НТО Терси-КБ

Каскад-САУ 4.0

Руководство пользователя

ГУКН.505290.000 04 90 9003

Саров
2023 г.

Оглавление

1 Введение

1.1 Что такое Каскад-САУ

Каскад-САУ - пакет приложений для создания систем управления и телемеханики, работающих в реальном времени.

Каскад-САУ может использоваться как исполнительная система программируемых контроллеров и систем сбора данных (PLC). Каскад-САУ позволяет организовать считывание данных с различных устройств, обработку данных и управление множеством устройствами по заданным пользователем алгоритмам.

Каскад-САУ может использоваться как система диспетчерского управления (SCADA). Каскад-САУ позволяет организовать отображение считанных с устройств данных на экране монитора в виде мнемосхем, графиков, списков событий и тревог, принимать команды управления от оператора и передавать их на исполнительные устройства.

Каскад-САУ позволяет организовать архивирование считанных с устройств данных в базе данных и передачу этих данных в другие системы управления.

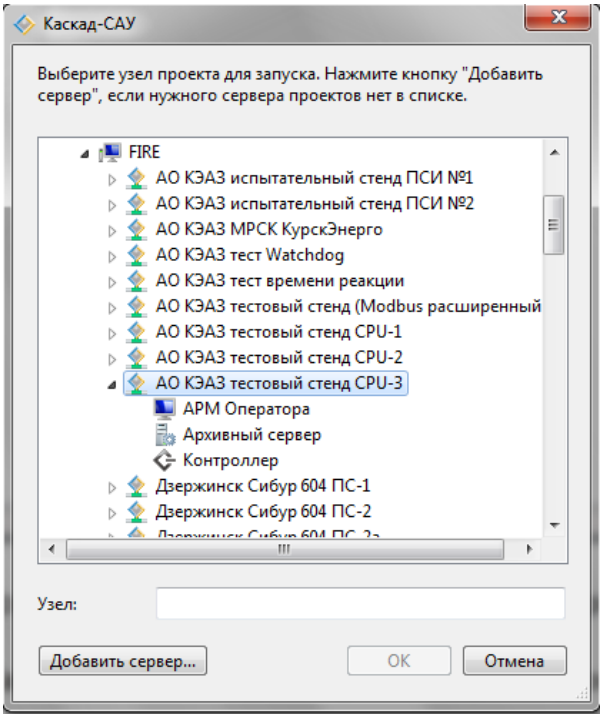
1.2 Состав Каскад-САУ

В состав Каскад-САУ входят среда разработки, сервер проектов и среда исполнения.

Среда разработки - программа для создания и редактирования проектов Каскад-САУ. Проект содержит конфигурацию среды исполнения Каскад-САУ, включая описание устройств ввода-вывода, пользовательское алгоритмы обработки данных, мнемосхемы визуализации данных, правила формирования событий, параметры архивирования и серверов данных.

Сервер проектов Каскад-САУ - программа для управления проектами и архивами Каскад-САУ.

Среда исполнения Каскад-САУ - набор программ и драйверов, обеспечивающих работу контроллеров, АРМ оператора, архивных серверов и серверов данных. Функции, которые выполняет среда исполнения, определяются конфигурацией из проекта.



1.3 Чем не является Каскад-САУ

Каскад-САУ плохо подходит для создания систем, работающих по расписанию, например, для систем сбора и анализа архивных данных с приборов учета расхода электроэнергии, тепла или газа.

Каскад-САУ не является MES-системой.

1.4 Поддерживаемые операционные системы

Среда исполнения Каскад-САУ поддерживает операционные Windows, Linux и eCos, (ОСРВ применяемая в контроллерах ВСЕ-5, ВСР-А9 производства НТО Терси-КБ).

Поддерживаемый функционал среды исполнения одинаков под разными операционными системами. Исключение составляют мнемосхемы и архивирование (только Windows), а также драйверы специализированных устройств (шина PLC4, DK8070 только для контроллеров ВСЕ-5 и ВСР-А9).

Среда разработки Каскад-САУ и сервер проектов Каскад-САУ поддерживают только операционную систему Windows.

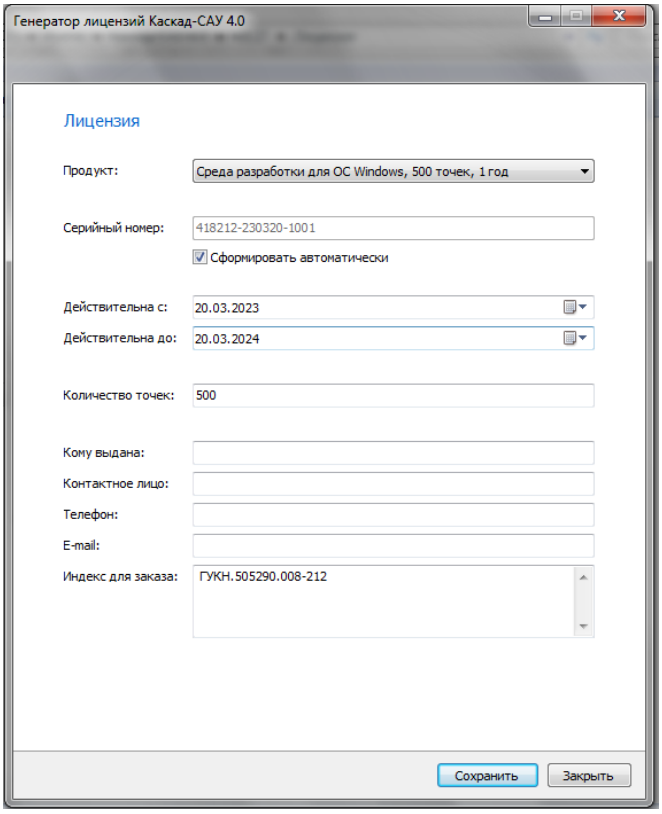
1.5 Лицензирование Каскад-САУ

Для работы Каскад-САУ требуется лицензия.

Лицензия - электронный документ, дающий право на использование Каскад-САУ на одном компьютере, контроллере или сервере. Для среды разработки, сервера проектов и среды исполнения требуются отдельные лицензии.

Лицензия на среду разработки определяет максимальное количество точек в проекте. Если количество точек проекта больше количества, определено лицензией, то среда разработки открывает проект в режиме "только для чтения" и не дает вносить в него изменения.

Действие лицензии на среду разработки может быть ограничено по времени. Например, годовая лицензия.



Лицензии на сервер проектов и среду исполнения Каскад-САУ не имеют ограничений на количество точек и ограничения по времени. Сервер проектов и среда исполнения будут работать с любыми проектами с любым количеством точек неограниченное время. Даже если закончится срок действия лицензии на среду разработки, и внесение изменений в проект станет невозможным, сервер проектов и среда исполнения продолжат работу с этим проектом без каких-либо ограничений.

В составе Каскад-САУ поставляется бесплатная лицензия на среду разработки на 100 точек без ограничения действия по времени, а также бесплатные лицензии на сервер проектов и среду исполнения. Купленные лицензии следует активировать вручную после установки Каскад-САУ (см. п. 2.3).

2 Установка Каскад-САУ

2.1 Системные требования

Для работы *среды разработки* и *сервера проектов* Каскад-САУ необходимо, чтобы аппаратные и программные средства компьютера соответствуют приведенным ниже характеристикам:

- центральный процессор не ниже Intel Core i3;
- оперативная память не менее 2 Гбайт;

- не менее 100 Мбайт свободного пространства на жестком диске;
- графический адаптер, поддерживающий разрешение экрана не ниже 1280x1024 точек;
- манипулятор типа «мышь» или другое совместимое указательное устройство;
- операционная система Microsoft Windows 7 или более новая.

Для работы *среды исполнения* Каскад-САУ необходимо, чтобы аппаратные и программные средства компьютера соответствуют приведенным ниже характеристикам:

- центральный процессор не ниже Intel Pentium 4;
- оперативная память не менее 64 Мбайт;
- не менее 30 Мбайт свободного пространства на жестком диске;
- операционная система Microsoft Windows XP или более новая, Linux.

2.2 Установка Каскад-САУ для Windows

Установка Каскад-САУ для операционной системы Windows выполняется с помощью программы установки. Программа установки Каскад-САУ поставляется в виде дистрибутива, который можно загрузить с сайта производителя. Для всех версий Windows используется один дистрибутив.

«**cascade-4.0.27-win32-x86.exe**».

Дистрибутив Каскад-САУ для операционной системы Windows содержит полный комплект приложений Каскад-САУ, включая среду исполнения с набором драйверов, среду разработки, сервер проектов и документацию.

Для установки Каскад-САУ запустите программу установки и следуйте ее инструкциям.

По умолчанию программа установки устанавливает полный набор программ Каскад-САУ. При необходимости вы можете выбрать выборочный вариант установки и указать, какие компоненты Каскад-САУ необходимо установить:

- **Среда исполнения** - как правило, устанавливается на программируемые контроллеры, персональные компьютеры и серверы, выполняющие роль контроллеров, АРМ оператора, архивных серверов или серверов сбора данных.
- **Среда разработки** - как правило, устанавливается на персональные компьютеры, на которых предполагается вести разработку и изменение проектов Каскад-САУ.
- **Сервер проектов** - как правило, устанавливается на персональный компьютер или сервер, на котором предполагается хранить проекты и архивы Каскад-САУ. Для высоконагруженных систем рекомендуется использовать для сервера проектов и архивного сервера отдельные выделенные серверы.

Сразу после установки Каскад-САУ готова к использованию, дополнительных действий пользователя по доработке Каскад-САУ и настройке не требуется.

2.3 Установка Каскад-САУ для Linux

Установка Каскад-САУ для операционной системы Linux выполняется с помощью программы установки. Программа установки Каскад-САУ поставляется в виде дистрибутива, который можно загрузить с сайта производителя. Для каждого типа центрального процессора используется соответствующий дистрибутив.

Дистрибутив Каскад-САУ для операционной системы Linux содержит только среду исполнения и набор драйверов.

Для установки Каскад-САУ распакуйте дистрибутив во временную папку запустите файл сценария **install.sh**. Дополнительных действий по установке не требуется.

2.4 Активация лицензии

Без лицензии Каскад-САУ работает ограничениями по количеству используемых точек в проекте (подробнее см. п. 1.5). Чтобы снять эти ограничения необходимо активировать лицензию, полученную при покупке Каскад-САУ.

Для активации лицензии:

- Нажмите кнопку **Пуск**, затем выберите *Программы > Каскад-САУ 4.0 > Служебные программы > Менеджер лицензий*.
- Нажмите кнопку **Добавить лицензию** и выберите файл лицензии.

Примечание. Вы можете активировать одновременно несколько лицензий. Во время работы Каскад-САУ автоматически выберет действующую лицензию с минимальными ограничениями.

2.5 Обновление версии Каскад-САУ

Для обновления Каскад-САУ запустите программу установки более новой версии. Программа установки автоматически удалит старую версию и установит новую.

Примечание. Во время обновления Каскад-САУ все созданные в предыдущей версии проекты и архивы остаются без изменения. Для поддержки новых возможностей Каскад-САУ рекомендуется обновить версию проектов и архивов сразу после обновления (см. п. 0).

2.6 Откат к предыдущей версии

Для отката к предыдущей версии Каскад-САУ удалите установленную версию и запустите программу установки нужной версии.

2.7 Удаление Каскад-САУ

Для удаления Каскад-САУ:

1. Откройте панель управления Windows. В разных версиях Windows панель управления открывается по-разному. Если вы не знаете, как отрыть панель управления в вашей версии Windows, воспользуйтесь встроенной справкой Windows.

2. Выберите *Программы > Программы и компоненты*.

3. Выделите в списке программ *Каскад-САУ 4.0*, затем выберите *Удалить*.

4. Следуйте инструкциям программы на экране.

Примечание. Во время удаления Каскад-САУ проекты и архивы, хранящиеся на компьютере, не удаляются. При необходимости вы можете удалить их вручную. По умолчанию проекты и архивы Каскад-САУ хранятся в следующей папке:

Документы > Общие документы > Cascade 4.0 > projects

3 Основы разработки систем управления в Каскад-САУ

3.1 Порядок разработки систем управления в Каскад-САУ

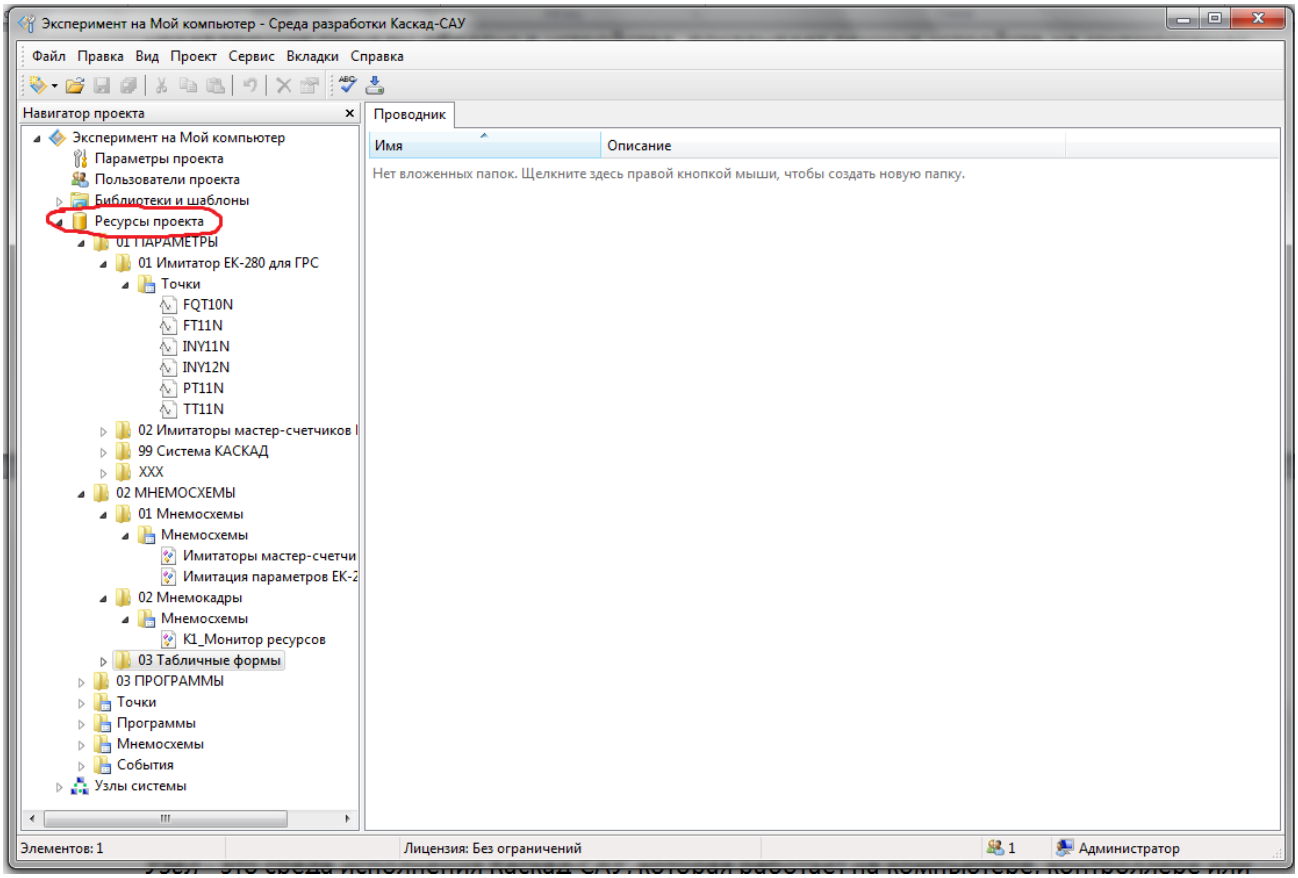
Разработка системы управления в Каскад-САУ начинается с создания проекта системы. Для создания и редактирования проектов используется *Среда разработки Каскад-САУ* (см. п. 4).

С помощью среды разработки в проекте определяются узлы системы (контроллеры, АРМ, серверы), настраивается конфигурация устройств ввода-вывода, подключенных к узлам, программы, архивы, список пользователей и прочие ресурсы (см. п. 3.2), которыми оперирует система управления.

Готовая конфигурация системы управления загружается из проекта на узлы системы и исполняется на них с помощью *Среды исполнения Каскад-САУ* (см. п. 15). Среда исполнения выполняет чтение данных с устройств, исполняет программы обработки данных, записывает управляющие сигналы обратно в устройства, показывает данные устройств на мнемосхемах, записывает в архив и передает в вышестоящие системы сбора данных и управления.

3.2 Ресурсы проекта

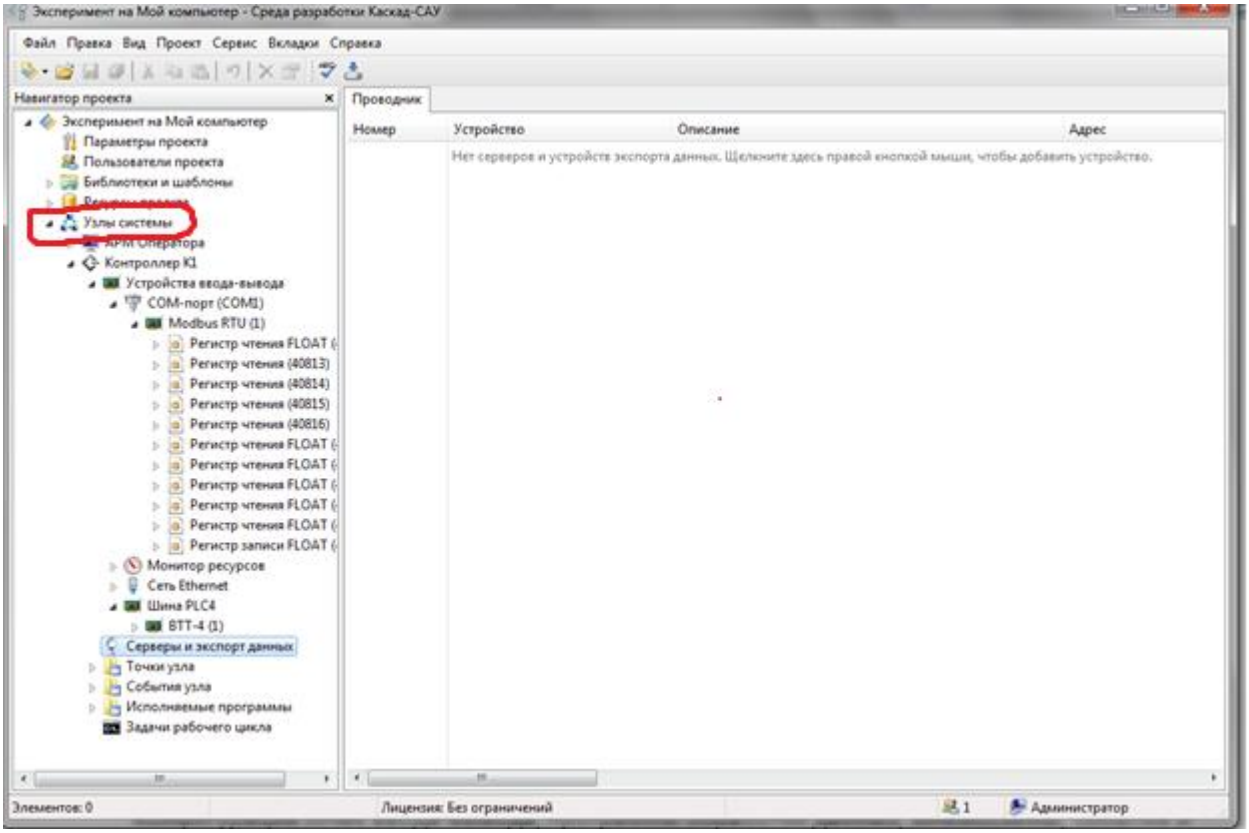
Конфигурация точек ввода-вывода, программы обработки данных, мнемосхемы, события и тревоги называется *ресурсами* проекта.



Ресурсы проекта разделяются между *узлами* системы управления (см. п. 3.3) и используются ими одновременно. Например, значения, записанные в точки ввода на одном узле, могут быть отображены на мнемосхеме на другом узле. Одна программа может исполняться одновременно на нескольких узлах.

3.3 Узлы системы управления

Узел - это среда исполнения Каскад-САУ, которая работает на компьютере, контроллере или сервере.

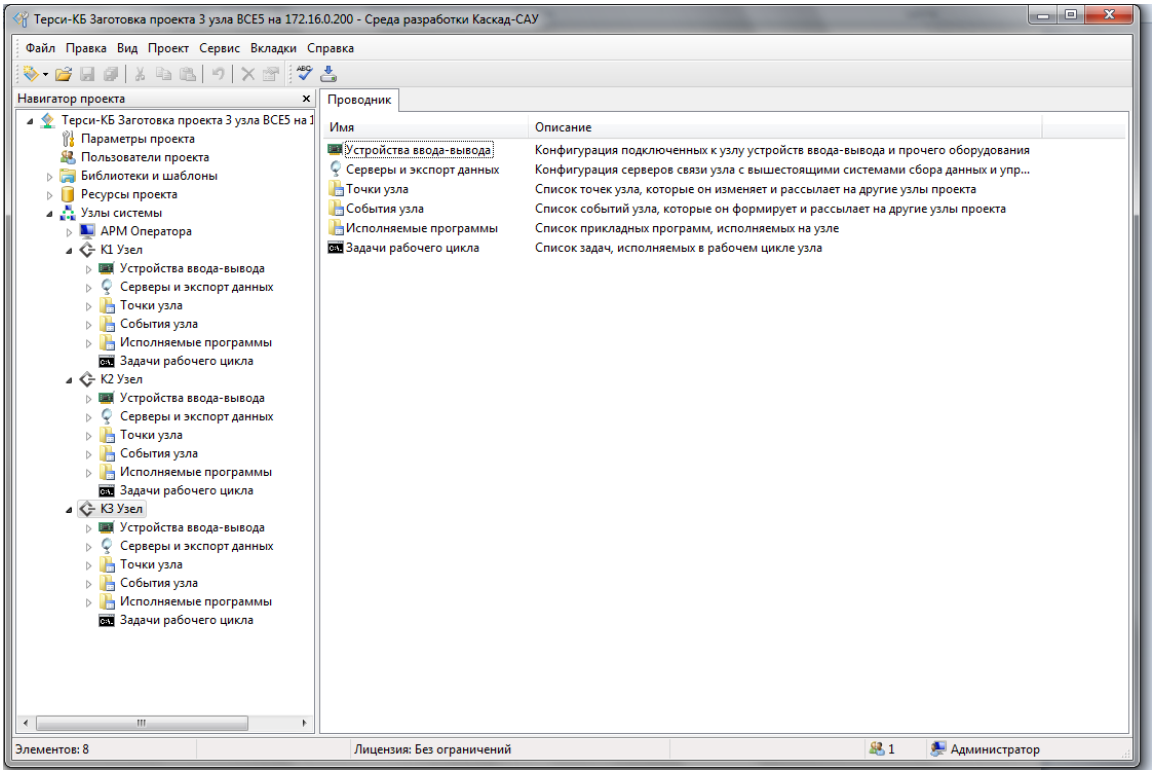


Примечание. В зависимости от контекста под понятием узел может пониматься непосредственно компьютер, контроллер или сервер, на котором работает среда исполнения Каскад-САУ. При этом узел может быть соединен с другими узлами по локальной сети.

Узел загружает конфигурацию из проекта в свою память и сохраняет последнюю удачно загруженную конфигурацию на жестком диске, карте памяти или другом устройстве долговременной памяти для использования при следующем запуске.

Загрузка изменений конфигурации из проекта на узел выполняется по команде из среды разработки. До этого узлы используют для работы последнюю удачно сохраненную конфигурацию.

На одном компьютере может одновременно работать несколько узлов как из одного, так и из разных проектов. При этом один конкретный узел проекта может быть запущен на компьютере лишь в единственном экземпляре.



3.4 Сервер проектов Каскад-САУ

Для управления проектами и архивами Каскад-САУ используется сервер проектов Каскад-САУ.

Сервер проектов обеспечивает *многопользовательский сетевой* доступ к проектам из среды разработки, загрузку конфигурации на узлы системы управления, запись с чтение данных и событий в архивы Каскад-САУ. По команде среды разработки сервер проектов создает новые проекты и архивы, выполняет резервное копирование и восстановление проектов и архивов.

Для хранения баз данных проектов и архивов сервер проектов использует сервер баз данных. Сервер баз данных может быть как внешним (устанавливается отдельно от Каскад-САУ), так и встроенным (встроен в сервер проектов).

Сервер проектов работает как служба Windows.

Внимание! Сервер проектов Каскад-САУ поддерживает только операционную систему Windows.

