программе ПЛК

6.2 OPC UA-клиент. Пример подключения к OPC UA-серверу ПЛК210

В данном примере будет рассмотрено подключение ПЛК210 в качестве клиента к OPC UA-серверу на ПЛК210. В качестве OPC UA-сервера использован сервер, настроенный в предыдущем примере.

Для настройки обмена в качестве OPC UA-клиента следует:

1. Добавить в проект блок OpcUAClient, настроить его входы.

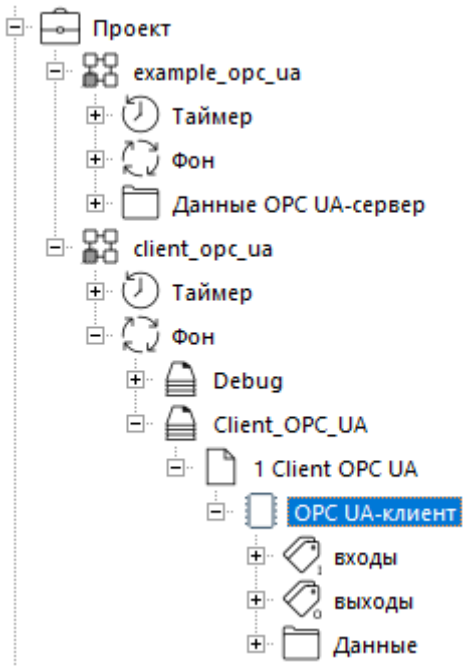


Рисунок 6.14 – Дерево проекта

На вход блока **Iip** – локальный IP-адрес контроллера подан SQL-запрос к соответствующему свойству текущего модуля.

Запрос IP-адреса (prop_ip):

```
"<sql>SELECT value FROM blocks_prop WHERE indx=:module AND type="prop_ip"</sql>"
```

Внутри блока ***OpcUAClient*** автоматически добавляется раздел ***Данные***.

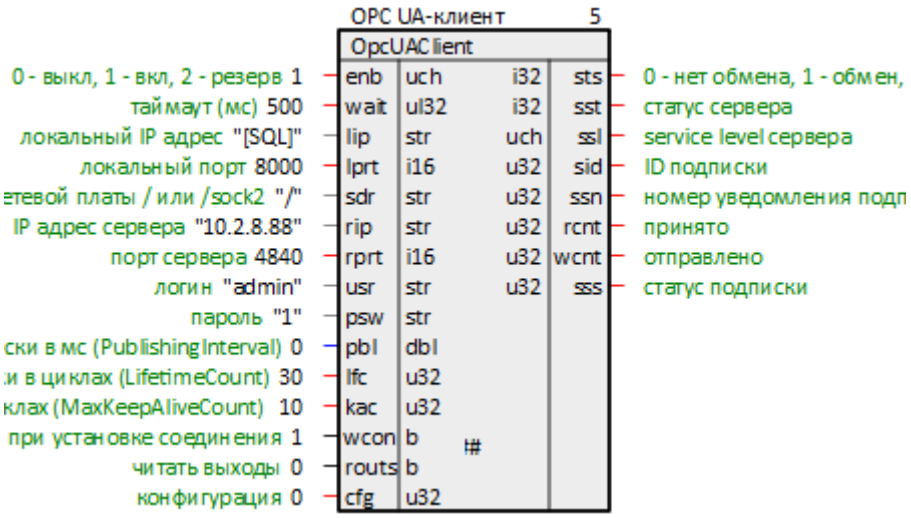


Рисунок 6.15 – Блок OPC UA-клиента

2. У OPC UA-сервера из предыдущего примера три входа и два выхода, что в разделе ***Данные*** клиента будет соответствовать трем выходам и двум входам (входы/выходы сервера можно связывать также с входами/выходами клиента в разделе Данные).