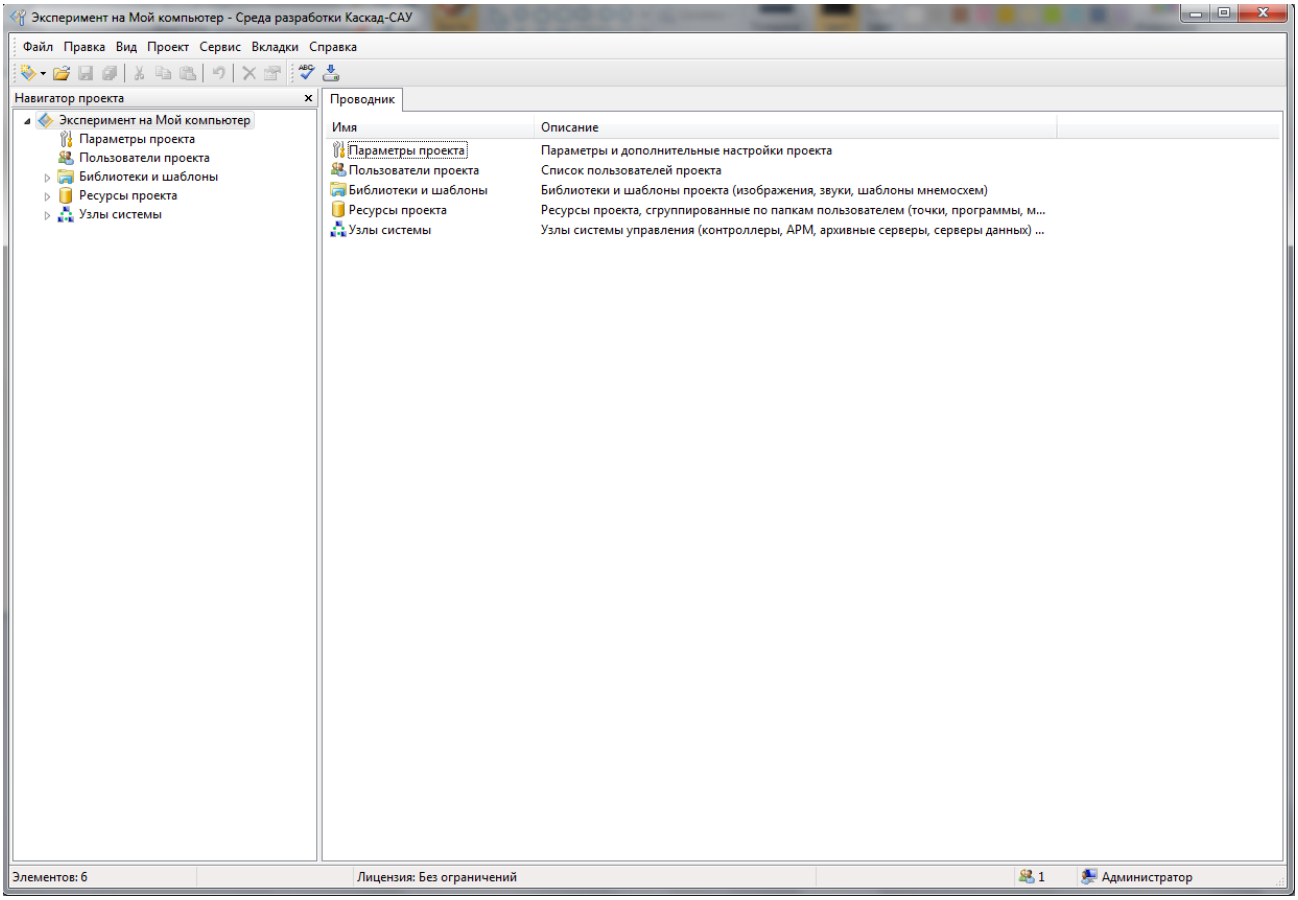
4 Среда разработки

4.1 Запуск среды разработки

Для запуска среды разработки нажмите кнопку Пуск, затем выберите *Программы > Каскад-САУ 4.0 > Среда разработки Каскад-САУ*.

4.2 Главное окно программы

Главное окно среды разработки разделено на две части: панель навигатора проекта слева и панель конфигурации справа.



На панели навигатора отображается содержимое проекта, представленное в виде иерархического дерева (или просто дерево проекта):

- **Параметры проекта** - набор системных параметров проекта.
- **Пользователи проекта** - список пользователей проекта.
- **Библиотеки и шаблоны** - библиотеки изображений и звуков, шаблоны мнемосхем, используемые в проекте.
- **Ресурсы проекта** - база данных общих ресурсов проекта, используемых узлами системы: точки ввода-вывода, программы, мнемосхемы и события.
- **Узлы системы** - список узлов системы и их конфигурация.

На правой панели отображаются вкладка **Проводник** с таблицей параметров, выделенной в дереве проекта, и вкладки открытых редакторов мнемосхем и программ.

Большую часть операций по изменению проекта делается с помощью контекстного меню элементов дерева проекта либо в таблице проводника.

В некоторых случаях для удобства работы можно открыть вторую дополнительную панель навигатора проекта, установив флажок **Навигатор проекта (дополнительная панель)** в меню **Вид**.

4.3 Панели инструментов

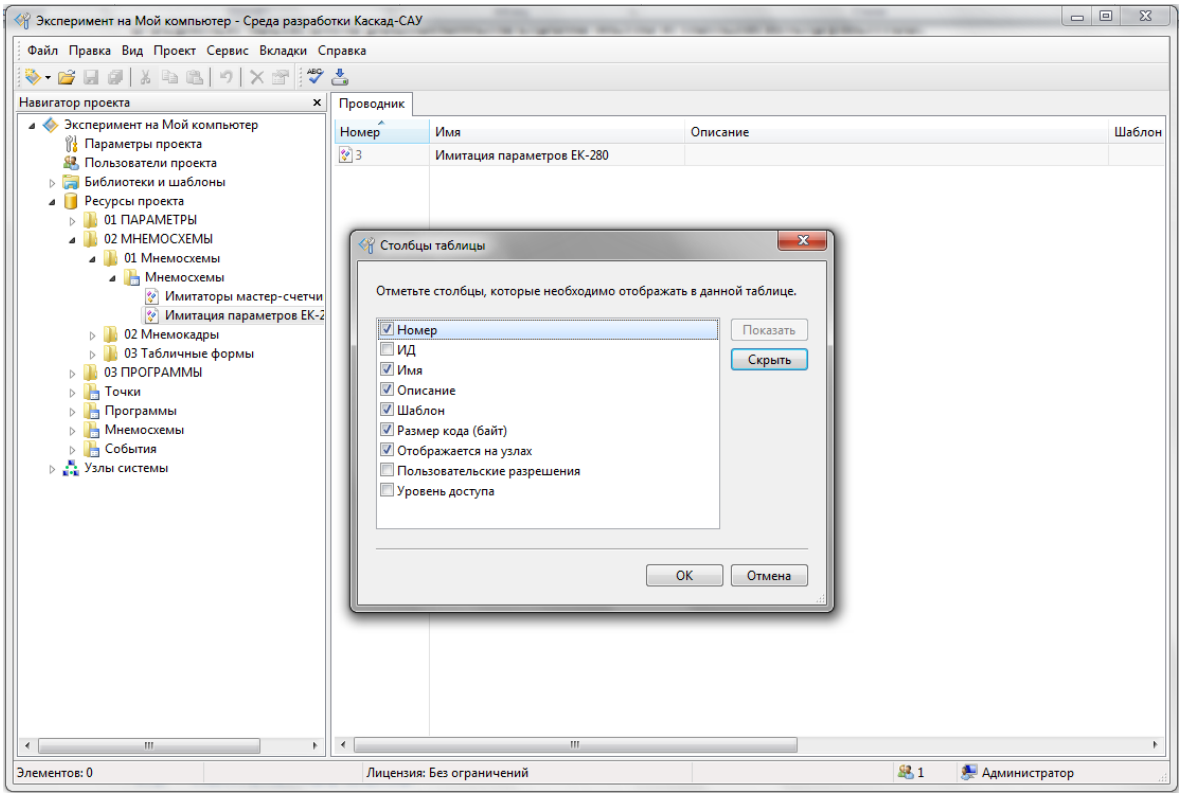
В верхней части окна расположена строка меню и панели инструментов.

Панели инструментов могут автоматически показываться и скрываться в процессе работы. Чтобы показать/скрыть панель вручную щелкните правой кнопкой на панели или строке меню и в открывшемся меню установите/снимите соответствующий флажок, либо установите/снимите флажок панели в меню **Вид**.

4.4 Таблицы редактирования параметров

В среде разработки Каскад-САУ не используются диалоговые окна изменения свойств элементов проекта. Вместо этого свойства изменяются непосредственно в таблицах вкладки Проводник.

Чтобы изменить столбцы, которые отображаются в таблице, щелкните правой кнопкой мышки на заголовке или выберите команду **Выбрать столбцы** в меню **Вид**. Порядок столбцов в таблице изменить нельзя.

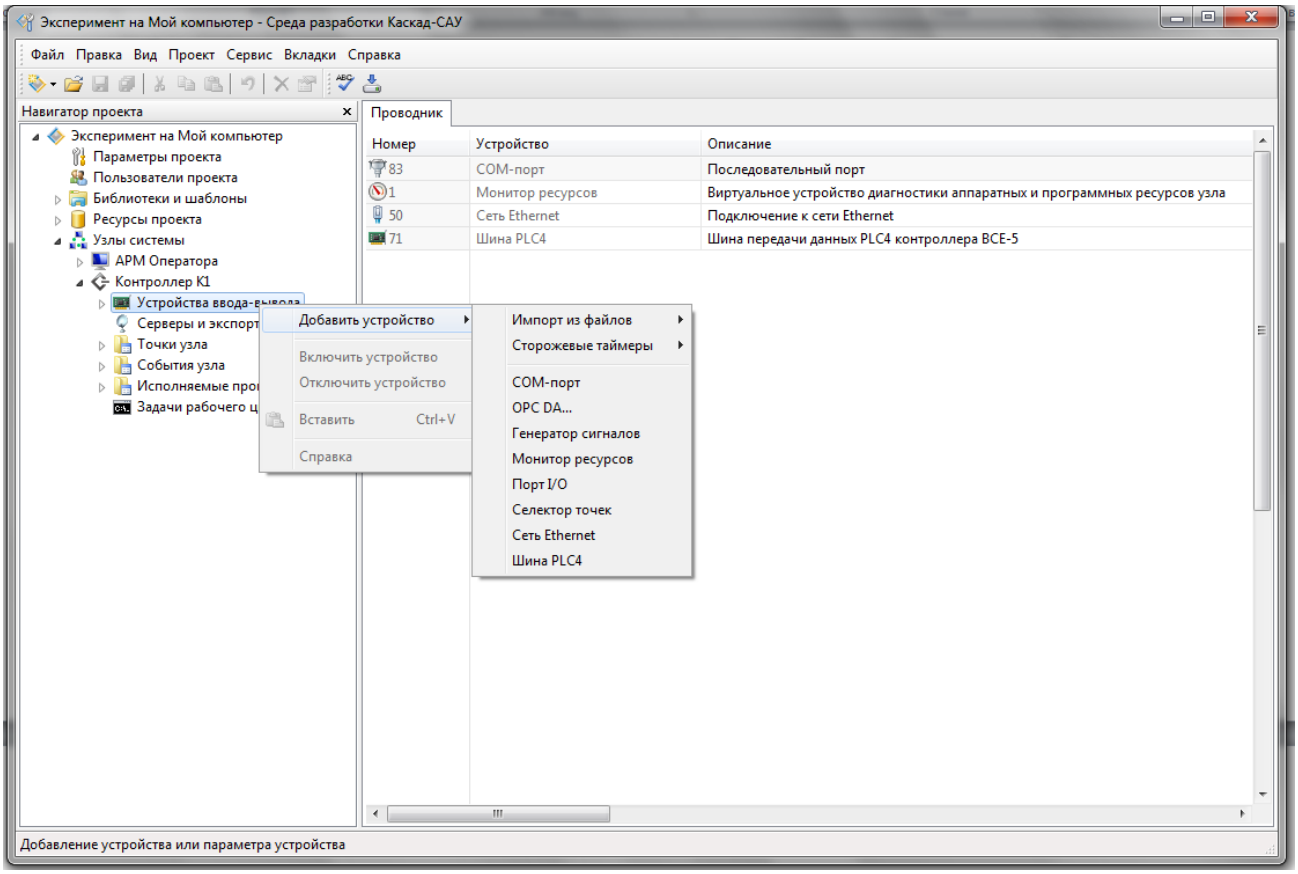


Чтобы быстро переключиться на таблицу вкладки **Проводник** дважды щелкните мышкой на элемент в дереве проекта. Чтобы изменить значение в таблице выделите нужную ячейку и нажмите клавишу **F2** либо начните вводить значение на клавиатуре.

Допускается не вводить значения числовых полей в таблицах. В процессе работы системы такие пустые значения воспринимаются равными 0.

4.5 Контекстное меню

Большинство действий в среде разработки выполняется с помощью контекстного меню, вызываемого по щелчку правой кнопкой мыши на элементе дерева проектов или строке таблицы на вкладке **Проводник**.



Набор команд контекстного меню зависит от элемента, для которого оно вызывается и может включать в себя:

- специализированные команды: создать точку, создать программу, добавить устройство, компилировать программу, запустить узел и т.д.,
- команды по работе с буфером обмена: вырезать, копировать, вставить,
- команды по работе с элементом: удалить, переименовать и дублировать.

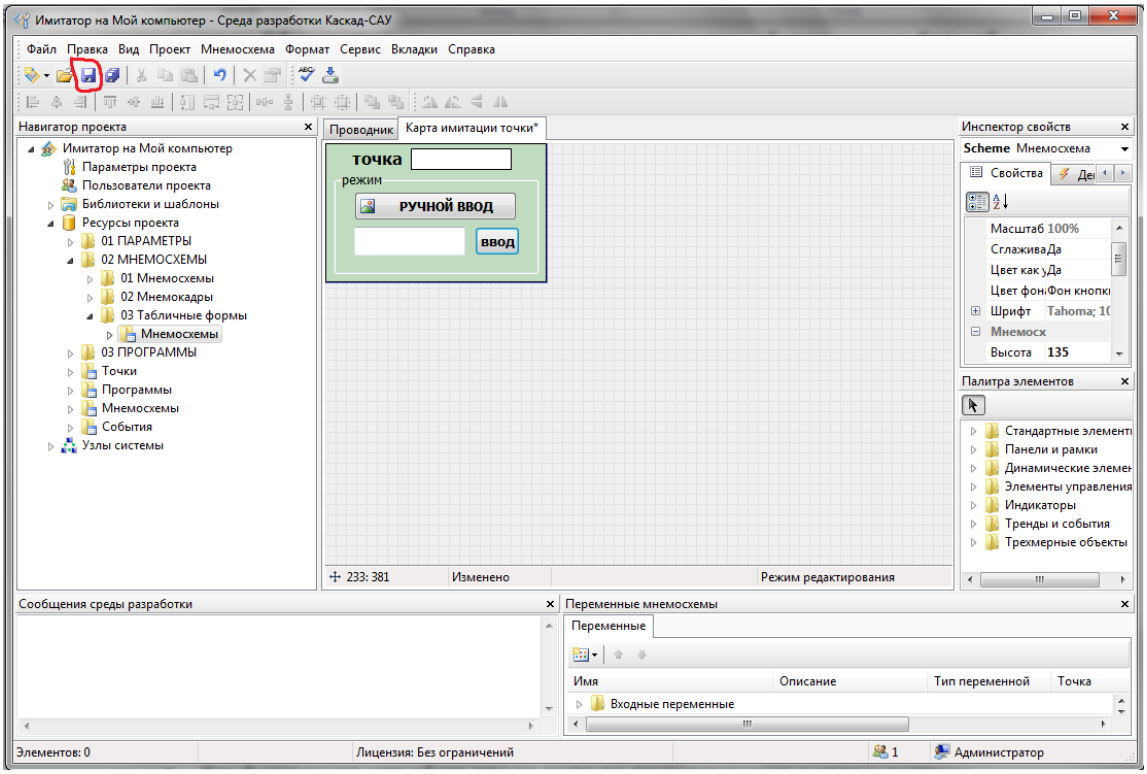
Специализированные команды всегда располагаются в верхней части меню перед общими командами. Общие команды всегда располагаются в нижней части меню. Если общая команда не поддерживается элементом, то она не отображается.

Примечание. Специализированные команды контекстного меню дублируются в главном меню среды разработки в подменю ***Проект****.*

4.6 Сохранение изменений проекта

Все изменения, сделанные в дереве проекта и таблицах, сразу же записывается в проект. Дополнительных действий по сохранению этих изменений не требуется.

Для сохранения изменений на вкладках редакторов мнемосхем и программ следует нажать кнопку **Сохранить** на панели инструментов.



5 Проект системы управления

5.1 Проект Каскад-САУ

Проект Каскад-САУ - это база данных, в которой хранится конфигурация системы управления. Эта конфигурация хранится в базе данных в виде набора таблиц и включает в себя:

- Конфигурацию узлов системы управления;
- Конфигурацию устройств ввода-вывода, подключенных к узлам системы.
- Конфигурацию точек ввода-вывода системы.

- Программы обработки данных, исполняемые на узлах.
- Мнемосхемы, используемые на узлах для отображения данных и приема команд управления от операторов.
- Библиотеку звуков и изображений, используемых в мнемосхемах.
- Конфигурацию событий и тревог.
- Конфигурацию архивов для хранения истории изменения данных и событий.
- Конфигурацию пользователей системы управления.

5.2 Создание нового проекта

Для создания нового проекта в среде разработки:

1. Выберите команду **Новый проект** в меню **Файл**. Откроется окно мастера создания нового проекта.

2. Введите в поле **Проект** имя нового проекта. Это имя будет отображаться в среде разработки при открытии проекта. Имя проекта не может состоять только из пробелов, а также содержать символы \/[]:|<>+=;,?*"

3. Введите в поле **База данных** полный путь и имя файла базы данных проекта. Имя файла может отличаться от имени проекта. Нажмите кнопку **Обзор**, чтобы выбрать другое имя файла либо папку проекта.

Примечание. Если не указать путь к файлу, то проект будет создан в папке проектов по умолчанию. По умолчанию проекты и архивы Каскад-САУ хранятся в следующей папке:

Документы > Общие документы > Cascade 4.0 > projects

4. Выберите в списке **Драйвер** тип драйвера базы данных проекта:

- **Firebird** - выберите, если на вашем компьютере установлен сервер баз данных Firebird. Этот драйвер рекомендуется для больших проектов и проектов, которые предполагается использовать для совместного редактирования несколькими пользователями.
- **InterBase** - выберите, если на вашем компьютере установлен сервер баз данных InterBase.
- **Firebird Embeded** - выберите, если на вашем компьютере не установлено никаких серверов баз данных. В этом случае будет использован встроенный сервер баз данных.

5. Установите флажок **Включить защиту паролем** и введите пароль администратора проекта, чтобы защитить доступ к проекту паролем.

6. Нажмите кнопку **Создать** для создания проекта. Дождитесь окончания создания.

7. Установите флажок **Открыть проект** и нажмите кнопку **Готово**.

8. В случае ошибки создания проекта установите флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе создания проекта, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок создания.

5.3 Уникальный GUID проекта

При создании каждому проекту присваивается уникальный глобальный идентификатор - *GUID*. Уникальный идентификатор проекта хранится непосредственно в проекте и не может быть изменен.

Для просмотра уникального идентификатора проекта щелкните правой кнопкой мышки на названии проекта и выберите команду **Свойства**. Идентификатор проекта отображается в поле **GUID** на вкладке **Общие**.

GUID проекта используется для его однозначной идентификации на сервере проектов. Команды автоматического запуска Каскад-САУ при старте Windows используют *GUID* для выбора проекта на сервере. Переименование проекта не приведет к сбою автоматического запуска Каскад-САУ.

GUID проекта используется для контроля подмены базы данных проекта. Если подменить файл базы данных проекта файлом другого проекта, сервер проектов по изменению *GUID* обнаружит подмену и запретит работу с подложным проектом.

GUID проекта используется для контроля резервной копии при восстановлении проекта. При выборе для восстановления файла резервной копии от другого проекта будет выдано соответствующее предупреждение (см. п. 5.11).

GUID проекта используется в протоколе обмена данными в распределенных системах. Если в результате ошибки конфигурации соединить узлы разных проектов, то данные, события и команды управления узла одного проекта не будут переданы на узел другого проекта (см. п. 16.2).

5.4 Общий доступ к проекту

По умолчанию проект можно открыть только на том компьютере, на котором он был создан.

Чтобы разрешить доступ к проекту с других компьютеров в сети необходимо в окне свойств проекта на вкладке **Доступ** установить флажок **Разрешить общий доступ к проекту**.

Включить общий доступ можно только на том компьютере, на котором хранится файл проекта. Удаленно выключить общий доступ для проекта на другом компьютере (см. п. 5.5) нельзя.

Примечание. В дереве проектов среды разработки иконка проекта, к которому разрешен доступ с других компьютеров, отображается со значком общего доступа.

Обычно проект с общим доступом хранят на выделенном сервере. Для небольших систем вместо сервера можно использовать рабочий компьютер одного из пользователей в сети.

5.5 Открытие проекта на другом компьютере

Чтобы открыть проект, расположенный на другом компьютере, выберите в меню *Файл* команду *Открыть > Проект*. В открывшемся окне **Выбор проекта** раскройте в папку *Сеть* и выберите требуемый сервер проектов, затем проект на этом компьютере.

Примечание. В списке проектов сервера показываются только те проекты, для которых разрешен общий доступ (см. п. 5.3).

Среда разработки автоматически ищет компьютеры в локальной сети, на которых установлены серверы проектов Каскад-САУ, и показывает их в папке *Сеть*. Для поиска серверов среда разработки использует рассылку широковещательных UDP пакетов.

Если нужный сервер не показывается (например, если межсетевой экран блокирует рассылку широковещательных UDP пакетов), то сервер проектов можно добавить вручную. Для этого в окне **Выбор проекта** нажмите кнопку **Добавить сервер**, в открывшемся окне **Выбор сервера** выберите сервер из списка или введите его IP-адрес в поле вводе **Сервер** и нажмите кнопку **ОК**.

Примечание. Серверы, добавленные вручную, отображаются в списке серверов со значком ярлыка.

5.6 Совместное редактирование проекта

Каскад-САУ позволяет редактировать один проект одновременно несколькими пользователями по локальной сети. Пользователь может открыть проект одновременно с другими пользователями и совместно работать над этим проектом. Это называется совместным редактированием проекта.

При совместном редактировании в окне среды разработки показываются только те изменения, который сделал текущий пользователь. Изменения, сделанные другими пользователями необходимо загружать вручную командой **Обновить** в меню **Вид** или нажатием клавиши **F5**. О том, что есть такие изменения, показывает надпись Требуется обновление в строке состояния.

Количество пользователей, которые сейчас редактируют текущий проект, отображается в строке состояния окна среды разработки. Чтобы посмотреть имена этих пользователей наведите курсор мыши на цифру с количеством пользователей в строке состояния.

Если два пользователя одновременно изменяют одну и ту же точку, программу или мнемосхему, в проекте будут сохранены изменения того, пользователя, кто сделал изменения последним. Разделение доступа к ресурсам проекта выполняется административными методами, по договоренности между пользователями.

5.7 Скрытый проект

Каждый созданный проект может быть выбран для запуска среды исполнения из диалога запуска Каскад-САУ (см. п.).

Чтобы не показывать проект в диалоге запуска Каскад-САУ следует установить проекту атрибут *Скрытый*. Для этого в окне свойств проекта необходимо на вкладке **Общие** установить флажок **Скрытый**.

*Примечание. Установка атрибута **Скрытый** не скрывает проект от среды разработки. В дереве проектов среды разработки скрытые проекты отображаются бледным значком.*

5.8 Проект только для чтения

Установка атрибута **Только чтение** помогает защитить проект от случайных изменений. Проект, доступный только для чтения, можно открыть в среде разработки, как и любой другой проект, но добавление, удаление или изменение данных в нем будут невозможны.

Чтобы установить атрибут **Только чтение** необходимо в окне свойств проекта на вкладке **Общие** установить соответствующий флажок.

Примечание. В дереве проектов среды разработки проекты, доступные только для чтения, отображаются со значком замка.

5.9 Резервное копирование проекта

Для создания резервной копии проекта:

1. Щелкните правой кнопкой мышки в дереве проектов на названии проекта и выберите команду **Создать резервную копию**. Откроется окно мастера резервного копирования проекта.

2. Введите в поле **Файл резервной копии** имя файла с резервной копией проекта.

3. При необходимости введите в поле **Комментарий** текст комментария к этой резервной копии.

4. Нажмите кнопку **Копировать**. Дождитесь окончания резервного копирования.

5. Установите флажок **Открыть папку с резервной копией**, чтобы открыть папку с созданной резервной копией в проводнике Windows, и нажмите кнопку **Готово**.

6. В случае ошибки резервного копирования установите флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе создания резервной копии, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок.

5.10 Автоматическое резервное копирование

Среда разработки Каскад-САУ поддерживает режим автоматического резервного копирования проектов с локального или удаленного сервера проектов. В этом режиме среда разработки при запуске выполняет резервное копирование указанного проекта и завершает работу.

Для запуска автоматического резервного копирования проекта запустите среду разработки с параметром командной строки **--backup-project**, например:

```
prjedit.exe --backup-project --server-name=127.0.0.1 --project-guid=GIUD
```

Замените в примере выше *127.0.0.1* на IP-адрес сервера проектов, на котором хранится проект, замените *GUID* на строку GUID проекта.

*Примечание. Для получения GUID проекта щелкните правой кнопкой мышки в дереве проекта на названии проекта и выберите команду **Свойства**. Щелкните в окне свойств правой кнопкой мышки на имени базы данных проекта, чтобы скопировать GUID в буфер обмена.*

Для получения подсказки по параметрам командной строки среды разработки запустите ее с параметром **-?** или **--help**.

Для запуска резервного копирования проекта по расписанию рекомендуется использовать *Планировщик заданий* Windows. Для этого:

1. Откройте окно **Управление компьютером**.

2. Щелкните правой кнопкой на папке *Планировщик заданий* и выберите команду **Создать задачу**.

3. На вкладке **Общие** введите в поле **Имя** название задачи (например, *Резервное копирование проекта*).

4. На вкладке **Триггеры** нажмите кнопку **Создать** и укажите расписание запуска резервного копирования.

5. На вкладке **Действия** нажмите кнопку **Создать**, выберите с помощью кнопки **Обзор** имя файла среды разработки *prjedit.exe*, укажите в поле **Добавить аргументы** параметры запуска (см. выше).

6. Нажмите кнопку **ОК**.

Для контроля результата автоматического резервного копирования среда разработки записывает событие о выполненном копировании в журнал событий Windows. Имя источника события *Cascade AutoBackup 4.0*.

5.11 Восстановление проекта из резервной копии

Для восстановления проекта из резервной копии:

1. Убедитесь в том, что проект не используется другими пользователями (см. п. 5.6).

2. Щелкните правой кнопкой мышки в дереве проектов на названии проекта и выберите команду **Восстановить из резервной копии**. Откроется окно мастера восстановления проекта.

3. Нажмите кнопку **Обзор** и выберите файл с резервной копией проекта.

4. Введите в поле **Комментарий** причину восстановления проекта.

5. Нажмите кнопку **Восстановить**. Дождитесь окончания восстановления проекта.

Примечание. Разрешается использовать для восстановления проекта резервную копию другого проекта. При использовании резервной копии другого проекта перед началом восстановления будет выдано соответствующее предупреждение.

6. В случае ошибки восстановления установите флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе восстановления, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок.

Внимание! Восстановление проекта полностью удаляет существующий проект.