   Чтобы отображать/изменять данные в модуле клиента, можно создать на любой странице проекта терминальные блоки:

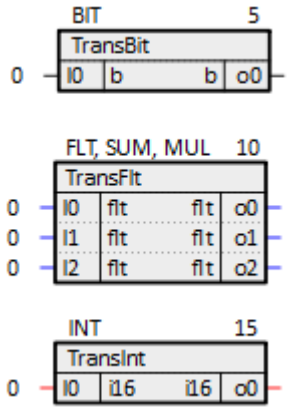


Рисунок 6.16 – Терминальные блоки

3. Добавить входы/выходы созданных терминальных блоков в раздел ***Данные***. Для этого, зажав **Ctrl**, перетащить вход/выход в раздел ***Данные*** и в выпадающем меню выбрать ***Добавить***.

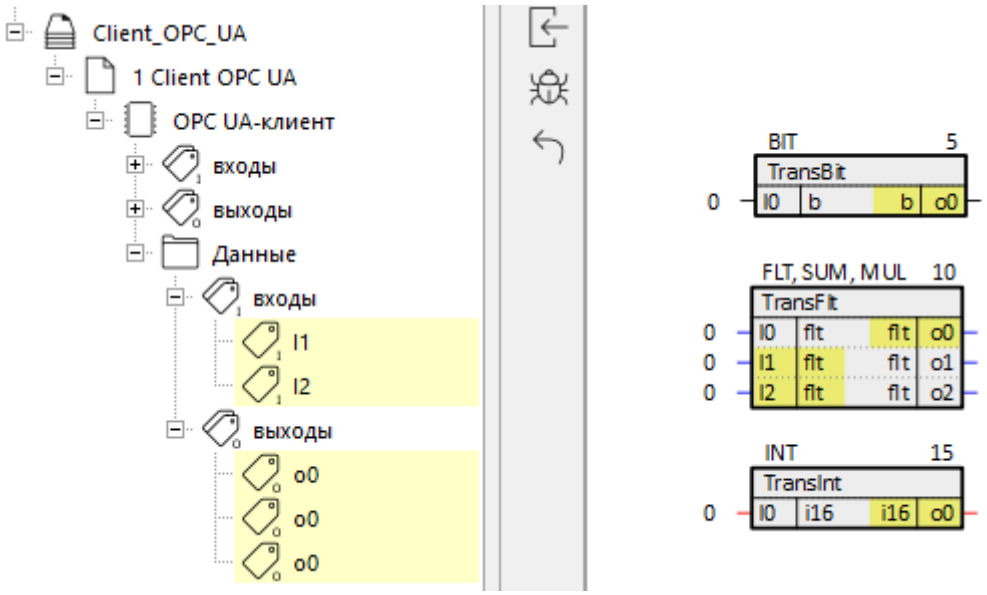


Рисунок 6.17 – Раздел Данные

4. Для того, чтобы привязать данные OPC UA-сервера к созданным входам/выходам раздела ***Данные***, следует назначить их источниками. Для этого следует открыть страницу модуля сервера с интересующими входами/выходами, зажав **Ctrl**, перетащить вход/выход к выходу/входу в разделе ***Данные***, в выпадающем меню выбрать ***Назначить источником***.

   То же самое можно сделать, перетаскивая вход/выход сервера из дерева проекта без зажатого **Ctrl**.

   В данной конфигурации клиент будет записывать данные входов сервера циклически. Чтобы записывать данные по изменению, следует выполнить действия из пп. 5 - 6 .