5.12 Создание нового проекта из резервной копии

Проект из резервной копии можно восстановить в новый проект. Например, для создания копии существующего проекта на сервере или для переноса проекта на другой сервер.

Внимание! Создать новый проект из резервной копии можно только на локальном компьютере. Удаленное создание проекта из резервной копии на другом сервере не поддерживается.

Чтобы восстановить проект из резервной копии в новый проект:

1. Выберите в меню Файл команду **Создать**, затем **Новый проект из резервной копии**.

2. Выдерите файл с резервной копией проекта. Откроется окно мастера создания нового проекта.

3. Введите в поле **Проект** имя нового проекта.

Примечание. По умолчанию мастер предлагает использовать имя проекта из резервной копии.

4. введите в поле **База данных** полный путь и имя файла базы данных проекта. Имя файла может отличаться от имени проекта. Нажмите кнопку **Обзор**, чтобы выбрать другое имя файла либо папку проекта.

Примечание. По умолчанию мастер предлагает создать базу данных в той же папке, из которой была сделана резервная копия.

5. Выберите в списке **Драйвер** тип драйвера базы данных проекта (см. п. 5.2)

6. Нажмите кнопку **Создать** для создания проекта. Дождитесь окончания создания.

7. Установите флажок **Открыть проект** и нажмите кнопку **Готово**.

8. В случае ошибки создания проекта установите флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе создания проекта, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок.

5.13 Создание копии проекта

Для создания копии проекта на сервере проектов:

1. Создайте резервную копию проекта (см. п. 5.9).

2. Создайте новый проект из созданной резервной копии (см. п. 5.12).

5.14 Перенос проекта на другой сервер

Для переноса проекта на другой компьютер:

1. Создайте резервную копию проекта (см. п. 5.9).

2. Запустите среду разработки на компьютере, на котором установлен и работает другой сервер проектов.

3. Создайте новый проект из созданной резервной копии (см. п. 5.12).

Перенос файла базы данных проекта вручную путем копирования его с одного сервера на другой не рекомендуется. Если во время копирования проект будет открыт в среде разработки, то копия файла будет испорчена. В крайнем случае, допускается копирование

файла, предварительно убедившись, что файл не открыт в других программах, например, переименовав его перед копированием.

Чтобы открыть на локальном компьютере проект, скопированный вручную, выберите в меню **Файл** команду **Открыть**, **Файл проекта**.

5.15 Обновление версии проекта

В некоторых случаях для поддержки возможностей, добавленных в новых версиях Каскад-САУ, требуется обновление версии проекта. Например, обновление версии проекта требуется для добавления в него шаблонов новых устройств (см. п. 8.2).

Внимание! Обновление версии проекта нельзя отменить. Перед обновлением версии сделайте резервную копию проекта (см. п. 5.9).

Для обновления версии проекта:

1. Щелкните правой кнопкой мышки в дереве проектов на названии проекта и выберите команду **Обновить версию**. Откроется окно мастера обновления версии проекта.

2. Нажмите кнопку Обновить. Дождитесь окончания создания.

3. В случае ошибки обновления версии установите флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе обновления проекта, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок.

Примечание. Обновление проекта выполняет сервер проектов. Максимальная версия, до которой можно обновить проект, определяется версией сервера проектов, н не версией среды разработки Каскад-САУ.

5.16 Журнал изменения проекта

Все изменения, сделанные в проекте в среде разработки, автоматически записываются в журнал изменений проекта. Журнал изменений проекта хранится в самом проекте и не может быть удален или изменен.

Журнал изменения проекта хранит записи о следующих событиях:

- создание проекта,
- резервное копирование и восстановление проекта,
- обновление версии проекта,
- добавление, изменение и удаление ресурсов проекта,
- применение изменений конфигурации для загрузки на узлы.

Для просмотра журнала изменения *откройте проект*, щелкните в дереве проектов правой кнопкой мышки на названии проекта и выберите команду **Журнал изменения**. Журнал будет открыт на отдельной вкладке.

Для каждого события в журнале указываются:

- дата и время события,
- имя пользователя, который сделал изменение,
- текст сообщения или описание изменения,
- дополнительные данные события.

Дополнительные данные содержат служебную информацию, например, значение измененного поля, имя файла резервной копии и прочее. Для просмотра дополнительных данных события щелкните правой кнопкой мышки на заголовке таблицы событий и установите флажок **Данные события**.

6 Узлы системы

6.1 Создание нового узла

Для создания нового узла щелкните в дереве проекта правой кнопкой на папке **Узлы системы**, выберите команду **Создать узел** и затем выберите тип нового узла:

- **Контроллер** - предназначен для ввода-вывода данных с подключенных устройств и обработку данных с помощью пользовательских программ.
- **АРМ оператора** - предназначен для отображения данных других узлов системы на мнемосхемах и приема команд от оператора.
- **Архивный сервер** - предназначен для архивирования данных и событий других узлов.
- **Сервер данных** - предназначен для передачи данных в другие системы управления.

6.2 Конфигурация узла

Деление узлов по типам является условным. Назначение узла определяется его конфигурацией.

Например, на узле типа **Контроллер** по умолчанию настраивается конфигурация устройств ввода-вывода и программы обработки данных, на узле типа **АРМ оператора** - мнемосхемы, на узле **Архивный сервер** - список архивов. Если настроить на узле типа **АРМ оператора** конфигурацию подключенных устройств, программы обработки данных и архивы, то он одновременно будет выполнять функции и узла типа **АРМ оператора**, и **Контроллера** и **Архивного сервера**.

Чтобы включить возможность настройки той или иной конфигурации узла щелкните правой кнопкой на узле, затем в меню **Видимые ресурсы** узла установите требуемые флажки:

- **Устройства ввода-вывода** - настройки конфигурации устройств ввода-вывода узла.
- **Серверы и экспорт данных** - настройки конфигурации протоколов обмена данными с другими системами.
- **Точки узла** - настройка точек, принадлежащих узлу.
- **События и тревоги** - настройка правил выдачи событий узла.
- **Исполняемые программы** - настройка программ, исполняемых на узле.
- **Отображаемые мнемосхемы** - настройка мнемосхем, отображаемых на узле.
- **Архивы узла** - настройка списка архивов, которые хранятся и работают на узле.
- **Задачи рабочего цикла** - настройка задач рабочего цикла узла.

6.3 Свойства узла

Для настройки свойств узла дважды щелкните в дереве проекта на папке **Узлы системы** и перейдите в таблицу на вкладке **Проводник**. Выделите строку с узлом и введите значения свойств.

Номер - порядковый номер узла в таблице узлов. Значение свойства выбирается автоматически при создании узла и не может быть изменено. При удалении узла его номер может быть назначен следующему новому узлу.

ИД - уникальный номер узла в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя узла, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя узла должно быть уникальным в проекте. Регистр символов имеет значение.

Описание - описание узла, строка не более 127 символов.

Тип узла - тип, выбранный при создании узла. Не может быть изменен.

IP-адрес К1 - IPv4 адрес узла в сети.

IP-адрес К1 (резерв. линия) - не используется, зарезервировано для будущих версий.

IP-адрес К2 - не используется, зарезервировано для будущих версий.

IP-адрес К2 (резерв. линия) - не используется, зарезервировано для будущих версий.

Рабочий цикла - длительность рабочего цикла узла в миллисекундах. Значение не может быть менее 1 мс. Подробнее о рабочем цикле узла см. п. 15.2.

Параметры - строка дополнительных параметров работы узла (см. ниже).

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства узла. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства узла. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

6.4 Параметры работы узла

Свойство **Параметры узла** содержит строку дополнительных параметров работы узла. Значения параметров задаются в формате *Параметр=Значение*, параметры разделяются точкой с запятой.

DataPort - номер порта сервера данных узла. Значение должно быть уникальным для всех узлов, запускаемых на одном компьютере. В противном случае данные будут приниматься только от одного из запущенных узлов, прием данных от других узлов работать не будет. Подробнее об обмене данными между узлами одного проекта см. п. 16.

EventPort - номер порта сервера событий узла. Значение должно быть уникальным для всех узлов, запускаемых на одном компьютере. В противном случае события будут приниматься только от одного из запущенных узлов, прием событий от других узлов работать не будет.

DirectPort - номер порта сервера управления узла. Значение должно быть уникальным для всех узлов, запускаемых на одном компьютере. В противном случае получать команды управления будет только один из запущенных узлов, другие узлы получать команды управления не будут.

ConnectTimeout - таймаут подключения к узлу, миллисекунды. Если по истечении указанного времени соединении с узлом не установлено, то выдается ошибка и подключение прерывается, повторная попытка подключения будет выполнена через 10 секунд. Значение по умолчанию 10000 мс. Введите -1, чтобы задать неограниченное время ожидания соединения, введите 0, чтобы использовать значение по умолчанию.

WriteTimeout - таймаут записи данных в миллисекундах. Если по истечении указанного времени запись не будет выполнена, то выдается ошибка, соединение с узлом разрывается. Значение по умолчанию 1000 мс. Введите -1, чтобы задать неограниченное время ожидания записи, введите 0, чтобы использовать значение по умолчанию.

KeepaliveTimeout - Период отправки пакетов проверки связи в миллисекундах. Если в течении указанного времени от узла не будет получено пакета проверки связи или пакета

данных, то считается, что связь с ним потеряна, соединение разрывается. Введите -1, чтобы задать неограниченное время ожидания записи, введите 0, чтобы использовать значение по умолчанию.

NvramUpdate - Интервал обновления устройства энергонезависимой памяти в циклах узла. Значение по умолчанию 10 (5 секунд при рабочем цикле 500 мс). Для продления срока службы FLASH-дисков и карт памяти не рекомендуется ставить значение менее 5 циклов. Подробнее о работе узла с энергонезависимой памятью см.п. XXX.

7 Точки

7.1 Точки Каскад-САУ

Точка – это переменная узла Каскад-САУ. Точки используются для хранения и обмена данными, которыми оперирует среда исполнения Каскад-САУ:

- в точках хранятся значения, считанные с устройств, и их статус (подробнее см. п. 7.7);
- каждое значение хранится в паре со статусом, который определяет качество и режимы обработки значения (подробнее см. п. 7.7).
- значения точек используются в пользовательских программах для расчетов;
- значения точек записываются в устройства и передаются в другие системы;
- значения точек отображаются на мнемосхемах и трендах;
- значения точек меняются по команде оператора с мнемосхем;
- значения точек используются для формирования событий и тревог;
- значения точек сохраняются в архив и в энергонезависимой памяти контролеров.

Аналогом точки Каскад-САУ в других SCADA системах является "тег" (англ. tag).

7.2 Максимальное количество точек в проекте

Максимальное количество точек, которое может быть в проекте, определяется лицензией на среду разработки Каскад-САУ.

По умолчанию проект создается с лимитом в 100 точек, даже если лицензия разрешает использовать большее количество. Если количества точек недостаточно для работы, следует увеличить лимит точек проекта.

В Каскад-САУ отсутствуют скрытые, промежуточные или автоматически создаваемые точки, которые учитываются лицензией. Количество точек проекта - это количество тех точек, которые созданы пользователем вручную. Текущее и максимальное количество точек в проекте можно посмотреть в диалоге **Лимит ресурсов** проекта.

Подробнее об изменении лимитов проекта см.п.

7.3 Создание новой точки

Для создания новой точки узла раскройте в дереве проекта папку узла, щелкните правой кнопкой на папке **Точки узла**, выберите команду **Создать точку** и затем выберите тип ввода-вывода новой точки:

- **Входная** – точка для чтения данных с устройства ввода.
- **Выходная** – точка для записи данных в устройство вывода.
- **Диагностическая** – точка для чтения признака наличия связи с устройством.
- **Виртуальная** – точка для хранения результатов промежуточных вычислений.

Все созданные таким образом точки будут иметь тип данных **LREAL**. Чтобы создать точку определенного типа данных раскройте папку **Точки узла**, щелкните правой кнопкой на папке типа ввода-вывода, выберите команду **Создать точку** и затем выберите тип данных новой точки:

- **BOOL** - булевое (True/False).
- **INT** - целое число со знаком, 16-бит.
- **DINT** - целое число со знаком, 32-бита.
- **REAL** - число с плавающей точкой, 32-бита.
- **LREAL** - число с плавающей точкой, 64-бита.
- **TIME** - интервал времени.
- **DATE** - дата и время.

Подробнее о типах данных точки см. п. 7.5.

Примечание. Входные, выходные и диагностические точки рекомендуется создавать непосредственно из контекстного меню устройства точки. При таком подходе тип данных точки будет выбран автоматически согласно типу данных значения устройства.

7.4 Свойства точки

Для настройки свойств точки дважды щелкните левой кнопкой значок **Точки узла** и перейдите в таблицу на вкладке **Проводник**. Выделите строку с точкой и введите значения свойств.

Номер - порядковый номер точки в таблице точек. Значение свойства выбирается автоматически при создании точки и не может быть изменено. При удалении точки ее номер может быть назначен следующей новой точке.

ИД - уникальный номер точки в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя точки, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя не может содержать одновременно русские и латинские буквы. Имя точки должно быть уникальным в проекте. Регистр символов имеет значение.

Описание - описание точки, не более 127 символов.

Тип ввода-вывода - определяет, как формируется и используется значение точки:

- **Входная** - точка, значение которой считывается с устройства ввода.
- **Выходная** - точка, значение которой записывается в устройство вывода.
- **Диагностическая** - точка, значение которой точки хранит признак наличия связи с устройством.
- **Виртуальная** - точка, значение которой используется для хранения промежуточных вычислений.

Тип данных - определяет допустимый диапазон значений точки:

- **BOOL** - булевое (True/False).
- **INT** - целое число со знаком, 16-бит.
- **DINT** - целое число со знаком, 32-бита.
- **REAL** - число с плавающей точкой, 32-бита.
- **LREAL** - число с плавающей точкой, 64-бита.
- **TIME** - интервал времени.
- **DATE** - дата и время.

Подробнее о типах данных точек см. п. 7.5.

Сохранение в энергонезависимой памяти - включение и выключение сохранения значения и статуса точки в энергонезависимой памяти узла. Значение и статус точек, у которых включен признак, восстанавливаются из энергонезависимой памяти при запуске узла (включении контроллера). Подробнее об использовании энергонезависимой памяти см. п. 7.7 и 15.15.

Сохранение в архиве - включение и выключение сохранение в архивной базе данных значения точки. Для поддержки архива в проекте должен быть создан и запущен узел типа **Архив** с назначенным текущим архивом. Подробнее о сохранении значения точки в архив см. п. 12 и п. 15.14.

Единицы измерения - строка с обозначением единиц измерения точки, не более 15 символов. Используется для отображения на мнемосхемах и трендах. Свойств носит информационный характер и не может использоваться для выбора преобразования значения точки из одних единиц в другие.

Формат отображения - строка с форматом отображения значения точки на мнемосхемах, в трендах и списке событий, не более 31 символа. Формат отображения различный для разных типов данных точки. Подробнее о формате отображения значения точки см. п 7.6.

Примечание – строка примечания точки, не более 247 символов. Используется для ввода произвольной дополнительной информации по точке. Не используется во время исполнения проекта.

Узел-владелец - узел проекта, который записывает значение в точку. Только один узел проекта может записывать значение в точку, остальные узлы будут получать значение от узла-владельца. Если точка не принадлежит ни одному из узлов проекта, то на всех узлах будут разные значения этой точки. Подробнее о поддержке распределенных систем см. п. 16.

Устройство ввода - устройство, значение которого записывается в точку. К устройству ввода или регистру чтения сервера данных может быть привязана только одна точка (см. свойство Сервер данных (прием) ниже). Подробнее о вводе данных с устройств в точки см. п. 15.6.

Устройство вывода - устройство, в которое записывается значение точки. Значение одной точки может записываться сразу в несколько устройств вывода, в этом случае эти устройства указываются через точку с запятой. Подробнее о выводе данных из точек в устройства см. п. 15.8.

Сервер данных (передача) - регистр чтения (или параметр, канал и т.п.) сервера данных, который используется для передачи значение точки во внешние системы. Значение одной точки может передаваться во внешние системы сразу через несколько серверов данных, в этом случае их регистры указываются через точку с запятой.

Сервер данных (прием) - регистр записи (параметр, канал и т.п.) сервера данных, команда записи в который из внешней системы будет записана в значение точки. К регистру записи или устройству ввода может быть привязана только одна точка (см. свойство Устройство ввода выше).

ТЕ мин. - минимальное значение диапазона технических единиц точки. Используется для преобразования значения устройства в значение точки (для входных точек) и обратно (для выходных точек). Подробнее о предобразовании см. п. 15.6 и п. 15.8.

ТЕ макс. - максимальное значение диапазона технических единиц точки. Используется для преобразования значения устройства в значение точки (для входных точек) и обратно (для выходных точек). Подробнее о предобразовании см. п. 15.6 и п. 15.8.

АЦП-ТЕ преобразование - вид преобразования значения устройства в значение точки (АЦП-ТЕ для входных точек) или значения точки в значение устройства (ТЕ -ЦАП для выходных точек). Вид преобразования зависит от типа ввода-вывода и типа данных точки:

- **Нет** - преобразование не выполняется, значение устройства записывается в точку "как есть", значение точки записывается в устройство "как есть".
- **y = ТЕ мин. + (x - АЦП мин.)·(ТЕ/АЦП)** - преобразование значения из диапазона АЦП устройства в диапазон ТЕ технических единиц точки. Только для точек типа **INT**, **DINT**, **REAL**, **LREAL**.
- **y = ЦАП мин. + (x - ТЕ мин.)·(ЦАП/ТЕ)** - преобразование значения из диапазона ТЕ технических единиц точки в диапазон АЦП устройства. Только для точек типа **INT**, **DINT**, **REAL**, **LREAL**.
- **y = НЕ(x)** - инверсия значения точки/устройства. Только для точек типа **BOOL**.

По умолчанию преобразование АЦП-ТЕ отключено. Преобразование не будет работать, если не задан (нулевой) диапазон АЦП устройства или диапазон ТЕ точки.

Подробнее о том, в какой момент выполняется преобразование АЦП-ТЕ см. п. 15.5 и п. 15.8.

Калибровочное преобразование - преобразование, которое применяется к значению устройства перед записью в точку (для входных точек) или к значению точки перед записью в устройство (для выходных точек):

- **Нет** - преобразование не выполняется.
- **y = B·x + C** - линейное.
- **y = A·x·x + B·x + C** - квадратичное.

Преобразование выполняется только для входных и выходных точек типа **INT**, **DINT**, **REAL**, **LREAL**. По умолчанию преобразование АЦП-ТЕ отключено. Преобразование не будет работать, если не задан (нулевой) диапазон АЦП устройства или диапазон ТЕ точки.

Примечание. В общем случае для коррекции значения устройств рекомендуется использовать калибровочное преобразование точки, так как коэффициенты калибровки понятнее считать в единицах ТЕ, а не в АЦП. Но если требуется считывать значение с устройства сразу в несколько точек так, чтобы у всех было одинаковое значение, то вместо преобразования у каждой точки можно использовать одно калибровочное преобразование у устройства.

Если точка привязана одновременно и к устройству ввода на чтение и к устройству вывода на запись, то калибровочное преобразование применяется только один раз согласно следующим правилам:

- для точек типа **Входная** калибровочное преобразование точки выполняется только при чтении данных из устройства в точку,
- для точек типа **Выходная** калибровочное преобразование точки выполняется только при записи значения из точки в устройство,

- для точек типа **Виртуальная** и **Диагностическая** калибровочное преобразование точки не выполняется никогда.

Например, значение входной точки считывается из AIN канала ввода и сразу передается сервером по протоколу Modbus во внешние системы. В этом случае калибровочное преобразование точки будет выполняться только при чтении значения из AIN канала ввода в точку, при записи значения точки в регистр Modbus преобразование выполняться не будет. Если сделать эту точку выходной, то калибровочное преобразование будет выполняться уже при записи значения в регистр Modbus, а при чтении из канала AIN - нет.

Подробнее о том, в какой момент выполняется калибровочное преобразование см. п. 15.5 и п. 15.8.

A (K1) - коэффициент A калибровочного преобразования.

B (K1) - коэффициент B калибровочного преобразования.

C (K1) - коэффициент C калибровочного преобразования.

A (K2) - не используется, зарезервировано для будущих версий.

B (K2) - не используется, зарезервировано для будущих версий.

C (K2) - не используется, зарезервировано для будущих версий.

Начальное значение - значение, которое присваивается точке при старте узла. Если у точки включено сохранение значения в энергонезависимой памяти, то начальное значение будет заменено значением из энергонезависимой памяти, если таковое имеется.

Зона нечувствительности - величина, на которое должно измениться значение точки, чтобы узел считал значение точки изменившимся. Отклонение значения точки меньше чем на указанную величину не считается изменением, узел не рассылает это значение на другие узлы проекта, значение не меняется на мнемосхеме и не сохраняется в архив. Используется для фильтрации дребезга значения точки. Расчет программ выполняется с текущим значением точки "как есть" без учета зоны нечувствительности. Запись значения точки в устройства вывода выполняется по изменению без учета зоны нечувствительности. Чтобы задать значение в процентах от диапазона технических единиц точки введите символ % после значения.

Нижний метрологический предел - зона обрыва, максимально допустимая величина выхода значения точки за нижний предел технических единиц ТЕ мин, при которой значение еще считается достоверным. Если значение точки станет меньше нижнего предела ТЕ мин более чем на указанную величину, то узел зафиксирует обрыв в связанном с точкой устройстве, и значение точки будет считаться недостоверным. Только для входных точек, связанных с

устройствами ввода. Чтобы задать значение в процентах от диапазона технических единиц точки введите символ % после значения.

Верхний метрологический предел - зона короткого замыкания, максимально допустимая величина выхода значения точки за верхний предел технических единиц ТЕ макс, при которой значение еще считается достоверным. Если значение точки превысит значение верхнего предела ТЕ макс более чем на указанную величину, то узел зафиксирует короткое замыкание в связанном с точкой устройстве, значение точки будет считаться недостоверным. Только для входных точек, связанных с устройствами ввода. Чтобы задать значение в процентах от диапазона технических единиц точки введите символ % после значения.

Скорость изменения - максимально допустимая величина изменения значения точки за 1 секунду. Если значение точки в течение 1 секунды изменяется более чем на указанную величину, то оно считается недостоверным. Только для входных точек, связанных с устройствами ввода. Чтобы задать значение в процентах от диапазона технических единиц точки введите символ % после значения. Подробнее об установке недостоверности см. п. 7.7.2.

Гистерезис тревоги - погрешность определения выхода значения точки за предел пороговой тревоги. Используется для вычисления состояния событий типа **Аналоговые пределы**. Подробнее об использовании гистерезиса тревоги см. п. 11.7.

LLL - нижний минимальный аварийный придел пороговой тревоги. Используется для вычисления состояния событий типа **Аналоговые пределы**. Подробнее об использовании предела см. п. 11.7.

LL - нижний аварийный придел пороговой тревоги.

L - нижний предупредительный предел пороговой тревоги.

L норм. - нижний нормальный предел пороговой тревоги.

H норм. - верхний нормальный предел пороговой тревоги.

H - верхний предупредительный предел пороговой тревоги.

HH - верхний аварийный предел пороговой тревоги.

HHH - верхний максимальный аварийный предел пороговой тревоги.

Репликация значений К1-К2 - не используется, зарезервировано для будущих версий.

Контроль расхождения значений К1-К2 - не используется, зарезервировано для будущих версий.

Запрет переключения К1-К2 при расхождении - не используется, зарезервировано для будущих версий.

Монопольная точка К1-К2 - не используется, зарезервировано для будущих версий.

Причина установки недостоверности - список коротких названий битов статуса точки, разделенных точкой с запятой. Установка одного из указанных битов приводит к установке бита недостоверности значения, снятие всех битов приводит к снятию бита недостоверности. Правило работает только для входных и выходных точек. Подробно о битах качества значения см. п. 7.7.2

Биты статуса, доступные оператору - список коротких названий битов статуса, определяющих режимы обработки значения точки. Указанные биты статуса будут доступны оператору для установки и снятия по команде с мнемосхем. Подробнее о битах режима работы см. п. 7.7.3.

Уровень передачи сигнала - уровень передачи значения точки с узла-владельца узлы и системы:

- **Локальный узел** - узел-владелец не передает значение точки на другие узлы проекта.
- **Узлы проекта** - значение точки передается только на другие узлы проекта. Серверы данных не могут передавать значение в другие системы управления.
- **Вышестоящие системы** - значение точки может быть передано в другие системы управления.

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства точки и отправлять команды изменения значения этой точки с мнемосхем. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства точки и отправлять команды изменения значения этой точки с мнемосхем. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

7.5 Тип данных точки

Тип данных определяет диапазон допустимых значений точки, а также набор применимых к значению калибровочных преобразований.

Каскад-САУ для хранения значений точек использует набор типов данных из стандарта **МЭК 61131-3**. Подробнее об поддерживаемых типах данных см. п. 9.6

7.6 Формат отображения значения точки

Свойство **Формат отображения** предназначено для задания строки формата, в котором будет отображаться значение точки на мнемосхемах, трендах и в списке событий.

Формат отображения может быть задан только для точек типа **REAL**, **LREAL**, **TIME** **DATE_AND_TIME**.

7.6.1 Формат отображения значений с плавающей точкой

Для значений точек типа **REAL** и **LREAL** (значение с плавающей точкой) формат отображения позволяет указать количество цифр, выводимых до десятичной запятой и количество десятичных знаков, до которых следует округлить число при выводе.

Строка формата может содержать следующие символы:

- **0** - Место цифры. Символ «0» определяет, что в этой позиции выводимой строки должна находиться цифра из соответствующей позиции значения. Если значение не имеет цифры в указанной позиции, то в этой позиции выводимой строки будет находиться символ «0».
- **#** - Место цифры. Символ «#» определяет, что в этой позиции выводимой строки может находиться цифра из соответствующей позиции значения. Если значение не имеет цифры в указанной позиции, то эта позиция в выводимой строке останется пустой.
- **.** (точка) - Знак десятичного разделителя. Символ "." в строке формата определяет позицию разделителя целой и дробной части числа в выводимой строке. Фактический символ, используемый в качестве десятичного разделителя в выводимой строке, определяется форматом чисел операционной системы. *Для операционной системы Windows формат чисел задается в разделе «Язык и Стандарты» в окне «Панель Управления»*.
- **;** - Разделитель секций для положительных, отрицательных чисел и нулевых значений в строке формата.
- **'xx' "xx"** - Символы, заключенные в одинарные или двойные кавычки, отображаются без изменений и не влияют на формат числа.