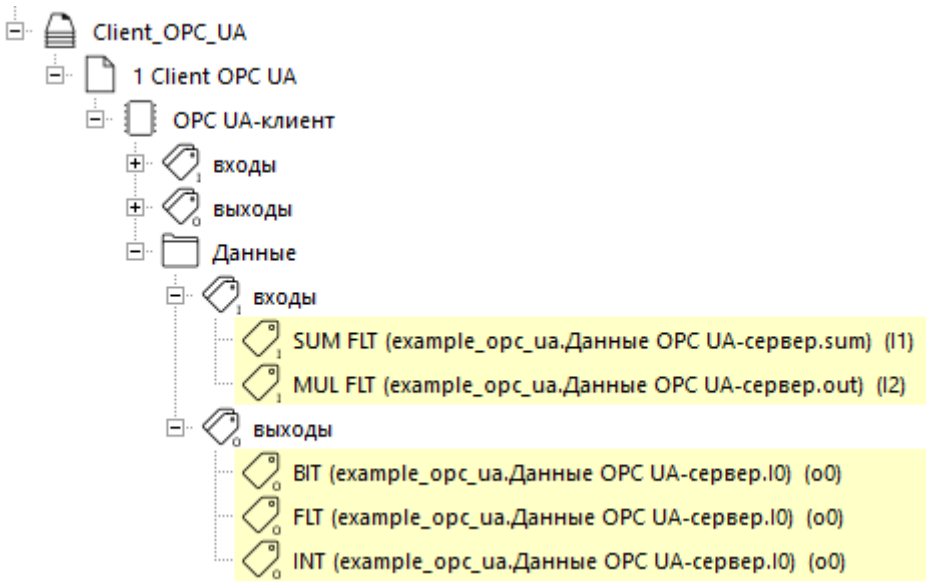


Рисунок 6.18 – Раздел Данные

5. Добавить выходам раздела ***Данные*** свойство ***Зона нечувствительности***, равное **0**.

   Если свойство ***Зона нечувствительности*** имеет значение **0**, то значение на выход передается по изменению. Если свойство имеет значение больше 0, то значение на выход передается при преодолении данной зоны нечувствительности.

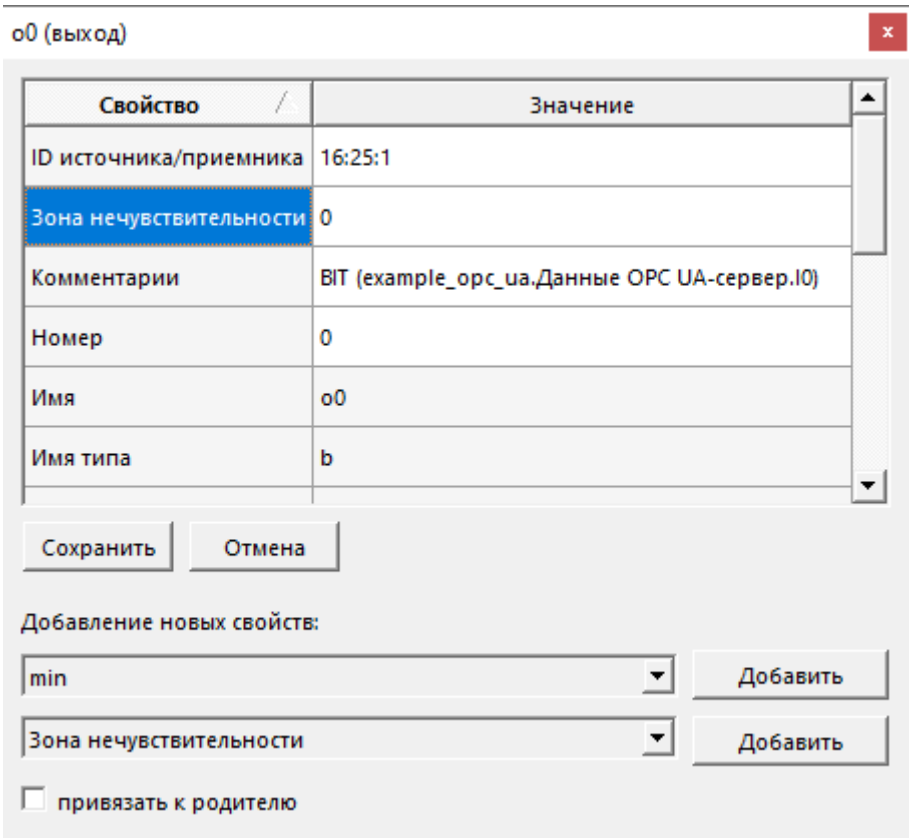


Рисунок 6.19 – Зона нечувствительности

6. В свойствах модуля добавить свойство ***Трансляция: включить свойства входов/выходов***, равное ***Только из разделов***.