Выводимое значение всегда округляется на столько десятичных знаков, сколько есть символов «0» или «#» справа от символа десятичного разделителя в строке формата. Если строка формата не содержит символа десятичного разделителя, то выводимое число округляется до ближайшего целого числа.

Если выводимое число в целой части содержит больше цифр, чем указано в строке формата слева от символа десятичного разделителя, то все дополнительные цифры будут выведены перед первым символом места цифры или первым символом десятичной точки.

Чтобы использовать различные форматы для положительных, отрицательных и нулевых значений строка формата должна содержать одну, две или три секции, разделенные точкой с запятой в соответствии со следующими правилами:

- Если указана одна секция, то строка формата будет применена ко всем значениям.
- Если указано две секции, то первая секция будет применена для положительных значений и нулей, вторая секция будет применена для отрицательных значений.
- Если указано три секции, то первая секция будет применена для положительных значений, вторая для отрицательных значений, третья для нулевых значений.
- Если первая секция (для отрицательных значений) или вторая секция (для нулевых значений) пуста, т.е. если нет ничего между точками с запятой, которые разграничивают секции, то вместо них будет использована секция для положительных значений.

Для значений, имеющих более 17 цифр, всегда используется научное представление с экспонентой.

Примеры использования форматов:

Строка формата	**0**	**0.5**	**1**	**1234**	**-1234.567**
	0	0.5	1	1234	-1234.567
0	0	1	1	1234	-1235
#		1	1	1234	-1235
0.0	0.0	0.5	1.0	1234.0	-1234.6
0.00	0.00	0.50	1.00	1234.00	-1234.57
0.000	0.000	0.500	1.000	1234.000	-1234.567
0.#	0	0.5	1	1234	-1234.6
0.##	0	0.5	1	1234	-1234.57
0.###	0	0.5	1	1234	-1234.567
#.#		.5	1	1234	-1234.6
#.##		.5	1	1234	-1234.57
#.###		.5	1	1234	-1234.567
.0	.0	.5	1.0	1234.0	-1234.6

.#		.5	1	1234	-1234.6
#.00	.00	.50	1.00	1234.00	-1234.57
#.#0	.00	.50	1.00	1234.00	-1234.57
000.0#	000.0	000.5	001.0	1234.0	-1234.57
0.#;(0.00)	0.0	0.5	1	1234	(1234.57)
;;Ноль	Ноль	0.5	1	1234	-1234.567
Да;;Нет	Нет	Да	Да	Да	Да
Плюс;Минус;Ноль	Ноль	Плюс	Плюс	Плюс	Минус
0%	0%	1%	1%	1234%	-1235%

7.6.2 Формат отображения интервала времени

Для значений точек типа **TIME** (интервал времени) формат отображения позволяет указать количество дней, часов, минут, секунд и миллисекунд интервала. Формат указывается в виде строки, символы которой заменяются при отображении соответствующими значениями интервала времени.

Строка формата может содержать следующие символы:

- **c** - Заменяется строкой интервала времени, состоящей из знака, количества полных суток и времени в длинном формате. Если количество полных суток равно 0, то сутки не отображаются.
- **d** - Заменяется на полное количество дней без ведущего нуля.
- **h** - Заменяется на часы без ведущего нуля (0-23).
- **hh** - Заменяется на часы, используя ведущий ноль (00-23).
- **hhh** - Заменяется на полное количество часов без ведущего нуля.
- **m** - Заменяется на минуты без ведущего нуля (0-59).
- **mm** - Заменяется на минуты, используя ведущий ноль (00-59).
- **mmm** - Заменяется на полное количество минут без ведущего нуля.
- **s** - Заменяется на секунды без ведущего нуля (0-59).
- **ss** - Заменяется на секунды, используя ведущий ноль (00-59).
- **sss** - Заменяется на полное количество секунд без ведущего нуля.
- **z** - Заменяется на миллисекунды без ведущего нуля (0-999).
- **zzz** - Заменяется на миллисекунды, используя ведущий ноль (000-999).
- **zzzz** - Заменяется на полное количество миллисекунд без ведущего нуля.

- **t** - Заменяется строкой короткого времени.
- **tt** - Заменяется строкой длинного времени.
- ***** - Заменяется на символ минуса для отрицательного интервала и пропускается, если интервал положительный.
- **:** - Заменяется на разделитель времени, определенный настройками региональных форматов.
- **'xx' "xx"** - Символы, заключенные в одинарные или двойные кавычки, отображаются без изменений и не влияют на формат даты и времени.

Примеры использования форматов:

Строка формата	**Результат**
*d hh:mm:ss.zzz	10 05:01:04.002
hhh:m:s	245:1:4
d 'сут.' hh 'ч.' m 'мин.'	10 сут. 05 ч. 1 мин.
hhh 'ч.' m 'мин.' s 'с.'	245 ч. 4 мин. 4 с.

7.6.3 Формат отображения даты и времени

Для значений точек типа DATE_AND_TIME (дата и время) формат отображения позволяет указать полный или краткий формат даты, вывод только даты, только времени и т.п. Формат указывается в виде строки, символы которой заменяются при отображении соответствующими значениями даты и времени.

Строка формата может содержать следующие символы:

- **c** - Заменяется строкой даты и времени в соответствии с форматом краткой даты и полного времени операционной системы. Если значение содержит только дату, то время не отображается.
- **d** - Заменяется на число месяца без ведущего нуля (1-31).
- **dd** - Заменяется на число месяца, используя ведущий ноль (01-31).
- **ddd** - Заменяется на сокращенное название дня недели (Пн-Вс).
- **dddd** - Заменяется на полное название дня недели (понедельник-воскресение).
- **ddddd** - Заменяется строкой даты в соответствии с форматом краткой даты операционной системы.
- **dddddd** - Заменяется строкой даты в соответствии с форматом полной даты операционной системы.

- **m** - Заменяется на месяц без ведущего нуля (1-12*). Если символ m указан непосредственно после символа часов h (см. далее), то вместо месяца отображаются минуты.*
- **mm** - Заменяется на число месяца, используя ведущий ноль (01-12). *Если символы mm указаны непосредственно после символа часов h (см. далее), то вместо месяца отображаются минуты.*
- **mmm** - Заменяется на сокращенное название месяца (янв-дек).
- **mmmm** - Заменяется на полное название месяца (Январь-Декабрь).
- **yy** - Заменяется на год двумя цифрами (00-99).
- **yyyy** - Заменяется на год четырьмя цифрами (0000-9999).
- **h** - Заменяется на час без ведущего нуля (0-23).
- **hh** - Заменяется на час, используя ведущий ноль (00-23).
- **n** - Заменяется на минуты без ведущего нуля (0-59).
- **nn** - Заменяется на минуты, используя ведущий ноль (00-59).
- **s** - Заменяется на секунды без ведущего нуля (0-59).
- **ss** - Заменяется на секунды, используя ведущий ноль (00-59).
- **z** - Заменяется на миллисекунды без ведущего нуля (0-999).
- **zzz** - Заменяется на миллисекунды, используя ведущий ноль (000-999).
- **t** - Заменяется строкой времени в соответствии с форматом краткого времени операционной системы.
- **tt** - Заменяется строкой времени в соответствии с форматом полного времени операционной системы.
- **/** - Заменяется на текущий разделитель компонентов даты, определенный в настройках региональных форматов операционной системы.
- **:** - Заменяется на текущий разделитель времени, определенный в настройках региональных форматов операционной системы.
- **'xx' "xx"** - Символы, заключенные в одинарные или двойные кавычки, отображаются как есть и не влияют на формат даты и времени.

Примеры использования форматов:

Строка формата	**Результат**
d mmmm yyyy 'г.'	3 Декабря 2020 г.
dd/mm/yy	03.12.20
dd/mm/yy h:mm	03.12.20 9:17
dd/mm/yy hh:nn:ss	03.12.20 09:17:05

dd/mm/yy hh:mm:ss	03.12.20 09:17:05
dd/mm/yyyy hh:mm	03.12.2020 09:17
dd/mm/yyyy hh:nn:ss.zzz	03.12.2020 09:17:05.607
dddd',' d mmm yyyy 'г.'	четверг, 3 дек 2020 г.
d/m/yy	3.12.20
h:nn:ss	9:17:05
h:mm:ss	9:17:05
hh:mm:ss	09:17:05
h:m	9:17

7.7 Статус точки

7.7.1 Статус точки Каскад-САУ

Статус точки - набор битов, описывающих качество значения и режимы обработки значения точки задачами ввода-вывода и технологическими программами.

В среде исполнения статус точки хранится в виде целого числа (4 байта). В этом же виде статус сохраняется в архиве и может передаваться в вышестоящие системы сбора данных и управления. Состояние битов статуса можно проверить и изменить в технологических программах или командой с мнемосхемы.

Все биты статуса условно делятся на четыре группы, которые различаются назначением и правилами установки и сброса:

- биты качества значения и аппаратных ошибок;
- биты режимов работы;
- биты технологических пределов;
- служебные биты.

Нормальное состояние значения точки определяется по отсутствию в статусе битов аппаратных ошибок и битов выхода значения за технологические пределы.

7.7.2 Биты качества значения и аппаратных ошибок

Биты качества предназначены для сигнализации о достоверности значения точек типа **ВХОДНАЯ** и **ВЫХОДНАЯ** и наличия *аппаратных ошибок*, приводящей к недостоверности значения.

У входных и выходных точек *биты качества* и *биты аппаратных ошибок* обновляются задачами ввода-вывода на каждом такте среды исполнения. Обновление *битов качества* и *битов аппаратных ошибок* в задачах ввода может быть отключено включением режима *имитации* или *маскирования* (см. п. 7.7.3).

Примечание. Если точка привязана одновременно и к устройству ввода и к устройству вывода, то по состоянию битов аппаратных ошибок нельзя будет однозначно определить причину недостоверности значения, так как неизвестно, на ошибку какого из устройств точки она указывает в данный момент. Поэтому привязывать точку одновременно и к устройству ввода и к устройству вывода рекомендуется только в том случае, если не предполагается никакого анализа бита аппаратных ошибок у точки в программах, трендах или архивах. Разрешается привязывать точку одновременно к устройству ввода-вывода и серверу данных, так как серверы данных не меняют биты аппаратных ошибок.

Биты качества и *биты аппаратных ошибок* могут быть установлены и сняты в технологических программах без каких-либо ограничений. Изменение *битов качества* и *битов аппаратных ошибок* оператором по команде из мнемосхем запрещено.

Серверы данных используют *биты аппаратных ошибок* для расшифровки причины недостоверности значения при передаче данных в вышестоящие системы сбора данных и управления, если передача причины недостоверности значения поддерживается используемым протоколом связи.

Группа *битов качества* и *битов аппаратных ошибок* может быть выделена в технологических программах из статуса точки по маске **0x0020F0F0**. Биты, отвечающие за аппаратные ошибки, взаимоисключающие.

Список битов группы и их описание приведено в таблице ниже.

Бит (маска)	**Сокращение**	**Описание**
Ошибка конфигурации (0x00001000)	КОНФ	Точка является входной или выходной, принадлежит какому-либо узлу, но не привязана к устройству ввода-вывода или у этого устройства неправильный адрес либо параметры, либо в списке задач узла отсутствует

		задача ввода-вывода этого устройства. Бит обновляется задачей ввода-вывода на каждом такте. При установленном бите ввод данных в точку из устройства и вывод из точки в устройство не производится. У точек, для которых в списке задач нет задачи ввода-вывода устройства, бит устанавливается *администратором среды исполнения* (***taskadm,*** см. п. 15.2) один раз при загрузке конфигурации.
Ввод-вывод отключен (0x00002000)	ОТКЛ	Устройство ввода-вывода, к которому привязана точка, или одно из его родительских устройств отключено в проекте, либо недостаточно памяти для работы задачи ввода-вывода на узле. Бит обновляется задачей ввода-вывода на каждом такте. При установленном бите ввод данных в точку из устройства и вывод из точки в устройство не производится.
Ошибка ввода-вывода (0x00000080)	ВВ	Ошибка ввода-вывода данных устройства, к которому привязана точка: нет связи с устройством, еще не получены данные от устройства, ошибка чтения данных, ошибка записи данных в устройство. У входных точек бит обновляется задачами ввода на каждом такте. При установленном бите задача ввода не обновляет значение входной точки устройства. Для выходных точек бит дополнительно выставляется на один такт при ошибке записи значения точки в это устройство, если протокол связи с устройством поддерживает контроль результата записи.
Короткое замыкание (0x00000020)	КЗ	Текущее значение точки больше верхней границы диапазона технических единиц точки **ТЕ макс** на величину, заданную свойством

		Верхний метрологический предел (см. п. 7.4). Бит устанавливается только для входных точек типа **INT**, **DINT**, **READL**, **LREAL**, у которых указан ненулевой диапазон **ТЕ**.
Обрыв датчика (0x00000040)	ОБ	Текущее значение точки меньше нижней границы диапазона технических единиц **ТЕ мин** на величину, заданную свойством **Нижний метрологический предел** (см. п. 7.4). Бит устанавливается только для входных точек типа **INT**, **DINT**, **READL**, **LREAL**, у которых указан ненулевой диапазон **ТЕ**.
Превышение градиента (0x00000010)	ГР	Превышена допустимая скорость изменения значения точки. Бит устанавливается только для входных точек типа **INT**, **DINT**, **READL**, **LREAL**, у которых указан ненулевой диапазон **ТЕ** и ненулевая **Скорость изменения** (см. п. 7.4).
Ошибка связи (0x00008000)	СВЗ	Нет связи с узлом, которому принадлежит точка. Значение точки на локальном узле устарело и может отличаться от реального значения. Подача команды управления по точке с локального узла не возможна. Бит обновляется *клиентом синхронизации данных узлов (**tasknode**, см. п. 16.4)* на каждом такте у всех точек, принадлежащих другому узлу.
Недостоверное значение (0x00020000)	НДСТ	Значение точки недостоверно. У входных и выходных точек бит обновляется задачами ввода на каждом такте, если установлен хотя бы один бит *аппаратной ошибки*, указанный в свойстве **Причина установки недостоверности** точки (см. п. 7.4). Бит сбрасывается, если не установлено ни одного бита аппаратных ошибок,

		перечисленных в свойстве **Причина установки недостоверности** точки.

7.7.3 Биты режимов работы

Биты режимов работы предназначены для управления обработкой значения точки в задачах ввода-вывода и в технологических программах.

Биты режимов работы могут быть установлены и сняты в технологических программах и по команде оператора из мнемосхем. Список *битов режимов работы*, доступных для управления оператору, задается в свойством **Биты статуса, доступные оператору** для каждой точки по отдельности.

Биты режимов работы независимы друг от друга. Группа битов режимов работы может быть выделена в технологических программах из статуса точки по маске **0x001F0000**.

Список битов группы и их описание приведен в таблице ниже.

Бит (маска)	**Сокращение**	**Описание**
Маскирование параметра (0x00010000)	МАСК	Включен режим маскирования. Включение режима маскирования отключает контроль достоверности значения для входных и выходных точек: ошибок ввода-вывода, выхода значения за диапазон, скорости изменения. Текущая причина достоверности значения "замораживается", ввод значения из устройств во входные точки продолжается в обычном порядке. Включение режима маскирования отключает формирование сообщений по точке в системе событий. Бит рекомендуется к использованию в технологических программах для блокировки алгоритмических защитных срабатываний.
Имитация значения (0x00040000)	ИМИТ	Включен режим имитации значения точки (ручной ввод). Включение режима имитации запрещает

		изменение значения и статуса точки задачами ввода-вывода, изменение значения точки в технологических программах и серверах данных. Значение точки может быть изменено только оператором с мнемосхем. У входных и выходных точек при включении режима имитации дополнительно сбрасываются биты аппаратных ошибок и бит недостоверности. При отключении имитации биты аппаратных ошибок и недостоверности восстанавливаются в соответствии с текущим состоянием ввода-вывода устройства точки.
Испытательный режим (0x00080000)	ИСП	Включен испытательный режим. Бит предназначен для использования по усмотрению пользователя, например, для блокировки алгоритмических защитных срабатываний в технологических программах.
Принудительная запись (0x00100000)	ЗАП	Принудительная запись значения. Установка бита у выходных точек приводит к принудительной записи текущего значения точки в устройство вывода. Бит предназначен для повторной записи одного и того же значения в устройство по команде оператора или из технологической программы. Бит автоматически сбрасывается в начале каждого рабочего цикла узла.
Архивация значения (0x00100000)	АРХ	Принудительная архивация значения. Установка бита включает архивацию значения и статуса точки независимо от того, включена ли архивация точки свойством **Сохранение в архиве**. При включении бита автоматически включается архивация связанных с точкой событий. Бит предназначен для временного включения

		архивирования данных по команде оператора с мнемосхемы или из технологической программы (например, на время проведения ответственных операций с оборудованием, пуск агрегатов и т.п.). Архивация точки с установленным битом выполняется с учетом **Зона нечувствительности** точки (см. п. 7.4).

7.7.4 Биты технологических пределов

Биты технологических пределов предназначены для сигнализации о выходе значения точки за заданные пределы.

Для каждой точки могут быть указаны до четырех верхних и четырех нижних пределов:

- технологический предел,
- предупредительный,
- аварийный предел,
- максимальный аварийный предел.

Примечание. Автоматическое снятие и установка битов технологических пределов не производится. Для установки и снятия битов технологических пределов должны быть написана соответствующая технологическая программа.

Биты технологических пределов независимы друг от друга. Группа битов пределов может быть выделена в технологических программах из статуса точки по маске **0x00000F0F**.

Список битов группы указан в таблице ниже.

Бит (маска)	**Сокращение**	**Описание**
Максимальный верхний аварийный предел (0x00000400)	HHH	Значение больше или равно пределу **HHH** точки.
Верхний аварийный предел (0x00000008)	HH	Значение меньше предела **HHH**, но больше или равно пределу **HH** точки.
Верхний предупредит. предел	H	Значение меньше предела **HH**, но больше или

(0x00000004)		равно пределу **H** точки.
Верхний технологич. предел (0x00000200)	NH	Значение меньше предела **H**, но больше или равно пределу **NH** точки.
Нижний технологич. предел (0x00000100)	NL	Значение меньше или равно пределу **NL**, но больше предела **L** точки.
Нижний предупредит. предел (0x00000002)	L	Значение меньше или равно пределу **L**, но больше предела **LL** точки.
Нижний аварийный предел (0x00000001)	LL	Значение меньше или равно пределу **LL**, но больше предела **LLL** точки.
Минимальный нижний аварийный предел (0x00000800)	LLL	Значение меньше или равно пределу **LLL** точки.

7.7.5 Служебные биты

Служебные биты статуса точки используются задачами среды исполнения для внутренних нужд и не могут быть установлены или сняты оператором или в технологической программе.

Служебные биты игнорируются при передаче данных в вышестоящие системы сбора данных и управления (например, при конвертации статуса точки в качество тега OPC).

Группа служебных битов может быть выделена в технологических программах из статуса точки по маске **0xFFF00000**.

Список битов группы указан в таблице ниже.

Бит (маска)	**Сокращение**	**Описание**
Целое значение (0x08000000)		Точка имеет тип данных **INT** или **DINT**.
Дискретное значение (0x10000000)		Точка имеет тип данных **BOOL**.
Значение TRUE		Точка типа **BOOL** имеет значение **TRUE**.

(0x20000000)		
Управление импульсом (0x40000000)		Выполняется команда управления импульсом (см. п. 15.6). В настоящий момент выполняется команда изменения значения точки импульсом. Прочие команды управления импульсом по точке игнорируются.
Калибровка параметра (0x80000000)	КЛБР	Включен режим калибровки точки. В режиме калибровки задачи ввода-вывода отключают калибровочное преобразование для точек (значение вводится из устройства в точку и из точки в устройство без изменения), в технологических программах отключается запись значения в точку.

7.7.6 Сообщение об изменение статуса точки

Для каждого бита статуса точки можно включить автоматическое формирование события об изменении состояния бита.

Текст сообщения формируется автоматически из полного названия бита и слова **УСТАНОВЛЕНО** или **СНЯТО**. Например, при установке бита ошибки связи будет выдано событие "Ошибка связи УСТАНОВЛЕНО".

7.7.7 Отображение статуса на мнемосхеме

Для каждого бита статуса задан свой цвет и приоритет отображения на мнемосхемах.

Цвет бита используется для отображения текущего значения точки на мнемосхеме. Если в статусе точки установлено сразу несколько битов, то значение отображается цветом бита с наивысшим приоритетом.

Цвет и приоритет отображения бита статуса задаются в разделе системных параметров проекта. Настройки параметров бита статуса действуют на все точки проекта.

Приоритет и цвет отображения по умолчанию для битов статуса приведены в таблице ниже.

Бит	**Сокращение**	**Приоритет**	**Цвет**
Калибровка параметра	КЛБР	200	Темно-синий

Маскирование параметра	МАСК	100	Коричневый
Имитация значения	ИМИТ	90	Белый
Ошибка связи	СВЗ	87	Серый
Недостоверные данные	НДСТ	86	Фиолетовый
Ошибка конфигурации	БД	85	Фиолетовый
Ввод-вывод отключен	ОТКЛ	84	Фиолетовый
Ошибка ввода-вывода	ВВ	83	Фиолетовый
Короткое замыкание	КЗ	82	Фиолетовый
Обрыв датчика	ОБ	81	Фиолетовый
Превышение градиента	ГР	80	Фиолетовый
Испытательный режим	ИСП	70	Бирюзовый
Макс. верхний аварийный предел	HHH	62	Красный
Мин. нижний аварийный предел	LLL	61	Красный
Верхний аварийный предел	HH	52	Красный
Нижний аварийный предел	LL	51	Красный
Верхний предупредительный предел	H	42	Желтый
Нижний предупредительный предел	L	41	Желтый
Верхний технологический предел	NH	3	Темно-зеленый
Нижний технологический предел	NL	2	Темно-зеленый
Нормальное состояние	НОРМ	0	Зеленый

7.7.8 Таблица соответствия статуса Каскад-САУ качеству тегов стандарта OPC

В таблице ниже приведено соответствие битов статуса точки Каскад-САУ значениям качества тегов OPC.

Бит (маска)	**Качество тега (маска OPC)**
Ошибка конфигурации (0x00001000)	Bad, Configuration Error 4 (0x04)
Ввод-вывод отключен (0x00002000)	Bad, Not Connected 28 (0x1C)
Ошибка ввода-вывода (0x00000080)	Bad, Device Failure 12 (0x0C)
Ошибка связи (0x00008000)	Uncertain, Last Usable Value 68 (0x44)
Короткое замыкание (0x00000020)	Uncertain, Sensor Not Accurate, High Limited 86 (0x56)
Обрыв (0x00000040)	Uncertain, Sensor Not Accurate, Low Limited 85 (0x55)
Превышение градиента (0x00000010)	Uncertain, Sensor Not Accurate 80 (0x50)
Имитация значения (0x00040000)	Good, Local Override 216 (0xD8)
Мин. нижний аварийный предел (0x00000800)	Good, Low Limited 193 (0xC1)
Нижний аварийный предел (0x00000001)	Good, Low Limited 193 (0xC1)
Нижний предупредительный предел (0x00000002)	Good, Low Limited 193 (0xC1)
Верхний предупредительный предел (0x00000004)	Good, High Limited 194 (0xC2)
Верхний аварийный предел (0x00000008)	Good, High Limited 194 (0xC2)

Макс. верхний аварийный предел (0x00000400)	Good, High Limited 194 (0xC2)
Остальные биты, нормальное состояние	Good, Non-specific 192 (0xC0)

7.8 Сохранение значения точки в энергонезависимую память

Для восстановления значений точек после перезапуска узла, используется энергонезависимая память.

Как правило, значение точки узла сохраняется в энергонезависимую память, если эта точка используется как уставка процесса, которую вручную ввел оператор, или когда нужно при перезапуске узла сохранить промежуточное значение программы. Сохранение статуса точки в энергонезависимую память используется в случае, когда требуется запомнить режим работы точки, установленный оператором, например, режим имитации значения (7.7.3).

Тока может сохраняться в энергонезависимую память автоматически либо вручную.

Автоматическое сохранение включается с помощью свойства **Сохранение в энергонезависимой памяти** точки (см. п. 7.4) для каждой точки по отдельности. В этом случае среда исполнения автоматически записывает в энергонезависимую память значение и статус точки в конце рабочего цикла.

Для сохранения вручную используются функции *NVRAM_SET_VALUE* и *NVRAM_SET_STATUS* в программе. Ручное сохранение используется, если не нужно записывать значение постоянно, но только при каких-то условиях. Значение точки, записанное в энергонезависимую память в программе, не будет автоматически считано в точку при старте необходимо считывать из энергонезависимой памяти также в программе.

Примечание. Можно вручную записать значение для точки, у которой включено автоматическое сохранение в NVRAM. Однако это не имеет смысла, так как записанное вручную значение будет в конце цикла автоматически перезаписано текущим значением точки.

При удалении точки из проекта и при отключении сохранения в энергонезависимую память ее значение и статус автоматически удаляется из энергонезависимой памяти узла. Удаление делается сразу после загрузки новой конфигурации на узел.

В проектах с несколькими узлами каждый узел из них имеет свою энергонезависимую память. При этом значение точки сохраняется в памяти того узла, которому принадлежит

точка. Если точка не принадлежит ни одному проекту, то каждый узел проекта будет сохранять ее в свою энергонезависимую память и восстанавливать при старте узла.

8 Устройства ввода-вывода

8.1 Ввод-вывод данных устройств

Узлы могут читать и записывать данные с различных устройств ввода-вывода, счетчиков, корректоров расхода, измерителей, контроллеров и прочих.

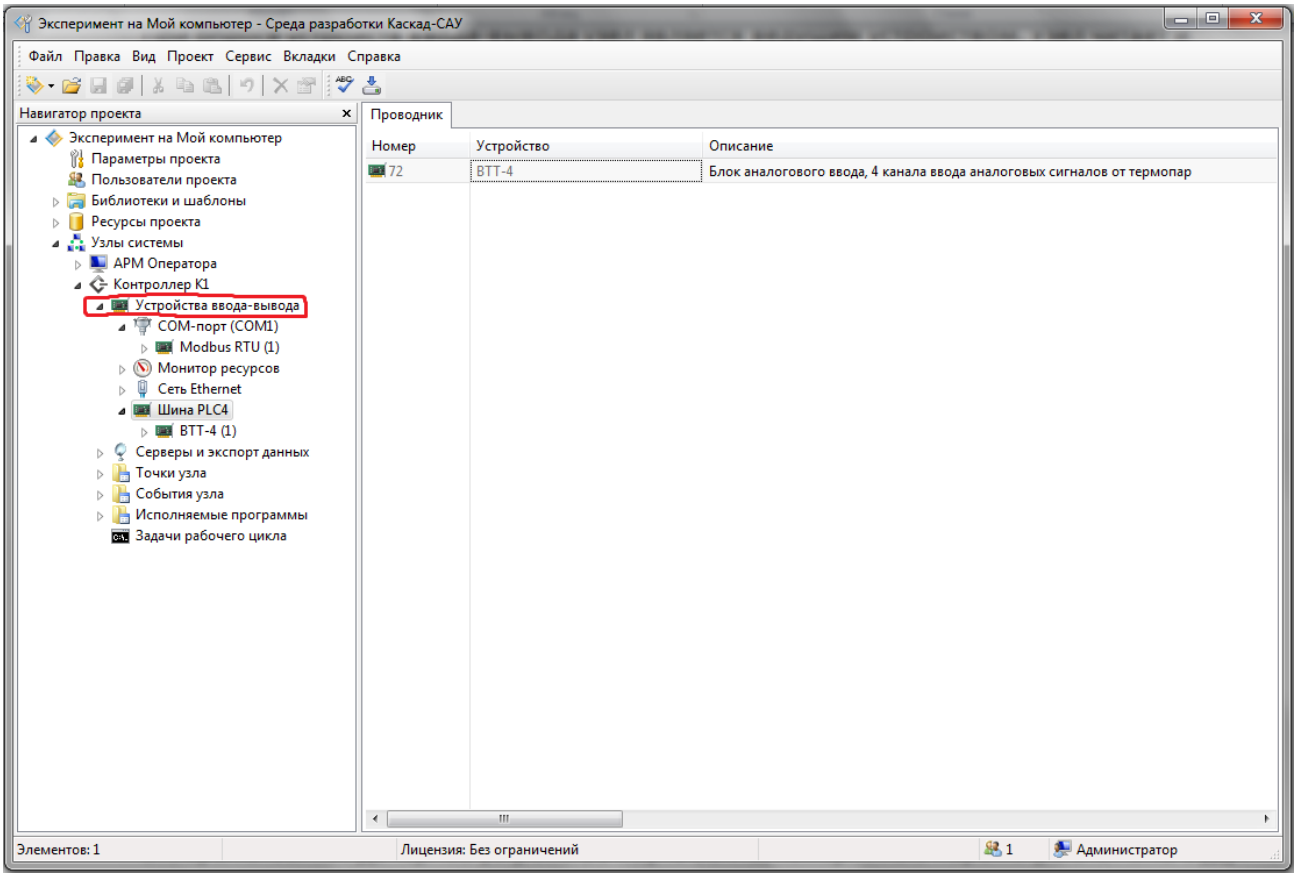
Чаще всего, устройства подключаются к узлу через последовательный порт RS-232/RS-485 или по сети Ethernet. Это универсальные подключения, которые есть практически у каждого современного компьютера. Реже узел может опрашивать устройства, подключенные через специализированную шину или порт ввода-вывода. Например, блоки ввода-вывода аналоговых и дискретных сигналов, подключенные к программируемому контроллеру.

При опросе устройств ввода-вывода узел является ведущим устройством. Узел читает и записывает данные в устройства. Подробнее о вводе-выводе данных с устройств см. п. 15.6 и п. 15.8.

Примечание. Читать и записывать данные устройств могут все узлы проекта, включая АРМ оператора и серверы. Например, АРМ может считывать показания источника бесперебойного питания. Сервер может считывать время с устройства точного времени.

8.2 Конфигурация устройств ввода-вывода

Для настройки конфигурации устройств ввода-вывода, подключенных к узлу, используется папка **Устройства ввода-вывода** узла в дереве проекта. Эта же папка используется для настройки виртуальных устройств узла и встроенных устройств контроллеров, например, сторожевых таймеров.



Среда исполнения использует конфигурацию устройств узла для запуска и настройки драйверов ввода-вывода в процессе работы.

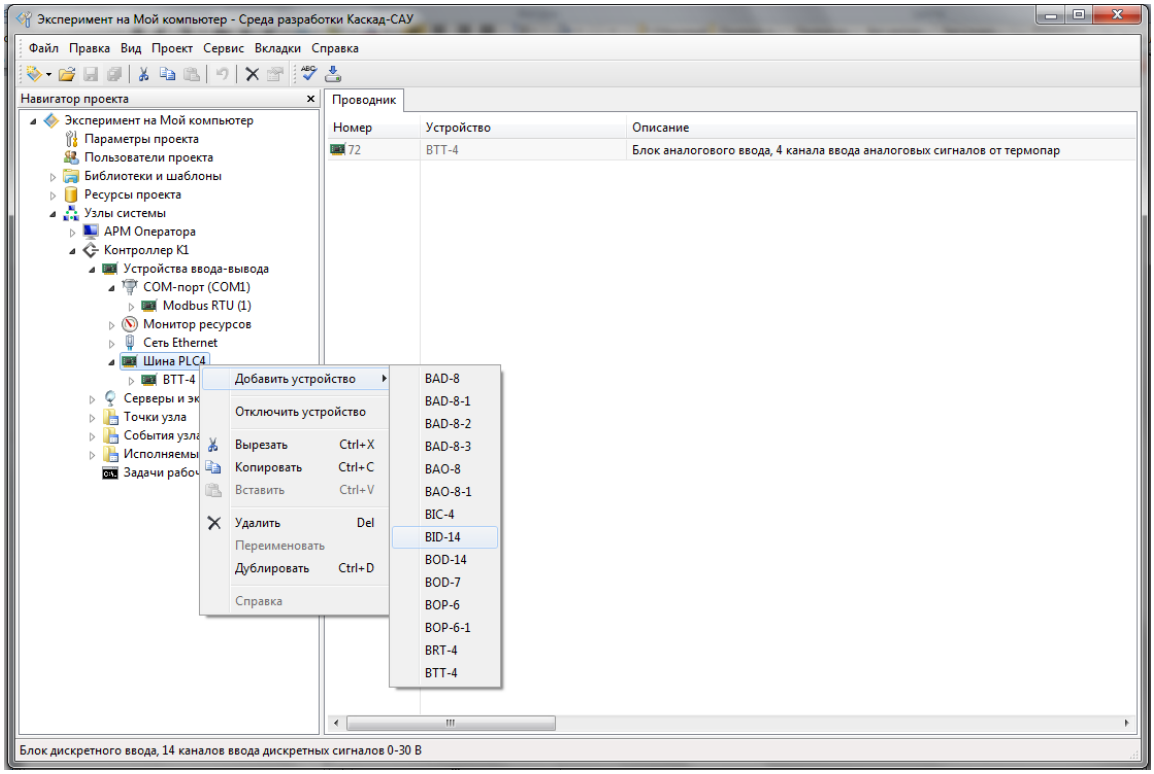
Все устройства ввода-вывода узла отображаются в папке в вид дерева в порядке физического подключения к узлу. Например, если к контроллеру через последовательный порт *COM1* подключен счетчик электрической энергии, то в папке **Устройства ввода-вывода** порт *COM1* будет отображаться как устройство первого уровня, счетчик электрической энергии будет отображаться как дочернее устройство порта *COM1*, измеряемые параметры счетчика будут отображаться в дереве как дочерние устройства счетчика.

*Примечание. По умолчанию папка **Устройства ввода-вывода** отображается только для узлов типа **Контроллер**. Чтобы включить отображение папка на другом узле, щелкните на его названии в дереве проекта и выберите команду Видимые ресурсы узла > Устройства ввода-вывода.*

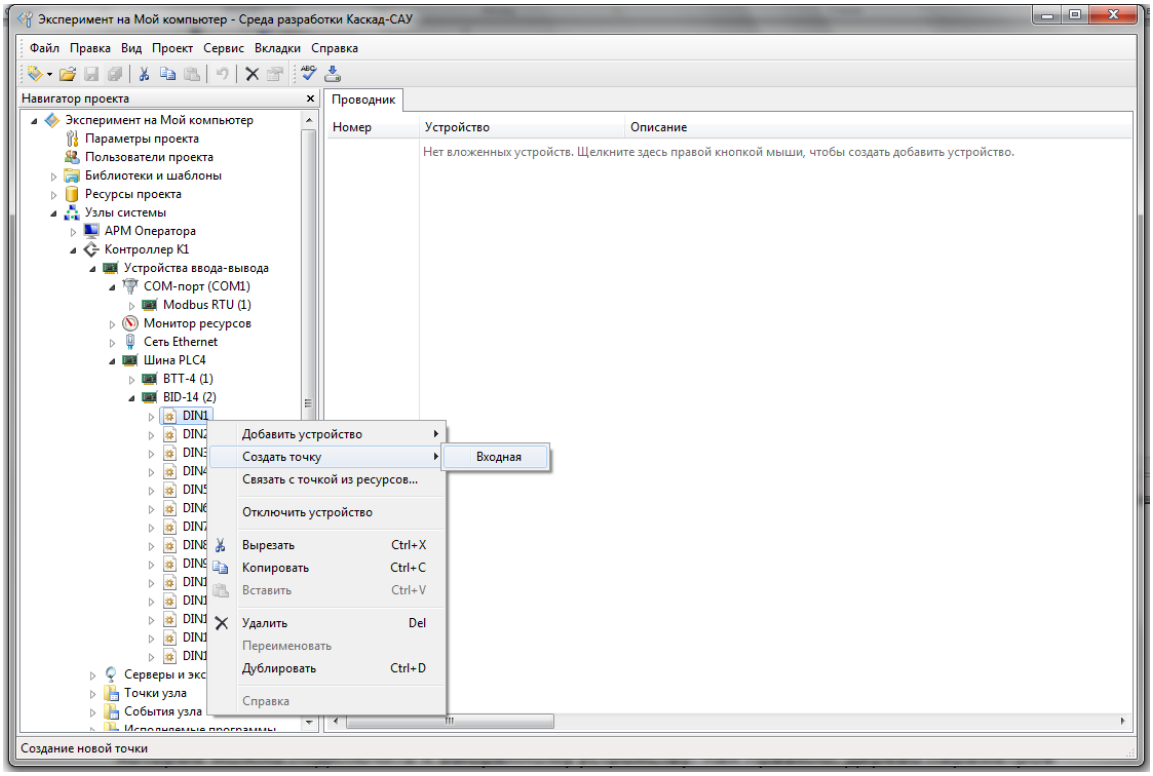
8.3 Добавление нового устройства

Для добавления нового устройства:

- Щелкните правой кнопкой на папке **Устройства ввода-вывода** узла, в контекстном меню выберите команду **Добавить устройство** и затем выберите нужное устройство из предлагаемого списка.



- Щелкните правой кнопкой на значке добавленного устройства и добавьте подключенные к нему устройства, каналы или регистры и так далее до тех пор, пока в дерево не будет добавлено все устройство.



Среда разработки автоматически показывает в контекстном меню только те устройства, которые можно подключить к выбранному устройству. Как правило, дерево параметров устройств ввода-вывода соответствует описанию протокола обмена данными с этим устройством. Перед добавлением нового устройства рекомендуется ознакомиться с документацией на устройство.

8.4 Шаблоны устройств

Для определения того, какие устройства можно подключать к выбранному, среда разработки использует шаблоны устройств.

Шаблоны всех поддерживаемых устройств хранятся в проекте. Список шаблонов заносится в проект в момент создания и определяется версией среды разработки Каскад-САУ, в которой он был создан.

Если в новой версии Каскад-САУ появляется поддержка нового устройства, то необходимо обновить версию проекта, чтобы добавить в него шаблон нового устройства и чтобы это устройство появилось в списке устройств, доступных для создания. Подробнее об обновлении версии проекта см. п. 5.13.

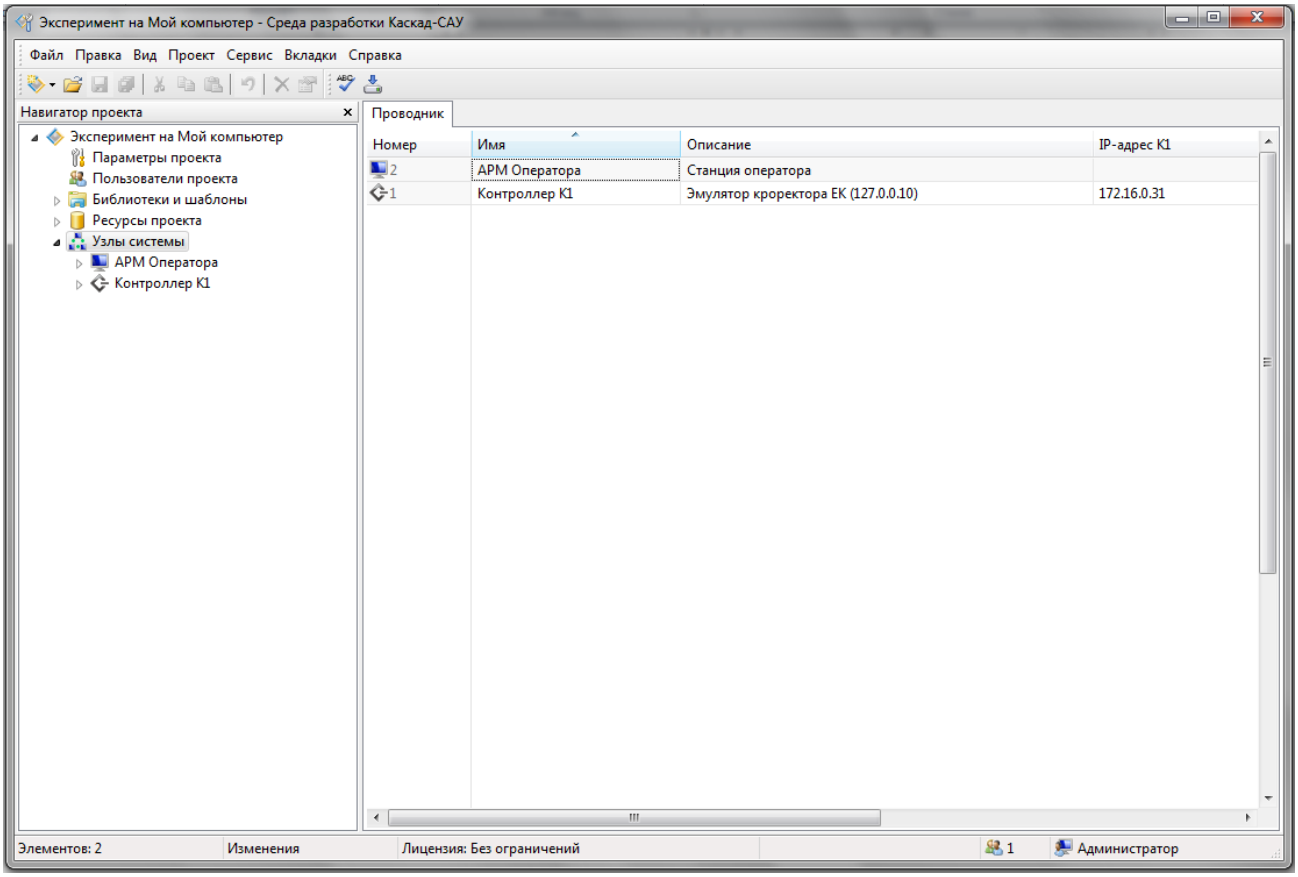
8.5 Адрес устройства

Некоторым устройствам необходимо указать адрес подключения либо адрес, который используется для опроса. Это адрес указывается в дереве проекта в скобках после названия устройства.

Для изменения адреса устройства используйте один из следующих способов.

Способ 1 .Щелкните на названии устройства в дереве и в контекстном меню выберите команду **Переименовать**.

Способ 2. Дважды щелкните левой кнопкой значок *родительского устройства* и перейдите в таблицу на вкладке **Проводник**. Выделите строку с устройством и введите адрес в колонку **Адрес**.



8.6 Свойства устройства

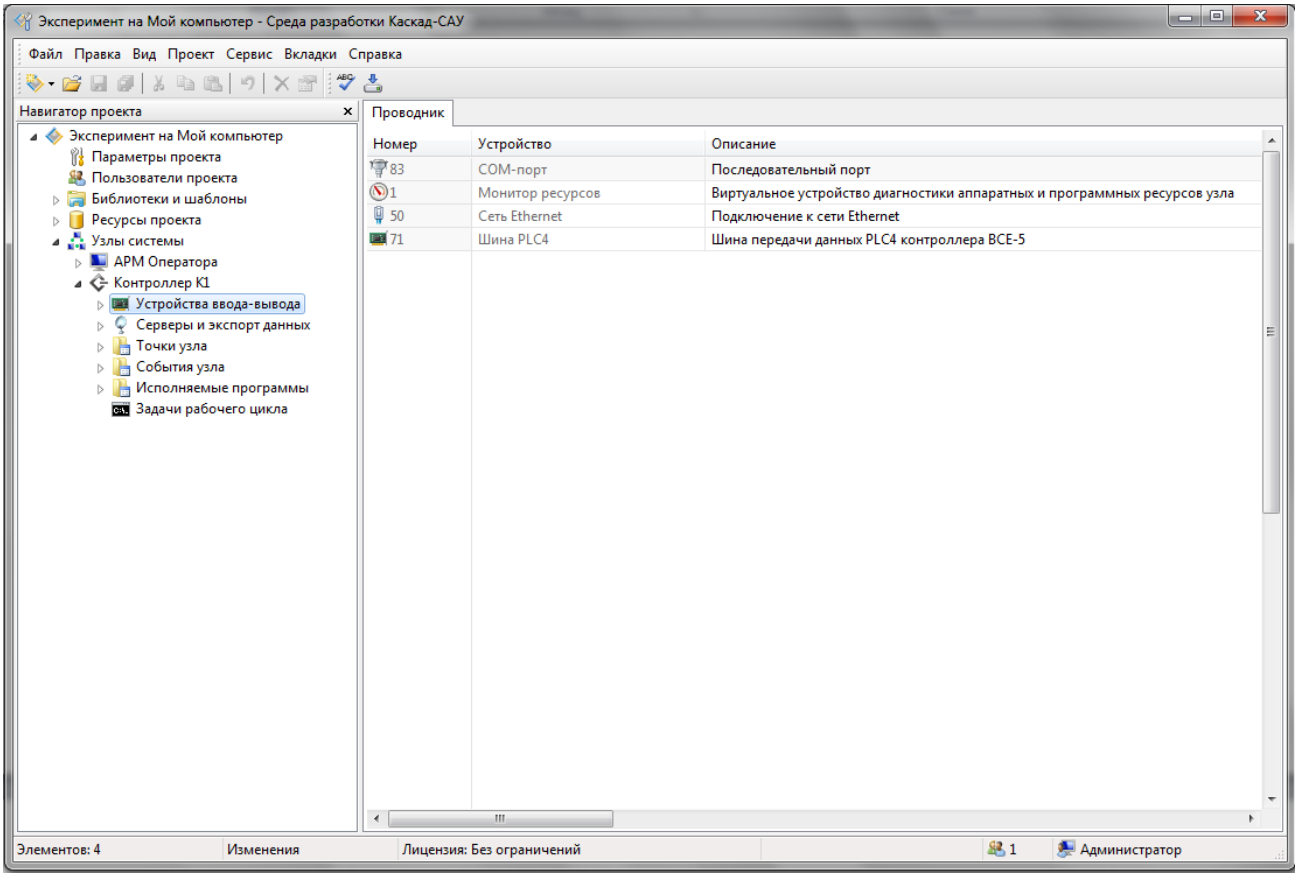
Для настройки свойств устройства дважды щелкните левой кнопкой значок родительского устройства и перейдите в таблицу на вкладке **Проводник**. Выделите строку с устройством и введите значения свойств.

Номер - порядковый номер устройства в таблице устройств. Значение свойства выбирается автоматически при создании устройства и не может быть изменено. При удалении устройства его номер может быть назначен следующему новому устройству.

ИД - уникальный номер устройства в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Устройство - тип устройства, выбранный при создании. Не может быть изменено.

Описание - описание устройства, не более 127 символов.



Адрес - адрес подключения к устройству: имя порта, адрес устройства на линии или другой адрес, необходимый для работы с устройством. Значение определяется типом устройства. Некоторые устройства не имеют адреса. Некоторые устройства имеют фиксированный адрес, который нельзя изменить. Адрес указывается в дереве проекта в скобках после названия устройства.

Состояние - состояние опроса устройства:

- **Отключено** - опрос устройства отключен.
- **Включено** - опрос устройства включен.

Значение точек ввода, подключенных к отключенным устройствам, недостоверно.

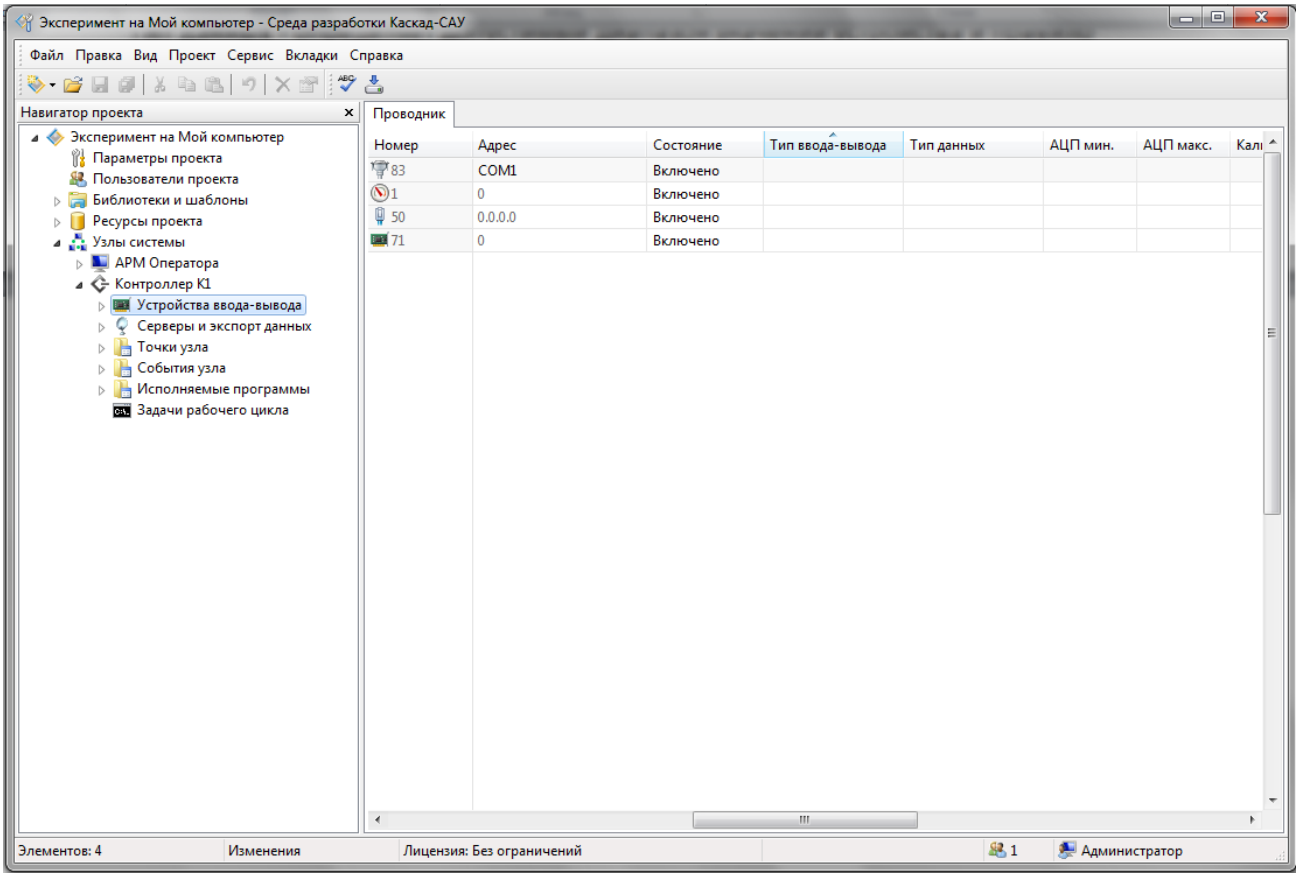
Тип ввода-вывода - чтение для устройств ввода, запись для устройств вывода. Свойство показывает направление передачи данных между устройством и точками узла: чтение из устройства в точки узла или запись из точек в устройство. Значение свойства определяется типом устройства, носит информационный характер и не может быть изменено. Подробнее о вводе данных из устройства в точки см. п. 15.6.

Тип данных - определяет допустимый диапазон значений устройства и правило преобразования значения устройства в тип данных точки. Значение свойства определяется

типом устройства, носит информационный характер и не может быть изменено. Подробнее о типах данных см. п. **Ошибка! Источник ссылки не найден.**.

АЦП мин. - минимальное значение диапазона допустимых значений устройства. Используется для преобразования значения устройства в значение точки в технических единицах (для устройств ввода) и обратно (для устройств вывода). Подробнее о преобразовании значения АЦП в ТЕ см. п. 15.6 и п. 15.8.

АЦП макс. - максимальное значение диапазона допустимых значений устройства. Используется для преобразования значения устройства в значение точки в технических единицах (для устройств ввода) и обратно (для устройств вывода).



Калибровочное преобразование - преобразование, которое выполняется со значением устройства перед преобразованием в диапазон значений ТЕ точки (для устройств ввода) или перед записью значения точки в устройство (для устройств вывода). Вид преобразования зависит от типа данных устройства. Подробнее о применении калибровочного преобразования см. п. 15.6 и п. 15.8.

A (K1) - коэффициент A калибровочного преобразования.

B (K1) - коэффициент B калибровочного преобразования.

C (K1) - коэффициент C калибровочного преобразования.