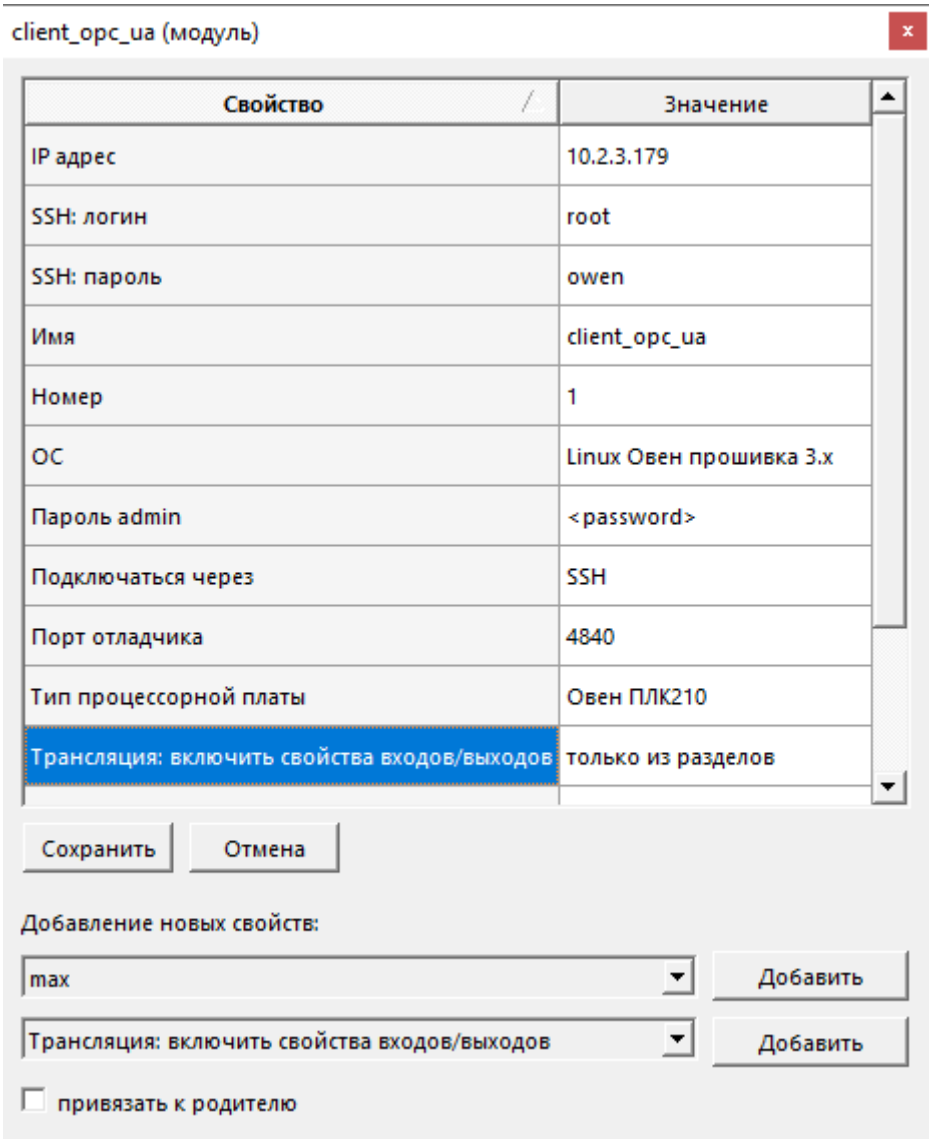


Рисунок 6.20 – Свойства модуля OPC UA-клиента

7. Транслировать модуль OPC UA-клиента и запустить программу на обоих ПЛК.
8. Подключиться ***Отладчиком*** к обоим модулям, наблюдать корректный обмен данными.

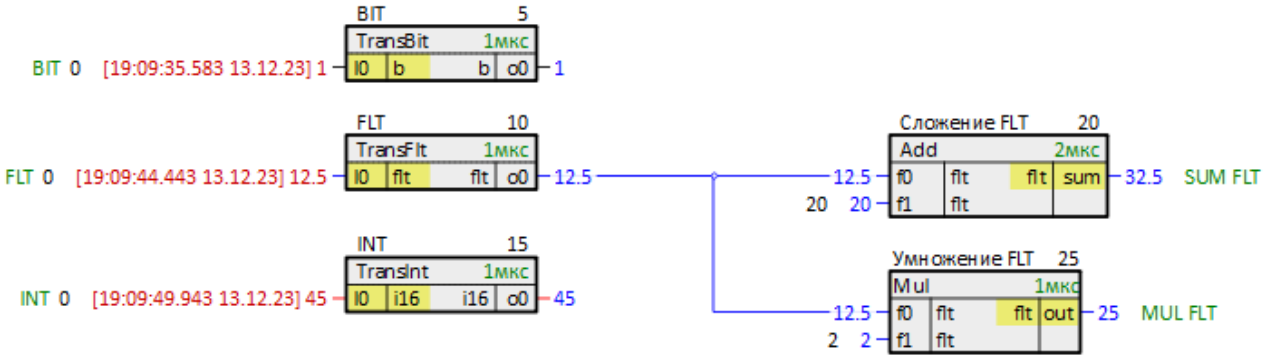


Рисунок 6.21 – Изменение данных в программе ПЛК-сервера

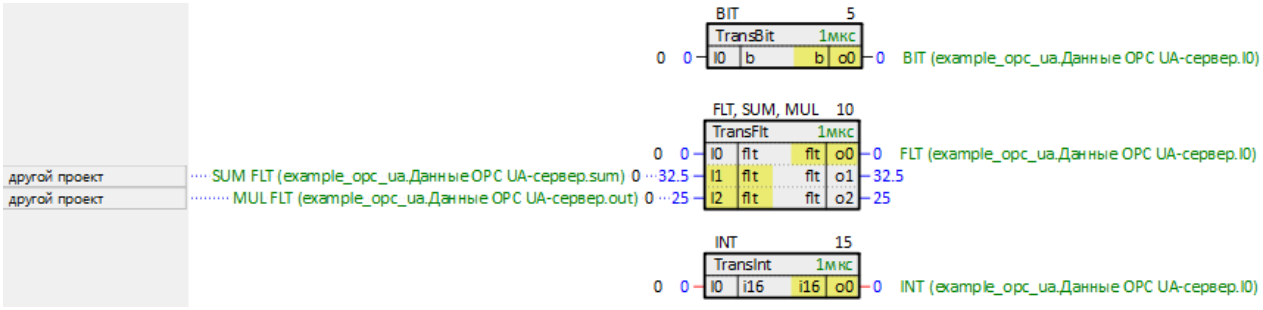


Рисунок 6.22 – Чтение данных ПЛК-клиентом

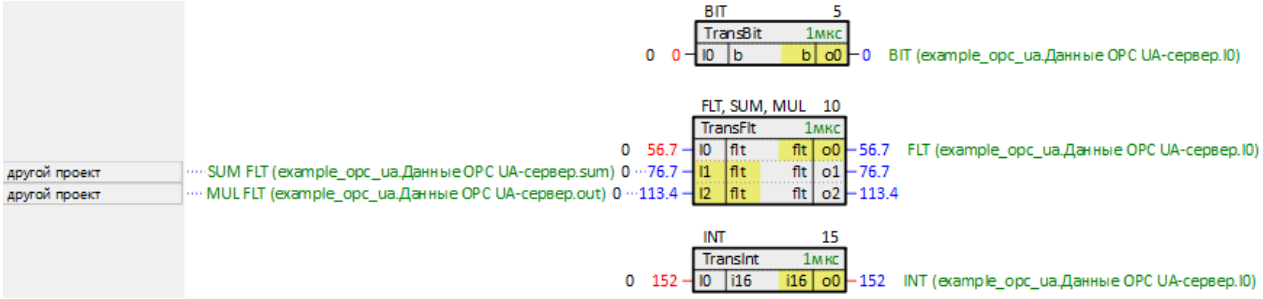


Рисунок 6.23 – Запись данных ПЛК-клиентом

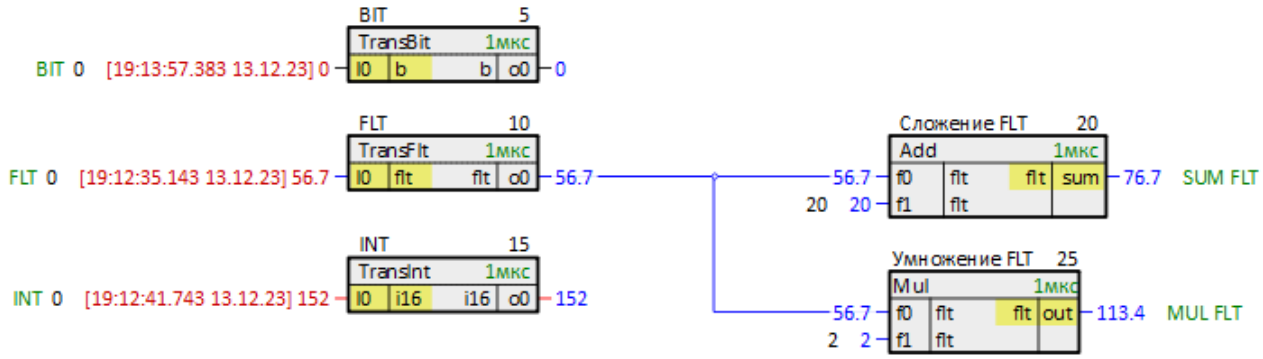


Рисунок 6.24 – Изменение данных в программе ПЛК-сервера после записи ПЛК-клиентом

Приложение А. Поддержка сервисов по спецификации OPC UA

Таблица А.1 – Поддержка сервисов по спецификации OPC UA

Service Set/Service	Поддержка	Комментарии
Discovery Service Set		
FindServers	Да	
GetEndpoints	Да	
Session Service Set		
CreateSession	Да	
ActivateSession	Да	В качестве UserIdentityToken поддерживаются только: AnonymousIdentityToken (анонимный доступ) и UserNameIdentityToken (доступ с логином и паролем)
CloseSession	Да	
Cancel	Нет	
NodeManagement Service Set	Нет	
View Service Set		
Browse	Да	В качестве displayName используется свойство: ***Полный алиас***, или ***Комментарии***, если ***Полный алиас*** пустой, или ***Имя***, если ***Комментарии*** и ***Полный алиас*** пустые. В качестве BrowseName используется индекс узла в шестнадцатеричном формате
BrowseNext	Да	