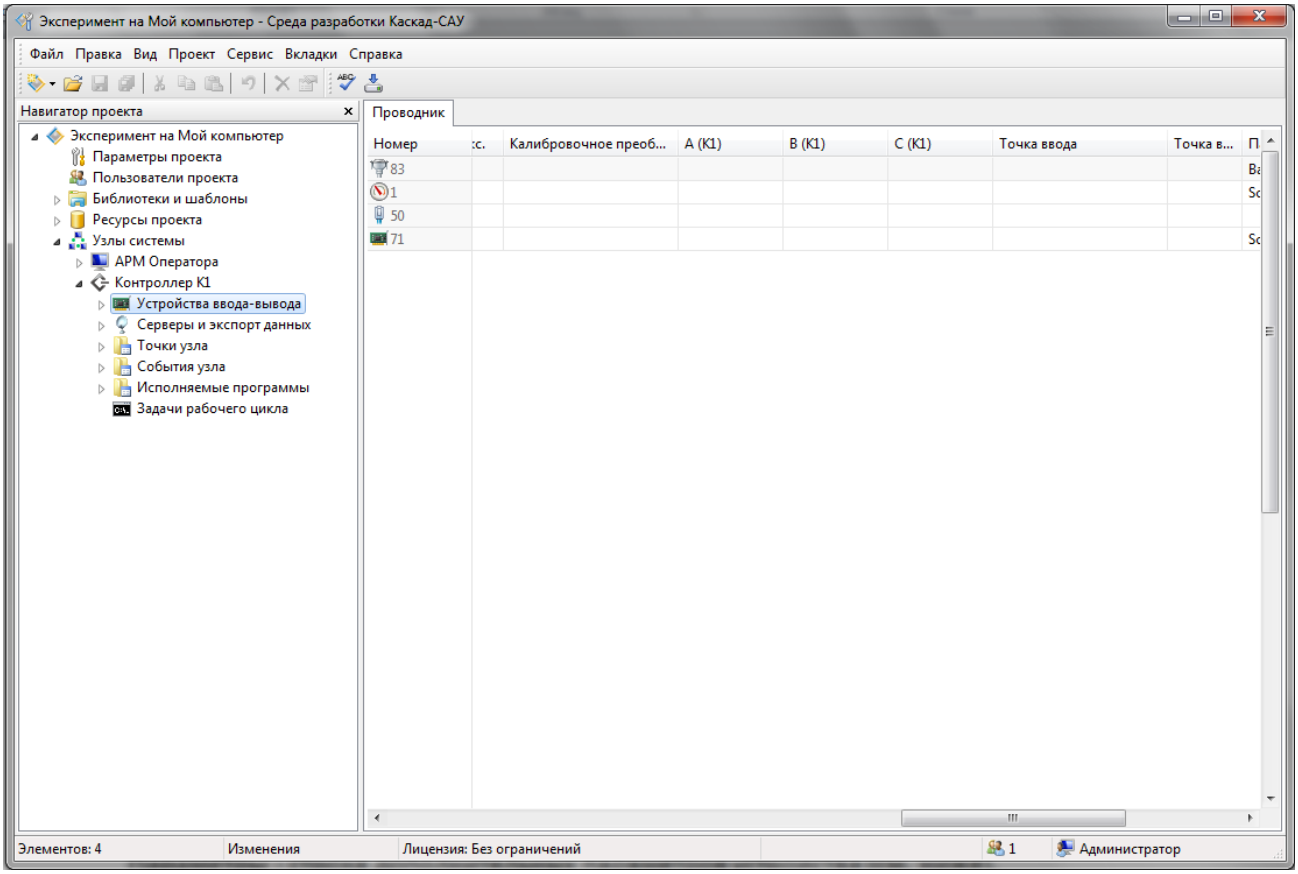
A (K2) - не используется, зарезервировано для будущих версий.

B (K2) - не используется, зарезервировано для будущих версий.

C (K2) - не используется, зарезервировано для будущих версий.

Точка ввода - точка, в которую записывается значение, считанное с устройства ввода. Значение с устройства может записываться одновременно в несколько точек. С устройством ввода можно связывать только точки типа **ВХОДНАЯ**. Поле доступно для редактирования только для устройств ввода. Подробнее о привязке точек к устройствам см.п.XXX.

Точка вывода - точка, значение которой записывается в устройство вывода. Для записи в устройство может использоваться только одна точка. С устройством вывода можно связывать только точки типа **ВЫХОДНАЯ**. Поле доступно для редактирования только для устройств вывода.



Параметры - строка дополнительных параметров устройства (см. ниже).

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть

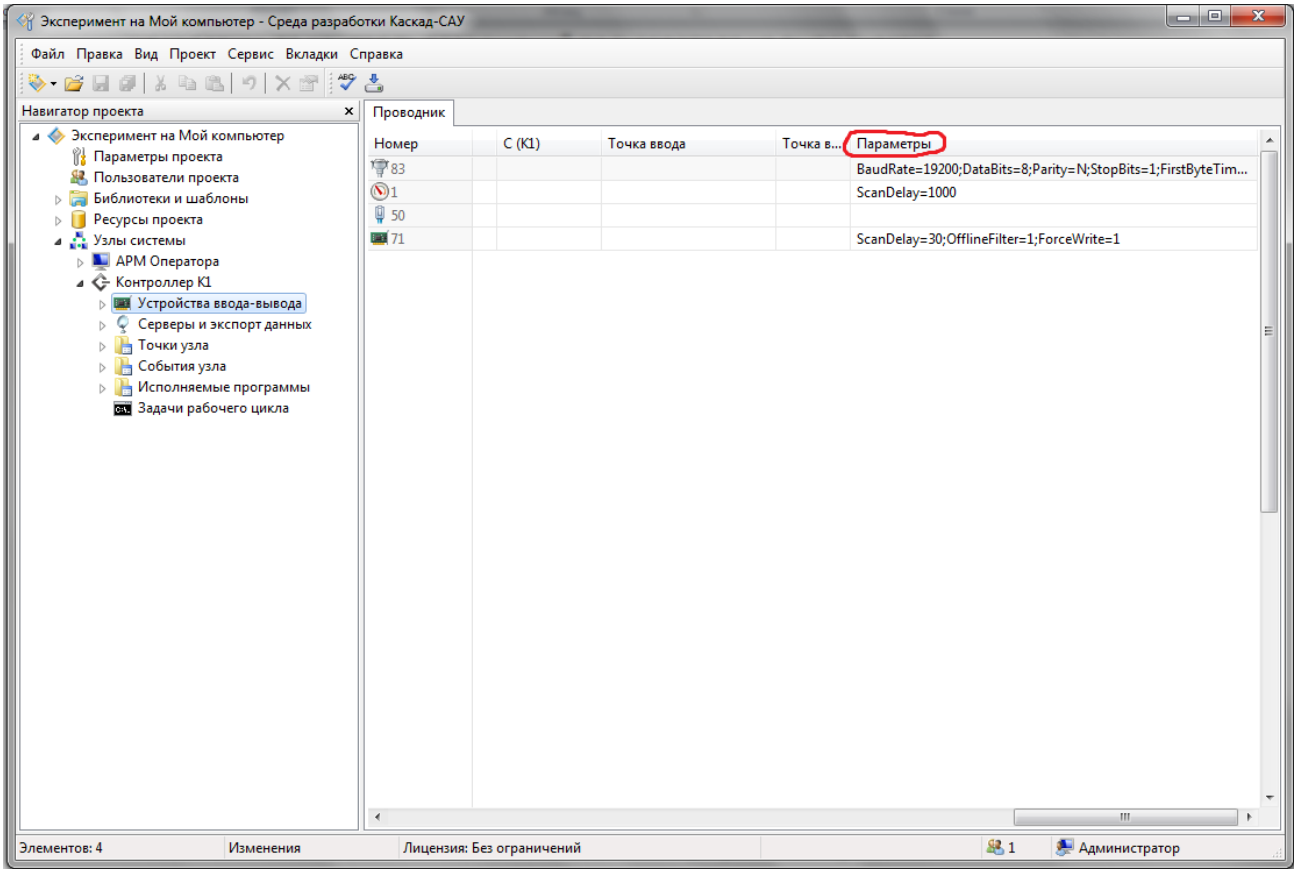
хотя бы одно из указанных разрешений, могут менять свойства устройства. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства устройства. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

8.7 Параметры работы устройства

Свойство **Параметры** устройства содержит строку дополнительных параметров, необходимых для настройки работы устройства. Значения параметров задаются в формате *Параметр=Значение*, параметры разделяются точкой с запятой.

Список параметров зависит от типа устройства. Например, для COM-порта с помощью строки Параметры задаются скорость, биты данных, четность и стоповые биты. Полный список параметров приведен в описании устройства.

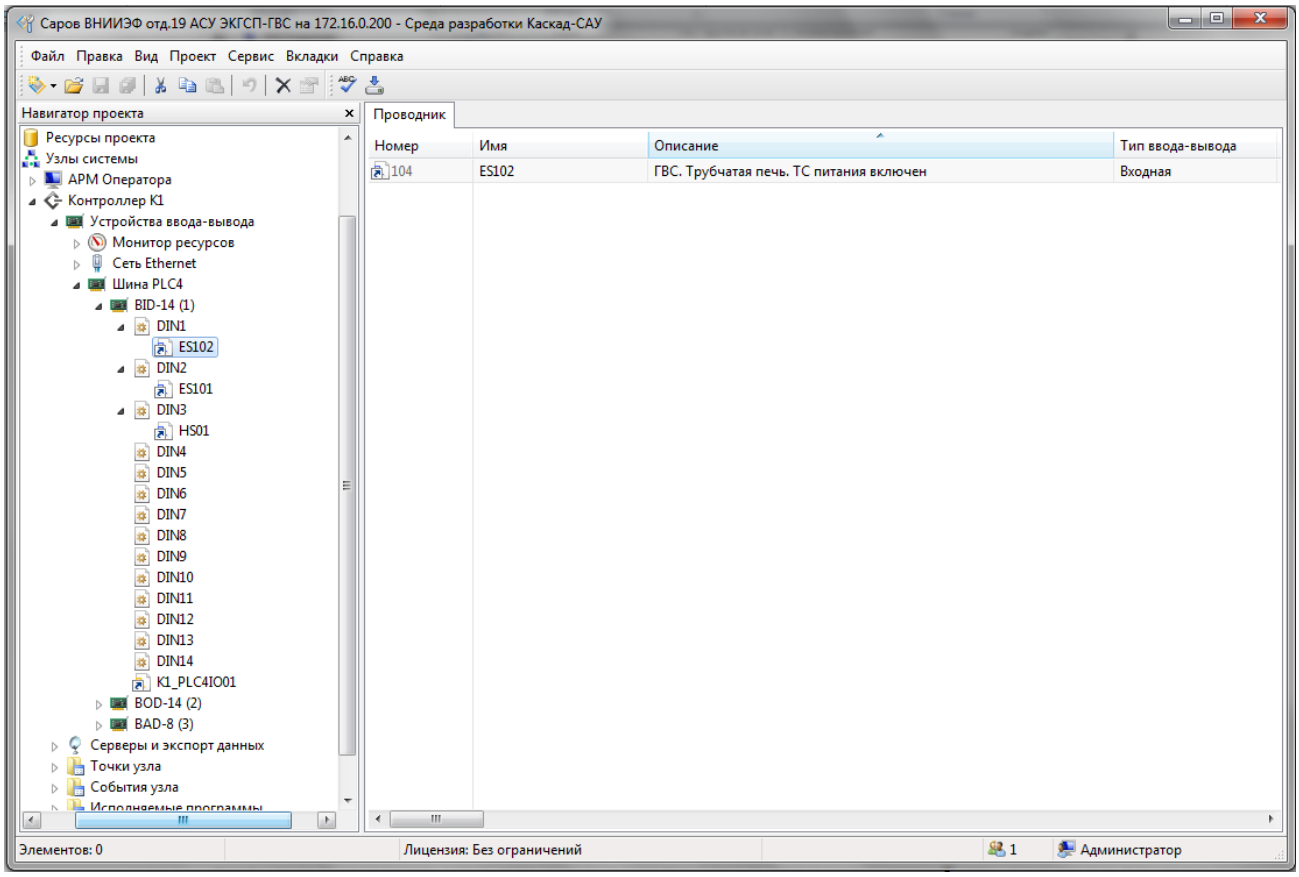


*Примечание. Все параметры, необходимые для настройки работы устройств, настраиваются с помощью свойства **Параметры** непосредственно в проекте. Никаких дополнительных настроек в конфигурационных файлах на узлах не требуется.*

8.8 Чтение и запись значений устройства

Чтобы выполнять чтение или запись значений устройства нужно связать его точкой:

- значение устройства ввода будет записано в связанную с ним **ВХОДНУЮ** точку;
- значение **ВЫХОДНОЙ** точки будут записано в устройство вывода, с которым она связана;
- в **ДИАГНОСТИЧЕСКУЮ** точку будет записан признак наличия связи с устройством: 0/False - нет связи, 1/True - связь установлена.



В дереве проекта привязанные к устройству точки отображаются в виде ярлыков, прикрепленных к значку устройства:

- Значение одного устройства ввода может быть записано одновременно в несколько входных точек. В этом случае у устройства будет несколько ярлыков привязанных к нему точек ввода. К одному устройству ввода может быть привязано неограниченное количество точек.
- Одна входная точка может получать данные только от одного устройства ввода. При попытке привязать уже привязанную точку ко второму устройству ввода ее привязка к предыдущему устройству будет автоматически удалена.

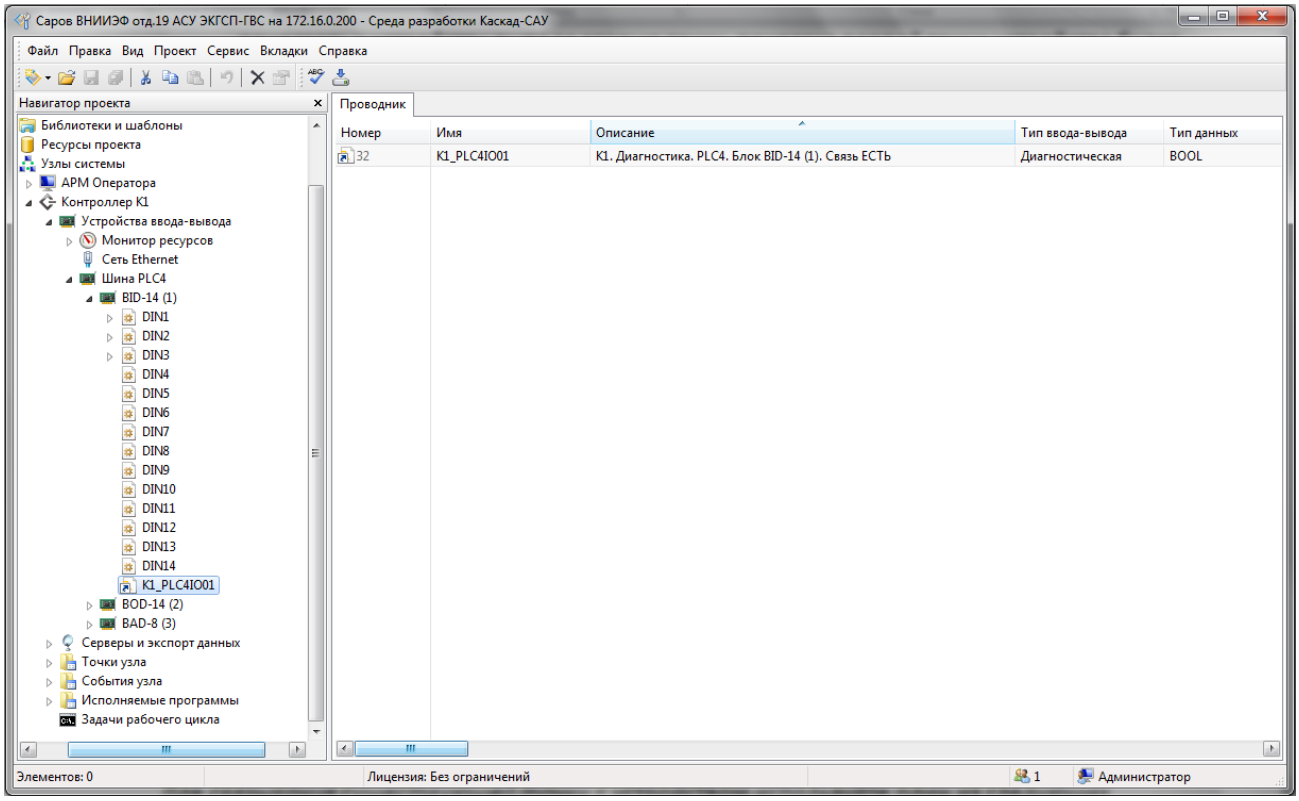
- Одно устройство вывода может получать данные только одной точки. При попытке привязать к устройству вывода вторую точку, привязка первой точки устройства будет автоматически удалена.
- Значение одной точки может быть записано одновременно в несколько устройств вывода. В этом случае ярлык этой точки будет повторяться сразу у нескольких разных устройств. Количество выходных устройств, в которые может быть записано значение точки, не ограничено.

8.9 Диагностика связи с устройствами

Некоторые устройства поддерживают диагностику наличия связи.

Для ввода диагностики связи нужно связать устройство точкой типа **ДИАГНОСТЧЕСКАЯ**. Тогда в такую точку будет записано не значение устройства, а признак наличия связи с ним:

- **0** или **False** - нет связи;
- **1** или **True** - связь установлена.

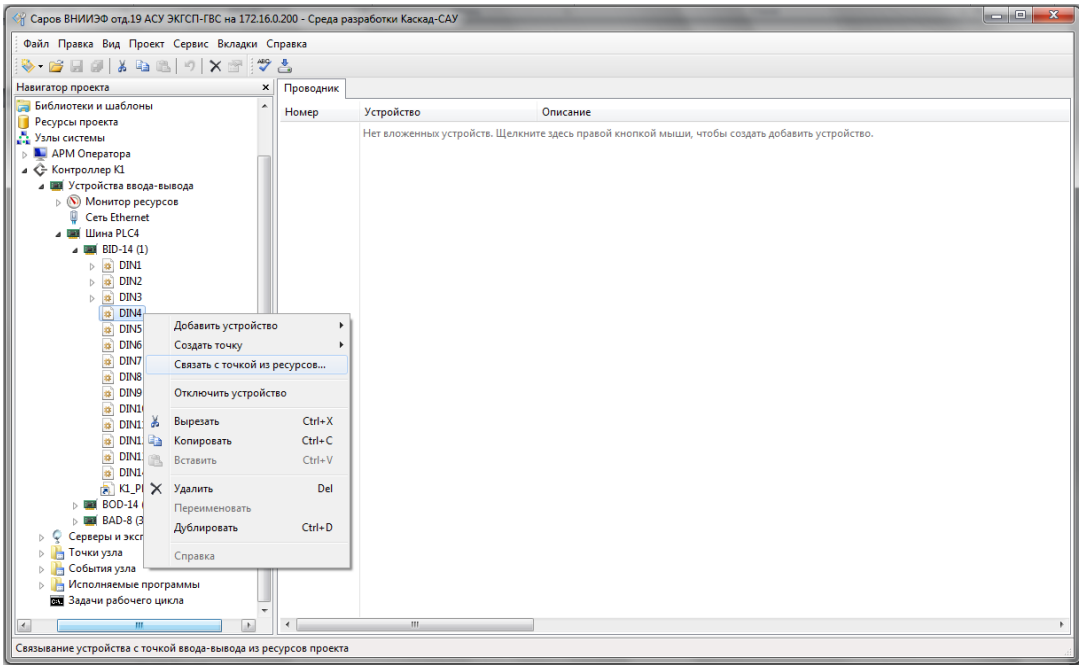


8.10 Связывание устройств с точками

Для *создания новой точки*, связанной с устройством, щелкните правой кнопкой устройство и выберите команду **Создать точку**, затем тип точки. Среда разработки автоматически создаст новую точку и свяжет ее с устройством.

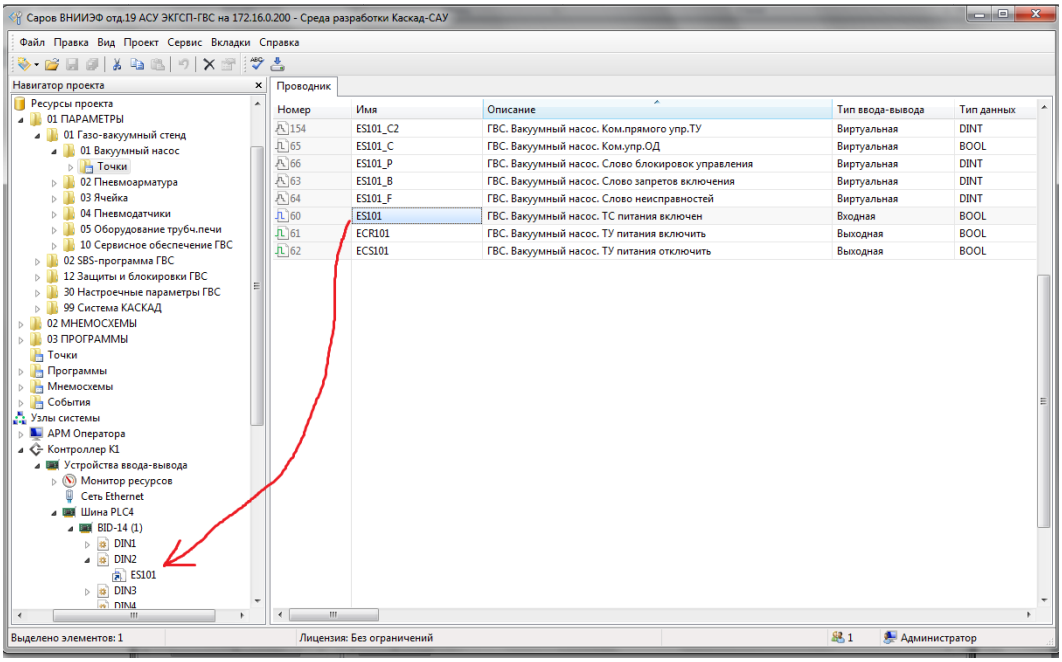
Для *связывания существующей точки* с устройством используйте один из следующих способов:

Способ 1. Щелкните правой кнопкой устройство и выберите команду **Связать с точкой из ресурсов**. В диалоговом окне **Выбор точки** выберите нужную точку и нажмите **ОК**.



Способ 2. В дереве проекта раскройте папку **Точки узла**, найдите в ней нужную точку и перетащите ее на устройство.

Способ 3. В дереве проекта раскройте папку **Ресурсы проекта**, найдите в ней нужную точку и перетащите ее на устройство.



Способ 4. Дважды щелкните левой кнопкой значок *родительского устройства*, перейдите в таблицу на вкладке **Проводник**, в строке устройства щелкните свойство **Точка ввода** или **Точка вывода**, затем щелкните по кнопке с троеточием и выберите точку с помощью диалога.

Примечание. Связанные с устройством ввода точки всегда принадлежат тому узлу, которому принадлежит это устройство ввода. Подробнее о рассылке значения точки на другие узлы проекта см. п 16.4.

9 Программирование логики обработки данных

9.1 Программы обработки данных Каскад-САУ

Алгоритмическая обработка данных на узлах выполняется с помощью программ.

Как правило, программа пишется, чтобы выполнить некоторый технологический алгоритм. В самом простом случае программа получает из памяти узла значения входных точек, полученные с устройств ввода, сравнивает эти значения с какой-либо константой и меняет значения выходных точек. Значения выходных точек выводятся на устройства вывода, реализуя, таким образом, алгоритм управления объектом.

Программы пишутся на языке программирования *Structured Text* или *Function Block Diagram* в *редакторе программ*. Текст программы *компилируется* в байт-код и загружается на узлы. На узлах программы исполняет задача среды исполнения *виртуальная машина IEC 61131-3* (см. п. 15.8).

9.2 Структура программы

9.2.1 Главная функция программы

Исполнение программы начинается с исполнения *главной функции*. Имя главной функции программы совпадает с именем программы.

Главная функция использует для работы *переменные*, *константы* и другие *функции* и *функциональные блоки*.

Переменные программы могут быть *связаны с точками* проекта. Перед исполнением программы значения из точек автоматически копируются в переменные, по окончании расчета программы значения из переменных копируются в точки.

9.2.2 Переменные

Переменная - область памяти программы, используемая для хранения промежуточных значений вычислений (*глобальные* и *локальные* переменные), либо входных и выходных параметров функции и функциональных блоков (*входная* или *выходная* переменная).

Все переменные, используемые в программе, должны быть предварительно объявлены (см. п. 9.10.4).

9.2.3 Константы

Константа - область памяти программы с фиксированным значением. Изменение констант в программе запрещено.

Все константы, используемые в программе, должны быть предварительно объявлены (см. п. 9.10.6).

9.2.4 Функции

Функция - фрагмент кода программы, который можно вызвать на исполнение несколько раз из разных мест программы.

Функция подставляет собой набор выражений, результатом выполнения которых является значение - *результат функции*. Функция может принимать параметры - *входные* и *выходные переменные*. Функция может использовать для хранения промежуточных вычислений *локальные переменные*.

Функция не сохраняет свое состояние от вызова к вызову. Значения выходных и локальных переменных функции перед началом исполнения всегда сбрасывается в значение по умолчанию.

Все функции, используемые в программе, должны быть предварительно объявлены (см. п. 9.10.6).

9.2.5 Функциональные блоки

Значения локальных переменных функционального блока сохраняются от одного вызова функционального блока к другому.

В отличие от функции, каждый экземпляр функционального блока, вызываемый в программе, должен быть объявлен в программе в виде переменной. Эта переменная используется для хранения внутреннего состояния блока. Если вызываемый блок не объявлен явно, то для него во время компиляции программы будет автоматически создана переменная.

Все функциональные блоки, используемые в программе, должны быть предварительно объявлены (см. п. 9.10.9).

9.2.6 Стандартные функции

Стандартные функции - это функции из стандарта IEC 61131-3, которые можно использовать, не определяя их в программе.

Каскад-САУ содержит большой набор стандартных функций и функциональных блоков различных категорий:

- арифметика,
- математика,
- тригонометрия,
- биты и логика,
- выбор и сравнение,
- присваивание
- преобразование типов,
- триггеры и детекторы фронтов,
- счетчики,
- таймеры,
- перечисления и диапазоны,
- массивы,
- дата и время,
- ПИД регулирование,
- газовые расчеты,
- системные функции,
- точки,
- энергонезависимая память.

9.3 Поддерживаемые языки программирования

Для написания программ в Каскад-САУ используются языки программирования из стандарта **МЭК 61131-3**. В Каскад-САУ поддерживаются следующие языки:

- Structured Text
- Function Block Diagram.

Особенности реализации языков согласно приложению D стандарта приведены в п. 9.12.4.

9.4 Язык программирования Structured Text

Язык программирования Structured Text (структурированный текст, далее ST) является текстовым языком, в котором за основу взят язык программирования Pascal.

9.4.1 Выражения

Программа на языке ST состоит из *выражений*, которые вычисляют значения. Выражения разделяются *точкой с запятой*.

Выражения в программе исполняются сверху вниз. За один рабочий цикл узла исполняются все выражения программы, при условии, что перед ними не выполняется оператор RETURN (см. п. 9.4.7) или EXIT (см. п. 9.4.8).

9.4.2 Операнды и операторы

Выражения состоят из *операндов* (название переменной, константа, вызов функции или блока) и *операторов*. Если в одном выражении встречаются несколько операторов, то они вычисляются согласно приоритету, операторы с одинаковым приоритетом вычисляются слева направо.

Список операторов языка ST приведен в таблице ниже.

№	**Описание**	**Символ**	**Пример**	**Приоритет**
1	Скобки	()	(A+B/C), (A+B)/C, A/(B+C)	11 (высший)
2	Вызов функции		LN(A), MAX(X,Y)	10
3	Отрицание	-	-A, - A	8
4	Унарный плюс	+	+B, + B	8
5	Дополнение	NOT	NOT C	8
6	Возведение в степень	**	A**B, B ** B	7
7	Умножение	*	A*B, A * B	6
8	Деление	/	A/B, A / B / D	6
9	Модуль	MOD	A MOD B	6
10	Сложение	+	A+B, A + B + C	5
11	Вычитание	-	A-B, A – B - C	5
12	Сравнение	< , > , <= , >=	A<B, A < B	4
13	Равенство	=	A=B	4

14	Неравенство	<>	A<>B, A <> B	4
15a	Логическое И	&	A&B, A & B, A & B & C	3
15b	Логическое И	AND	A AND B	3
16	Логическое исключающее ИЛИ	XOR	A XOR B	2
17	Логическое ИЛИ	OR	A OR B	1 (низший)

9.4.3 Присваивание

Для записи результата вычисления выражения в переменную используется оператор **:=** (присваивание).

Пример:

```
A := B;
```

9.4.4 Сравнение

Оператор **IF** используется, чтобы выполнить одно или несколько выражений, если выполняется определенное условие.

Выражения после оператора **IF** выполняются, если связанное с ним логическое выражение возвращает значение **TRUE**. Если условие возвращает **FALSE**, то выполняются выражения, указанные после оператора **ELSE**.

Пример:

```
IF A = 10 THEN
  A := 0;
  B := B + 1;
ELSIF B = 10 THEN
  B := 0;
ELSE
  A := A + 1;
END_IF;
```

9.4.5 Выбор

Оператор **CASE** используется, чтобы выполнить разные выражения при разных значениях переменной или выражения.

Выражения выполняются, если связанная с **CASE** переменная равна указанному значению. Если значение переменной не равно ни одному из значений, то выполняются выражения, указанные после оператора **ELSE**.

Пример:

```
CASE A OF
  1: B := B + 1;
  2: B := B - 1;
  3..4: B := B * 2;
ELSE
  B := 1;
END_CASE;
```

9.4.6 Циклы

Операторы **FOR**, **WHILE** и **REPEAT** используются, чтобы выполнить группу одно или более выражений несколько раз.

Выражения между операторами **FOR** и **END_FOR** будут выполняться, пока переменная управления циклом не достигнет указанного значения.

Пример:

```
A := 0;
FOR I := 1 TO 3 DO
  A := A + I;
END_FOR;
```

Выражения между оператором **WHILE** и **END_WHILE** будут выполняться, пока связанное с ним логическое выражение возвращает **TRUE**.

Пример:

```
A := 0;
I := 1;
WHILE I <= 3 DO
  A := A + I;
  I := I + 1;
END_WHILE;
```

Выражения между операторами **REPEAT** и **UNTIL** будут выполняться, пока логическое выражение, указанное после оператора **UNTIL** выражение возвращает **FALSE**.

Пример:

```
A := 0;
I := 1;
REPEAT
  A := A + I;
```

```
  I := I + 1;
UNTIL I > 3
END_REPEAT;
```

9.4.7 Возврат из функции

Оператор **RETURN** используется для раннего выхода из функции или функционального блока. Остальные выражения функции или функционального блока, стоящие после оператора **RETURN**, не выполняются.

Вызов оператора **RETURN** в теле самой программы приводит к завершению программы.

Пример:

```
IF A < 0 THEN
  Result := FALSE;
  RETURN;
END_IF;
```

9.4.8 Выход из программы

Оператор **EXIT** используется для раннего выхода из программы. Остальные выражения, программы стоящие после оператора **EXIT**, не выполняются.

Пример:

```
IF A > 100 THEN
  EXIT;
END_IF;
```

9.4.9 Комментарии

Комментарии используются для вставки поясняющих и других надписей в текст программы. Комментарии игнорируются при выполнении программы.

Комментарием является:

- текст, после комбинации символов **//** и до конца строки,
- текст, заключенный между символами **(*** и ***)** .

Пример:

```
// это однострочный комментарий

(*
 * это длинный многострочный
 * комментарий
 *)
```

9.4.10 Функции

Исходный текст функции должен начинаться с ключевого слова **FUNCTION**, за которым идет название функции, и заканчиваться ключевым словом **END_FUNCTION**.

Пример:

```
FUNCTION ST_Function
  // код функции
END_FUNCTION;
```

9.4.11 Функциональные блоки

Исходный текст функционального блока должен начинаться с ключевого слова **FUNCTION_BLOCK**, за которым идет название блока, и заканчиваться ключевым словом **END_FUNCTION_BLOCK**.

Пример:

```
FUNCTION_BLOCK ST_Block
  // код блока
END_FUNCTION_BLOCK;
```

9.4.12 Главная функция программы

Исходный текст главной функции программы должен начинаться с ключевого слова **PROGRAM**, за которым идет название программы, и заканчиваться ключевым словом **END_ PROGRAM**.

Пример:

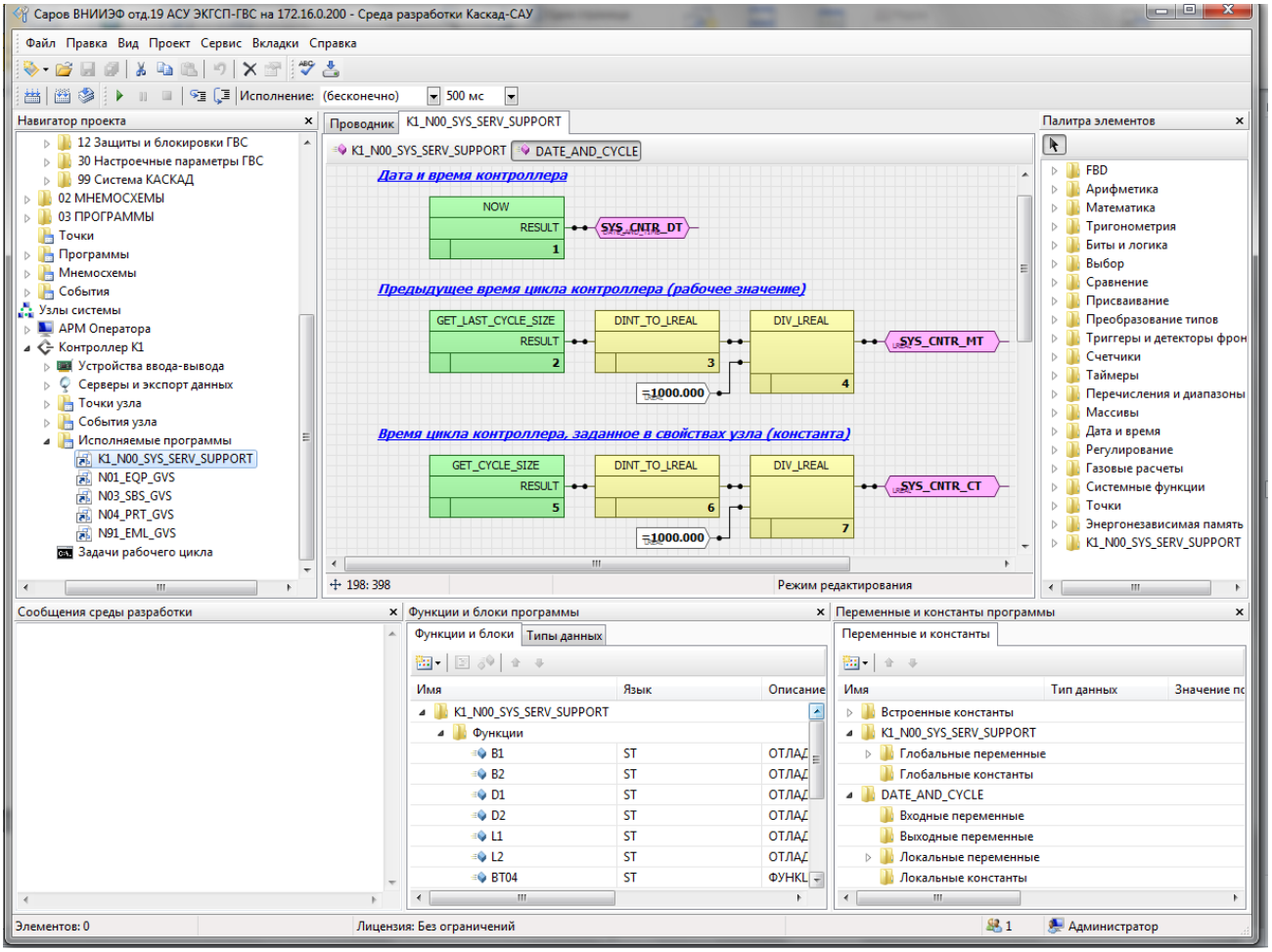
```
PROGRAM ST_Program
  // код программы
END_PROGRAM;
```

9.5 Язык программирования Function Block Diagram

Язык программирования Function Block Diagram (язык функциональных блоков, далее FBD) является графическим языком программирования.

9.5.1 Блоки, линии и соединительные точки

В языке FBD программа, функция или функциональный блок представляется в виде прямоугольных блоков, входы и выходы которых соединены линями.



Каждый блок представляет одну функцию или функциональный блок. Входные переменные отображаются ножками с левого края блока (входы), выходные переменные отображаются ножками с правого края блока (выходы). Внутри прямоугольника блока отображаются название функции, названия входных и выходных переменных и порядковый номер исполнения в программе (см. п. 9.5.3).

Места соединения линий и входов и выходов блоков обозначаются жирными точками. Линии могут быть соединены между собой, места таких соединений также отображаются жирными точками.

Внимание! Место соединения линии с ножкой блока также обозначается жирной точкой. Если точка отсутствует, то эта линия не соединена с ножкой, что приведет к ошибке компиляции программы.

К входам и выходов блоков могут быть подсоединены переменные. Переменные отображаются в виде вытянутых прямоугольников со скошенными сторонами. Для записи в переменную используется ножка слева, для чтения значения переменной используется ножка справа.

К входам и выходов блоков могут быть подсоединены константы. Константы отображаются в виде вытянутых прямоугольников со скошенной справа стороной. Для чтения значения константы используется ножка справа.

9.5.2 Передача данных между блоками

Во время исполнения программы передача данных между блоками определяется соединительными линиями:

- Чтобы подать значение с выхода одного блока на вход другого блока, нужно соединить линией ножку выхода с ножкой входа.
- Чтобы записать значения с выхода одного блока в переменную, нужно соединить линией ножку выхода блока с левой ножкой переменной.
- Чтобы подать значение переменной на вход блока нужно соединить правую ножку переменной и ножку входа блока.
- Чтобы подать значение константы на вход блока нужно соединить правую ножку константы и ножку входа блока.

Запрещается:

- подавать на один вход блока несколько выходов или переменных или констант,
- соединять одной линией несколько выходов блока, константы или переменной,
- соединять одной линией выход переменной или константы с входом переменной.

*Примечание. Для записи значения одной переменной в другую или значения константы в переменную используйте блок **MOVE**.*

*Внимание! Если на вход блока не соединен с выходом другого блока, переменной или константой, то во время исполнения на него будет автоматически подано значение **0** (**FALSE**).*

9.5.3 Порядок исполнения блоков

Порядок исполнения блоков в программе может выбираться автоматически либо вручную.

Для изменения порядка вычисления щелкните правой кнопкой мышки на пустом месте программы и в открывшемся меню выберите команду **Порядок исполнения**:

- При автоматическом режиме выбора блоки исполняются слева направо сверху вниз.
- При ручном режиме выбора блоки исполняются в порядке, указанном в списке в диалоге *Порядок исполнения*.

За один рабочий цикл узла исполняются все блоки программы, при условии, что перед ними не выполняется блок RETURN (см. п. 9.4.7).

9.5.4 Обратная связь

Если выход блока соединен линией с входом другого блока, который исполняется ранее, то значение с выхода первого блока будет подано на вход второго на следующем цикле расчета программы.

9.5.5 Циклы

Для организации циклов в программе, функции или функциональном блоке используются блоки **LABEL** и **GOTO**.

Блоки, исполняемые между блоками **LABEL** и **GOTO** с одинаковым именем метки, будут исполняться, пока на вход блока **GOTO** подается значение **TRUE**.

*Внимание. Если вход блока **GOTO** ни с чем не соединен, то цикл выполняться не будет.*

9.5.6 Возврат из функции

Блок **RETURN** используется для раннего выхода из функции или функционального блока. Остальные блоки функции или функционального блока, исполняемые после блока **RETURN**, не выполняются.

Вызов блока **RETURN** в теле самой программы приводит к завершению программы.

Выход из функции или функционального блока выполняется, если на вход блока **RETURN** подается значение **TRUE**.

Внимание. Если вход блока RETURN ни с чем не соединен, то возврат будет выполнен принудительно.

9.5.7 Комментарии

Для вставки комментария в программу используется элемент **Текст**. Во время исполнения программы элемент **Текст** игнорируется.

9.6 Типы данных

9.6.1 Поддерживаемые типы данных

Каскад-САУ использует для хранения значений переменных программ набор типов данных из стандарта **МЭК 61131-3**. Поддерживаемые типы данных приведены в таблице ниже.

Тип данных точки	**Размер, байт**	**Диапазон значений**	**Описание**
BOOL	1	True (1), False (0)	Логический тип

INT	2	-32768 ... 32767	16-битное знаковое целое
DINT	4	-214783648 ... 2147483647	32-битное знаковое целое
REAL	4	1.4·10-45 ... 3.4·10+38	32-битное число с плавающей точкой, 7 значащих цифр
LREAL	8	5.0·10-324 ... 1.7·10+308	64-битное число с плавающей точкой, 15 значащих цифр
TIME	8	-365 д 00:00:00.000 ... 365 д 00:00:00.000	Интервал времени с точностью до миллисекунд
DATE_AND_TIME	8	31.12.0099 00:00:00.000 … 31.12.9999 00:00:00.000	Дата и время с точностью до миллисекунд

Примечание. Для значений с плавающей точкой в таблице указаны наименьшее и наибольшее значащие значения

9.6.2 Преобразование типов данных

В каскад-САУ может выполняться автоматическое или явное преобразование значений из одного типа данных в другой.

Явное преобразование типов данных выполняется в программах технологических алгоритмов с помощью функций преобразования.

Автоматическое преобразование типов данных выполняется в следующих случаях:

- при считывании значения устройства в точку и при записи значения точки в устройство,
- перед и после расчета программ технологических алгоритмов,
- при приеме и передаче данных в вышестоящие системы.

При считывании данных из устройства ввода производится преобразование значения из типа данных устройства в тип данных точки (подробнее см. п. 15.6). При записи значения точки в устройство вывода производится обратное преобразование значения из типа данных точки в тип данных устройства (подробнее см. п. 15.8).

Перед расчетом программ технологических алгоритмов значения точек преобразуются из типа данных точки в тип данных входных переменных программы. После расчета значения выходных переменных программы преобразуются в тип данных точек.

OPC-сервер Каскад-САУ преобразует значения точек в соответствующий тип VARIANT перед отправкой клиентам (подробнее см. п. 15.13).

9.6.3 Таблица преобразования типов данных

Преобразование значений из одного типа данных в другой производится только для совместимых типов. Для некоторых типов данных преобразование приводит к потере точности.

В таблице ниже приведены правила преобразования типов данных Каскад-САУ.

Исходный тип	**BOOL**	**INT**	**DINT**	**REAL**	**LREAL**	**TIME**	**DATE_AND_TIME**
BOOL	+	+	+	+	+	+	+
INT	*	+	+	+	+	+	+
DINT	*	*	+	+	+	+	+
REAL	*	*	*	+	+	+	+
LREAL	*	*	*	*	+	+	+
TIME	*	*	*	*	+	+	-
DATE_AND_TIME	*	*	*	*	*	-	+

Результат преобразования:

- **+** - типы совместимы, преобразование производится без потери точности.
- ***** - типы совместимы, преобразование производится с потерей точности.
- **-** - типы не совместимы, преобразование не возможно.

9.6.4 Таблица соответствия типов данных Каскад-САУ типам стандарта OPC

Ниже приведена таблица соответствия типов данных точек Каскад-САУ версии 4.0 типам данных Каскад-САУ версии 3.2, стандарта OPC (OLE Automation) и языков программирования C/C++ и Pascal (Delphi).

Каскад-САУ 4.0, МЭК 61131-3	**Каскад-САУ 3.2**	**OPC**	**C/C++**	**Pascal (Delphi)**
BOOL	BOOLEAN	VT_BOOL	char	Boolean
INT	-	VT_I2	short	Smallint

DINT	INTEGER	VT_I4	int, long	Integer, Longint
REAL	-	VT_R4	float	Single
LREAL	DOUBLE	VT_R8	double	Double
TIME	-	VT_R8	double	Double
DATE_AND_TIME	-	VT_DATE	double	TDateTime

Примечание 1. В стандарте OPC и языке Pascal (Delphi) отсутствует тип интервала времени. В таблице указан тип, эквивалентный по внутреннему представлению значения в Каскад-САУ.

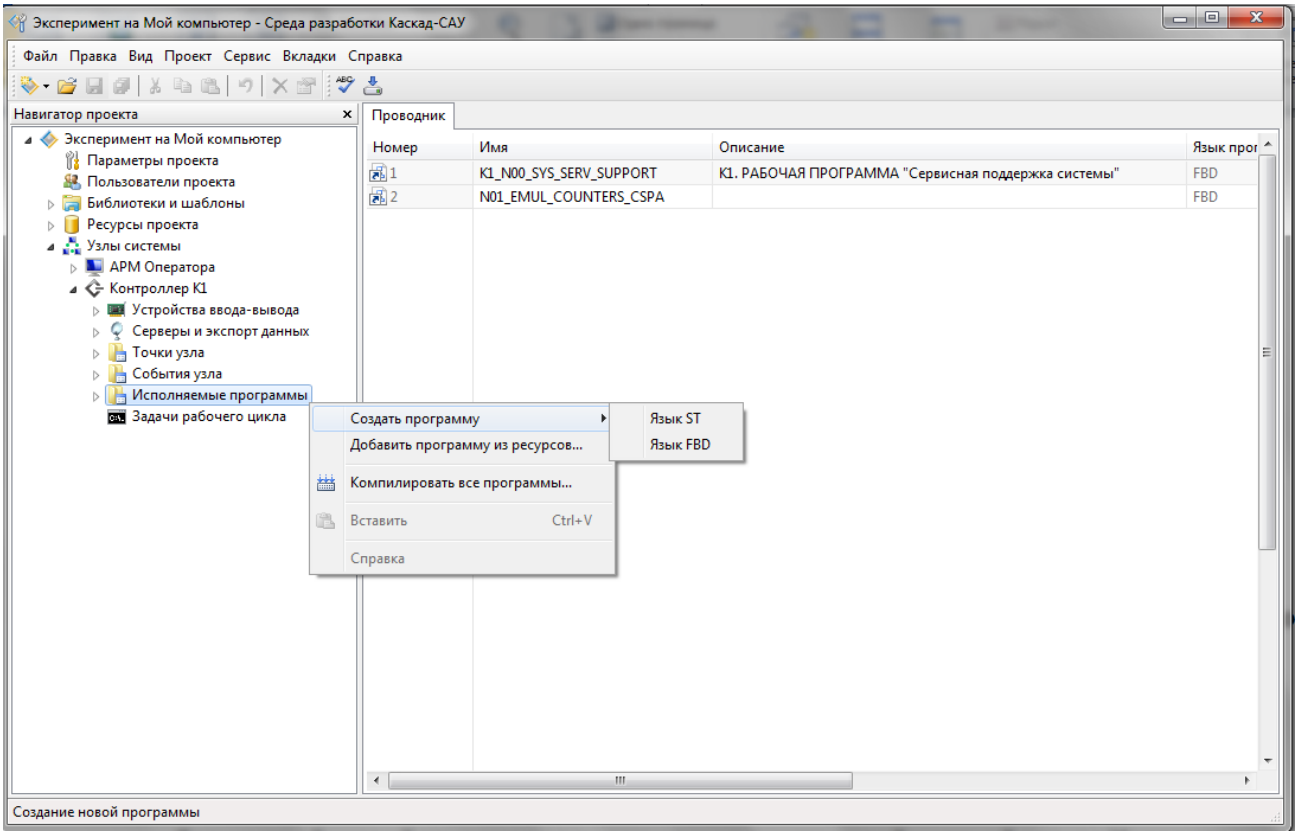
Примечание 2. В языке C отсутствуют тип даты и времени. В таблице указан тип, эквивалентный по внутреннему представлению значения в Каскад-САУ.

9.7 Создание новой программы

Для создания новой программы раскройте в дереве проекта папку узла, на котором она должна исполнятся, щелкните правой кнопкой на папке **Исполняемые программы**, выберите команду **Создать программу** и затем выберите язык программирования новой точки:

- **Язык ST** – язык программирования Structured Text.
- **Язык FBD** – язык программирования Function Block Diagram.

Сразу после создания откроется редактор исходного текста новой программы. Подробнее о редакторе см. п. 9.10.



9.8 Свойства программы

Для настройки свойств программы дважды щелкните левой кнопкой значок **Исполняемые программы** узла и перейдите в таблицу на вкладке **Проводник**. Выделите строку с программой и введите значения свойств.

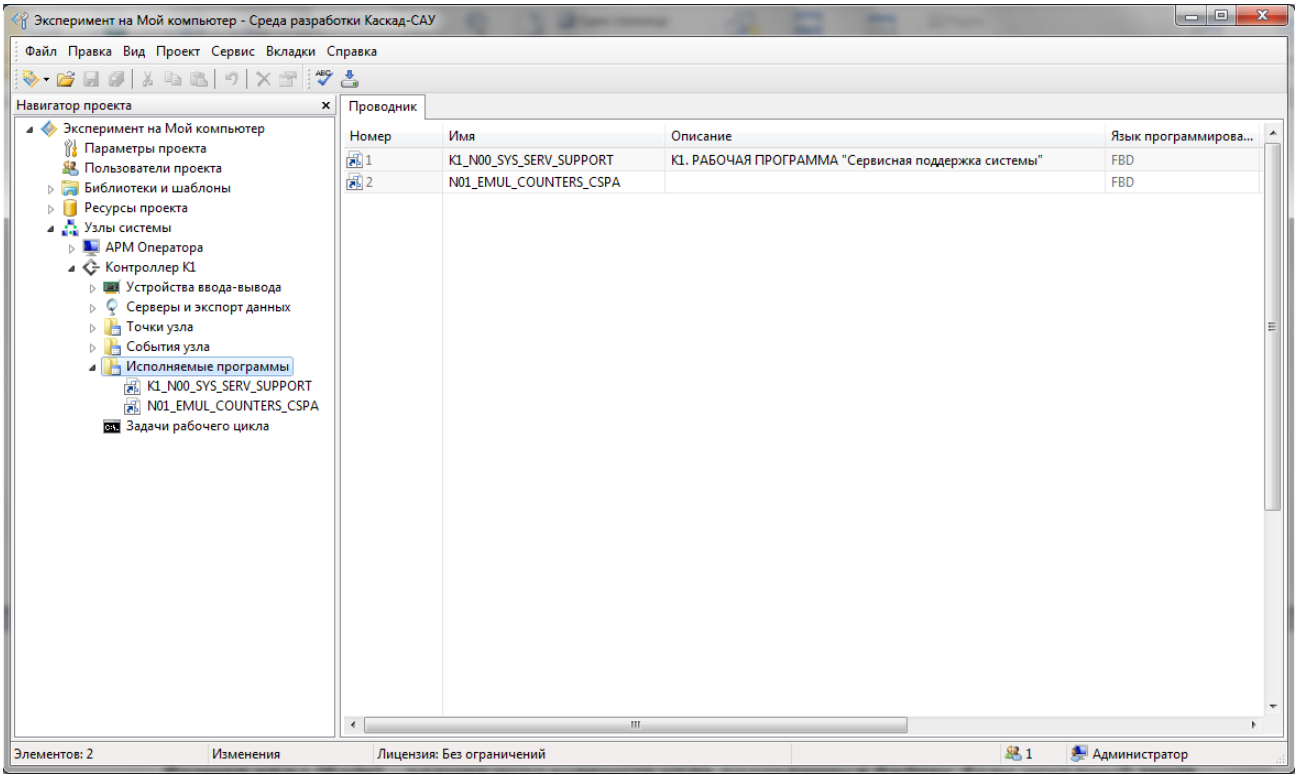
Номер - порядковый номер программы в таблице программ. Значение свойства выбирается автоматически при создании программы и не может быть изменено. При удалении программы ее номер может быть назначен следующей новой программе.

ИД - уникальный номер программы в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя программы, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя программы должно быть уникальным в проекте. Регистр символов имеет значение.

Описание - описание программы, не более 127 символов.

Язык программирования - язык программирования, на котором написана программа. Язык программирования выбирается при создании программы и не может быть изменен.