TranslateBrowsePathsToNodeIds	Да	Внутри RelativePath игнорируются поля referenceTypeId, isInverse и includeSubtypes. В качестве targetName используется свойство: ***Полный алиас***, или ***Комментарии***, если ***Полный алиас*** пустой, или ***Имя***, если ***Комментарии*** и ***Полный алиас*** пустые
RegisterNodes	Да	Ничего не делает
UnregisterNodes	Да	Ничего не делает
Query Service Set	Нет	
Attribute Service Set		
Read	Да	Параметр maxAge игнорируется, сервер всегда возвращает текущее значение
HistoryRead	Нет	
Write	Да	Поддерживается запись только значений (атрибут Value). Запись отдельных элементов массива не поддерживается
HistoryUpdate	Нет	
Method Service Set		
Call	Да	
MonitoredItem Service Set		
CreateMonitoredItems	Да	Из фильтров поддерживается только DataChangeFilter, и только с абсолютной зоной нечувствительности
ModifyMonitoredItems	Да	
SetMonitoringMode	Да	
SetTriggering	Нет	
DeleteMonitoredItems	Да	
Subscription Service Set		
CreateSubscription	Да	

Продолжение таблицы А.1

Service Set/Service	Поддержка	Комментарии
ModifySubscription	Да	
SetPublishingMode	Да	
Publish	Да	
Republish	Да	Размер очереди равен 10
TransferSubscriptions	Да	
DeleteSubscriptions	Да	

Россия, 111024, Москва, 2-я ул. Энтузиастов, д. 5, корп. 5
тел.: +7 (495) 641-11-56, факс: (495) 728-41-45
тех. поддержка 24/7: 8-800-775-63-83, support@owen.ru
отдел продаж: sales@owen.ru
Веб-сайт ООО "ПромАвтоматика-Софт": www.pa.ru
рег.:1-RU-134175-1.2