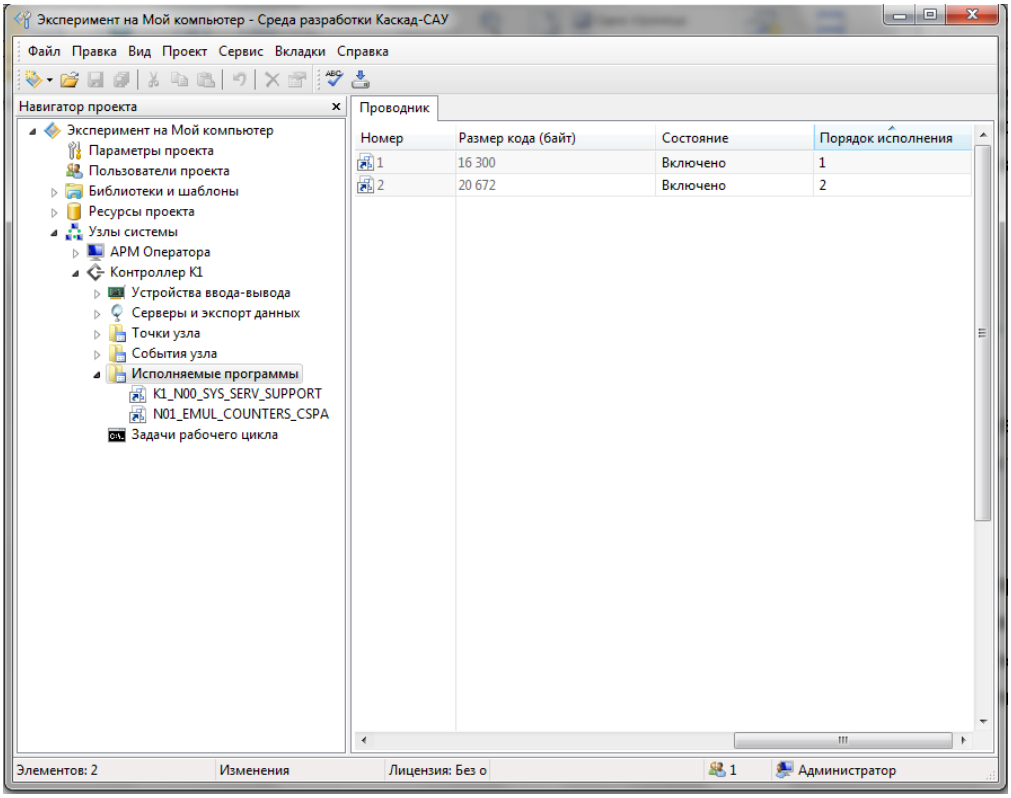
Размер кода (байт) - размер исполняемого кода программы в байтах. Если исходный текст программы изменен, но программа еще не компилирована, то в поле пишется надпись "Требуется компиляция". Свойство только для чтения.

Контрольная сумма кода - контрольная сумма исполняемого кода программы. Вычисляется по алгоритму CRC32. Используется для контроля изменения исполняемого кода на узле. Если исходный текст программы изменен, но программа еще не компилирована, то в поле пишется надпись "Требуется компиляция". Свойство только для чтения.

Состояние - состояние исполнения программы:

- **Отключено** - программа загружается на узел, но не исполняется.
- **Включено** - программа исполняется на узле.

Порядок исполнения - порядковый номер исполнения программы на узле. Используется для ручной настройки порядка исполнения нескольких программ на одном узле.



Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства, исходный текст и исполняемый код программы. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства, исходный текст и исполняемый код программы. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

9.9 Назначение программы на исполнение на узле

На одном узле может исполняться более одной программы.

Чтобы программа начала исполнятся на узле, перетащите ее значок в папку **Исполняемые программы** этого узла. Назначенные на исполнение программы отображается в дереве проекта в виде ярлыков.

*Примечание. Одна программа может исполняться одновременно на нескольких узлах. В этом случае у каждого узла в папке **Исполняемые программы** будет отображаться ярлык этой программы.*

9.10 Редактирование программы

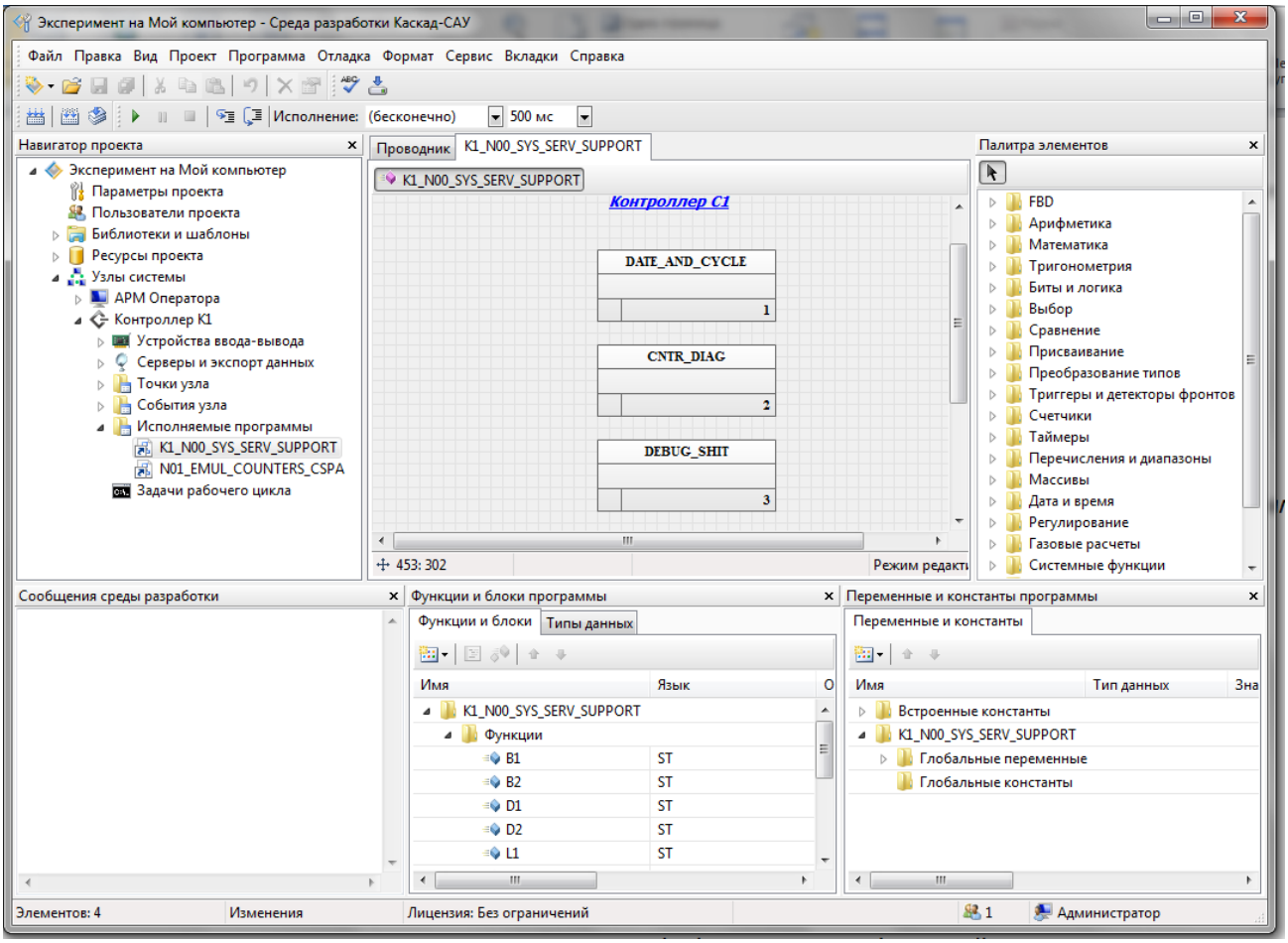
9.10.1 Открытие редактора программы

Чтобы начать редактирование программы дважды щелкните на ней в дереве проекта или таблице на вкладке **Проводник**.

Редактор программы откроется на отдельной вкладке программы. В зависимости от используемого языка программирования (см. п. 9.2.6) это может быть:

- текстовый редактор для программы на языке Structured Text,
- редактор графической схемы для программы на языке Function Block Diagram.

Примечание. По умолчанию в редакторе открывается текст главной функции программы (см. п. 9.2.1).



9.10.2 Редактор программ на языке Structured Text

Редактор программ на языке Structured Text - обычный текстовый редактор.

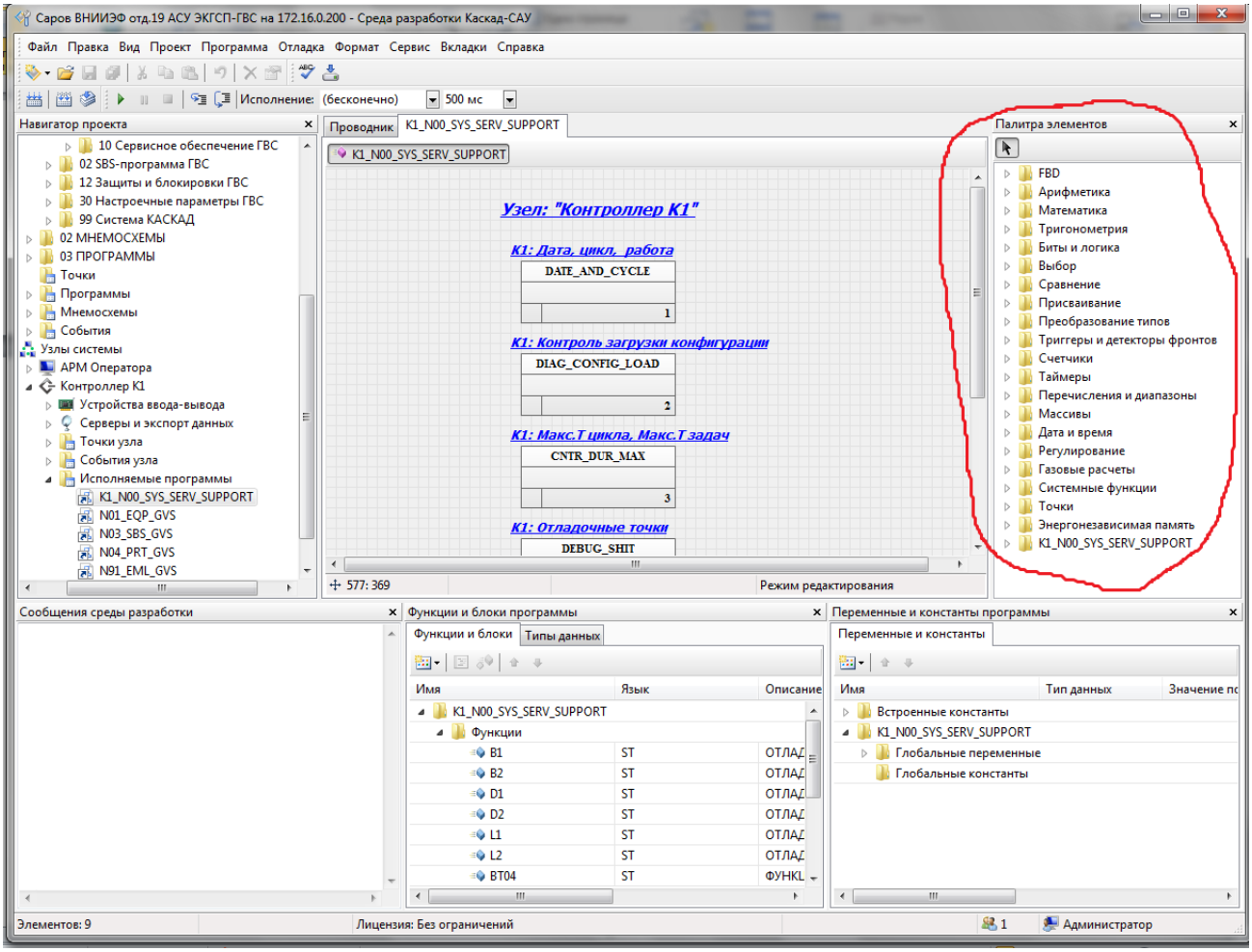
Для облегчения работы в редакторе поддерживается подсветка синтаксиса выражений языка.

9.10.3 Редактор программ на языке Function Block Diagram

Редактор программ на языке Function Block Diagram - графический редактор.

Создание и перетаскивание блоков, рисование линий схемы программы в редакторе выполняется с помощью мышки.

Чтобы вставить в программу стандартную функцию или функциональный блок откройте панель **Палитра элементов** и перетащите нужную функцию на схему.



*Примечание. Для открытия закрытой панели выберите в меню **Вид** команду **Другие панели**, затем **Палитра элементов**.*

Чтобы вставить в программу пользовательскую функцию (см. п. 9.10.6) или функциональный блок (см. п. 9.10.9) перетащите ее из панели **Функции и блоки программы** на схему.

Чтобы вставить в программу переменную или константу (см. п. 9.10.4) перетащите ее из панели **Переменные и константы программы** на схему.

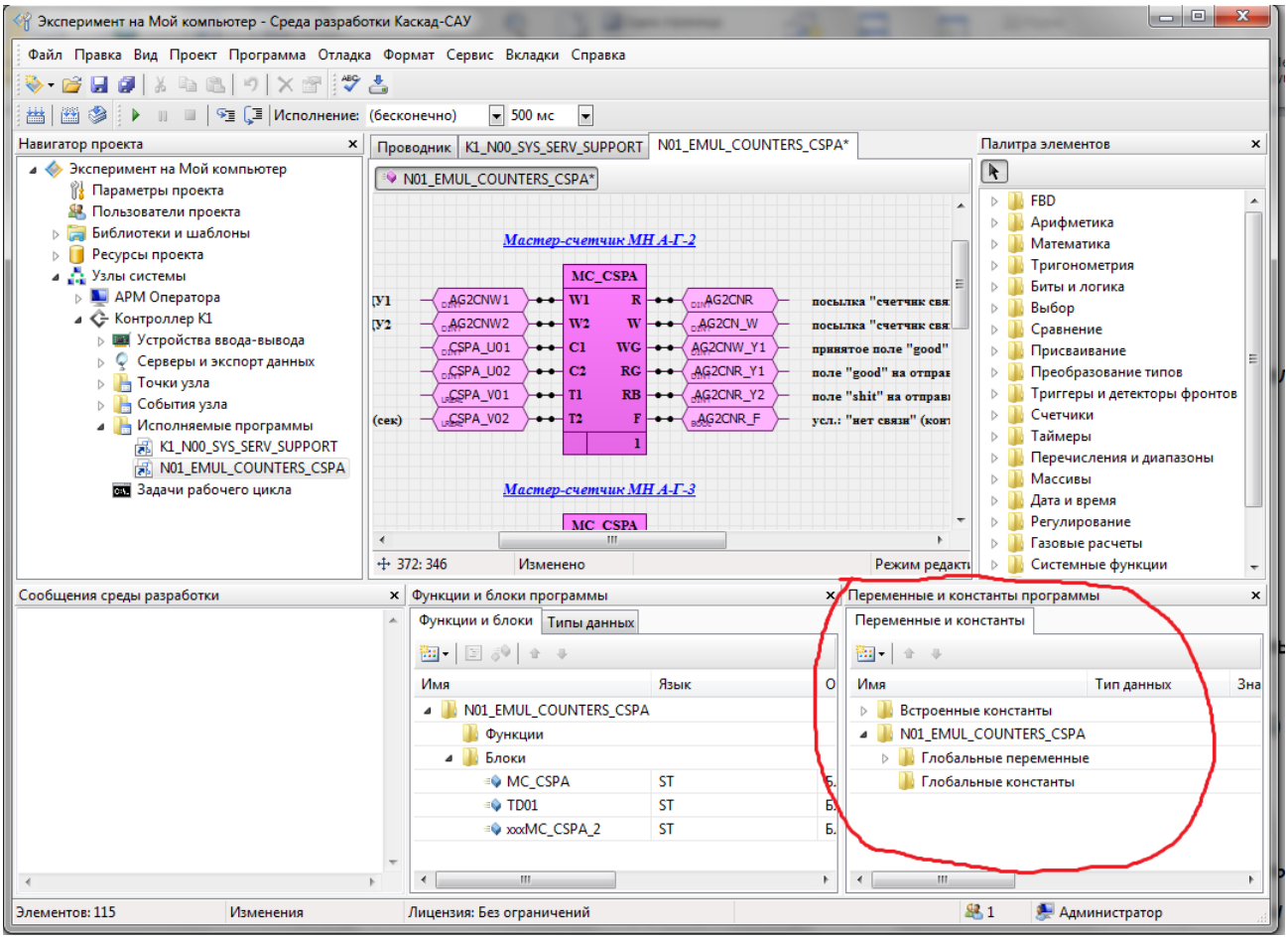
Чтобы начать нарисовать линию выберите в палитре элементов в папке *FBD* значок **Линия соединения**. Щелкните левой кнопкой в том месте схемы, где необходимо начать линию.

Щелкните там, где необходимо закончить первый сегмент линии, затем второй сегмент и так далее. В последней точке линии щелкните дважды. Чтобы рисовать горизонтальные и вертикальные сегменты линии нажмите и удерживайте клавишу SHIFT.

Чтобы быстро соединить линией две ножки входа и выхода функций, блоков или переменных щелкните на кончике первой ножки для начала рисования линии, затем щелкните на кончике второй ножки для завершения линии.

9.10.4 Объявление переменных

Список переменных программы отображается на панели **Переменные и константы**. Панель открывается автоматически при открытии редактора.



*Примечание. Для открытия закрытой панели выберите в меню **Вид** команду **Другие панели**, затем **Переменные и константы**.*

Для добавления новой переменной щелкните правой кнопкой мышки на папке **Глобальные переменные** *либо* **Локальные переменные**, в открывшемся меню выберите команду **Добавить переменную**. После объявления переменной укажите ее тип данных (см. п. 9.6) и значение по умолчанию.

*Примечание. **Глобальные** переменные - переменные, к которым можно обращаться в любом месте программы, функции или функциональном блоке. К **локальным** переменным можно обращаться только из той функции или функционального блока, в котором они объявлены.*

9.10.5 Привязка переменных программы к точкам проекта

Для работы программе, как правило, нужны значения, считанные из устройств. Однако программа не имеет прямого доступа данным устройств. Вместо этого она использует значения, записанные задачей ввода в точки в памяти узла.

Программа может получить и изменить значение точки двумя способами:

- Связать глобальные переменные с точками. Перед исполнением программы значения из точек автоматически копируются в глобальные переменные, по окончании исполнения значения из переменных копируются в точки.
- Считывать и записывать вручную с помощью функций **POINT_XXX** (см. п.).

Примечание. Связывание переменных функциональных блоков с точками признано устаревшим и не рекомендуется к использованию. В следующих версиях Каскад-САУ такое связывание не будет поддерживаться и может привести к ошибке компиляции программы.

Связывание глобальных переменных с переменных выполняется в панели **Переменные и константы**.

Примечание. Связывание переменных с точками в Каскад-САУ является аналогом Directly represented variables стандарта IEC 61131-3, то есть аналогом объявления переменной, связанной с адресом в памяти контроллера с использованием символа "%". Например, %IW215, %QB7 и прочих.

Для связывания переменной с точкой выделите в списке переменную и введите в поле **Точка** название точки либо дважды щелкните по полю и выберите точку в диалоге **Выбор точки**.

В **Поле точки** выберите поле точки. Для работы в программе доступны только числовые поля точки.

В поле **Доступ** выберите тип доступа к полю точки:

- **Чтение** - значение выбранного поля будет записано в переменную в начале исполнения программы.
- **Запись** - значение переменной будет записано в поле точки в конце исполнения программы.

- **Чтение+запись** - в начале исполнения программы значение выбранного поля будет записано в переменную, в конце исполнение значение из переменной будет записано обратно в поле точки.

*Внимание! При включении режима имитации или режима калибровки у точки (биты статуса **ИМИТ** и **КЛБР**, см. п. 7.7.3) программа автоматически отключается от записи значения в точку. То есть при завершении расчета значения из переменных не будут записаны в точку, вызов функции **POINT_SET_VALUE** (см. п.) не изменит значения точки.*

Примечание. Во время чтения значения поля в переменную и записи значения переменной в поле выполняется неявное преобразование типов данных значения (см. п. 9.6.2).

9.10.6 Объявление констант

Список констант программы отображается на панели **Переменные и константы**.

Для добавления новой константы щелкните правой кнопкой мышки на папке **Глобальные константы** *либо* **Локальные константы**, в открывшемся меню выберите команду **Добавить константу**. После объявления константы укажите ее тип данных (см. п. 9.6) и значение по умолчанию.

*Примечание. **Глобальные** константы - константы, к которым можно обращаться в любом месте программы, функции или функциональном блоке. К **локальным** константам можно обращаться только из той функции или функционального блока, в котором они объявлены.*

9.10.7 Встроенные константы

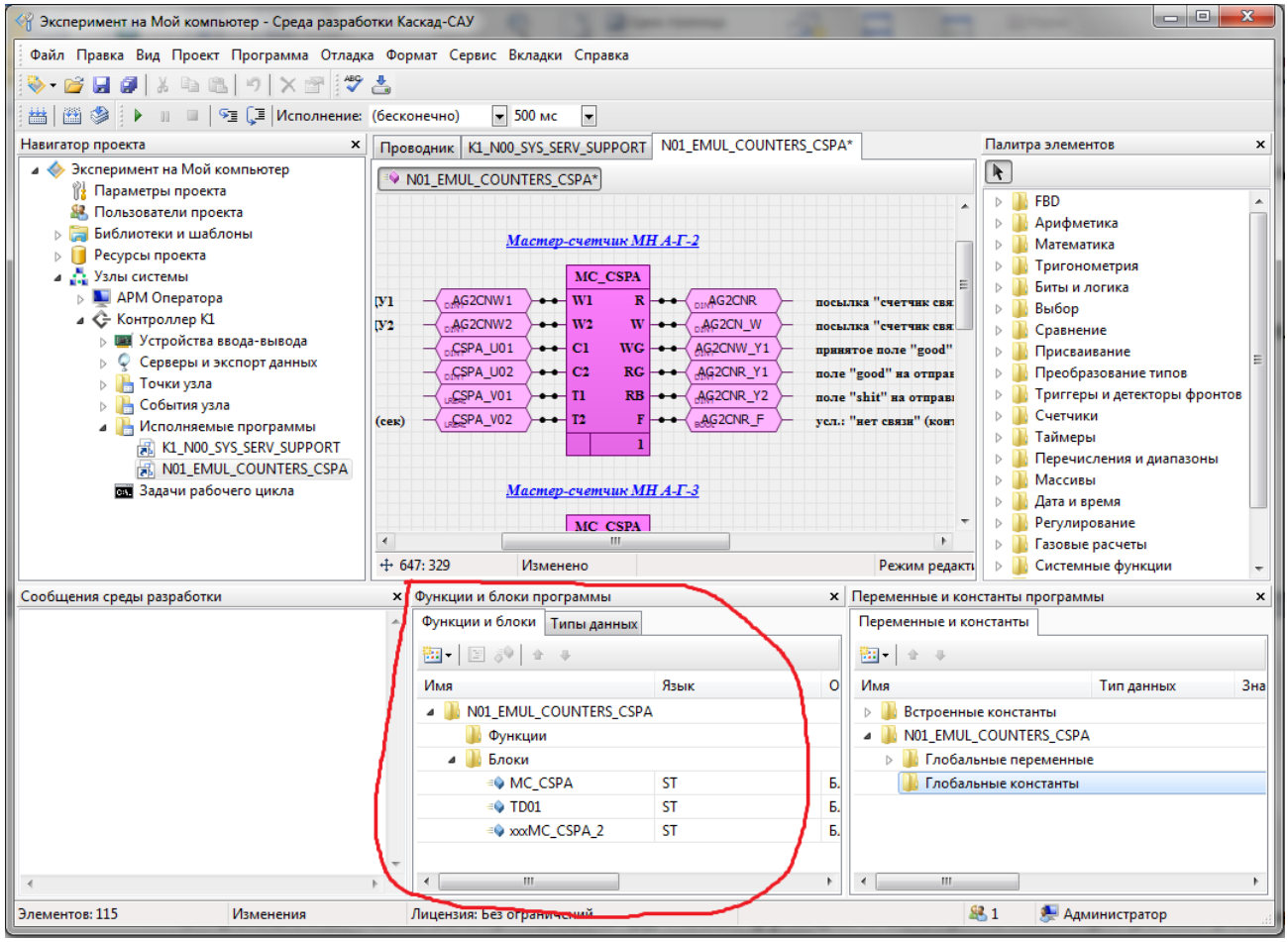
Встроенные константы - предварительно объявленные константы, которые можно использовать в программе:

- маски битов статуса точки (см. п. 7.7),
- номера полей точки для функции **POINT_GET_FIELD** (см. п.),
- номера дней недели,
- константы для работы с датой и временем,
- число π.

Список встроенных констант отображается на панели **Переменные и константы программы** в папке **Встроенные константы**.

9.10.8 Функции

Список функций программы отображается на панели **Функции и блоки программы**. Панель открывается автоматически при открытии редактора.



Примечание. Для открытия закрытой панели выберите в меню ***Вид*** *команду* ***Другие панели****, затем* ***Функции и блоки программы****.*

Для добавления новой функции щелкните правой кнопкой мышки на папке **Функции**, в открывшемся меню выберите команду **Новая функция** и выберите язык программирования для написания ее кода.

Примечание. Язык программирования, на котором написана функция, не обязательно должен совпадать с языком, на котором написана программа, другие функции и функциональные блоки.

9.10.9 Функциональные блоки

Функциональный блок - это функция, которая сохраняет свое состояние от вызова к вызову.

Список функциональных блоков программы отображается на панели **Функции и блоки программы**. Для добавления нового функционального блока щелкните правой кнопкой мышки на папке **Блоки**, в открывшемся меню выберите команду **Новый блок** и выберите язык программирования для написания его кода.

Примечание. Язык программирования, на котором написан функциональный блок, не обязательно должен совпадать с языком, на котором написана программа, другие функциональные блоки и функции.

9.10.10 Стандартные функции и блоки

Список стандартных функций, сгруппированный по категориям, отображается на панели **Палитра элементов**. Для добавления в программы стандартной функции дважды щелкните на ней в палитре либо перетащите мышкой из палитры в текст программы.

*Примечание. Для открытия закрытой панели выберите в меню **Вид** команду **Другие панели**, затем **Палитра элементов**.*

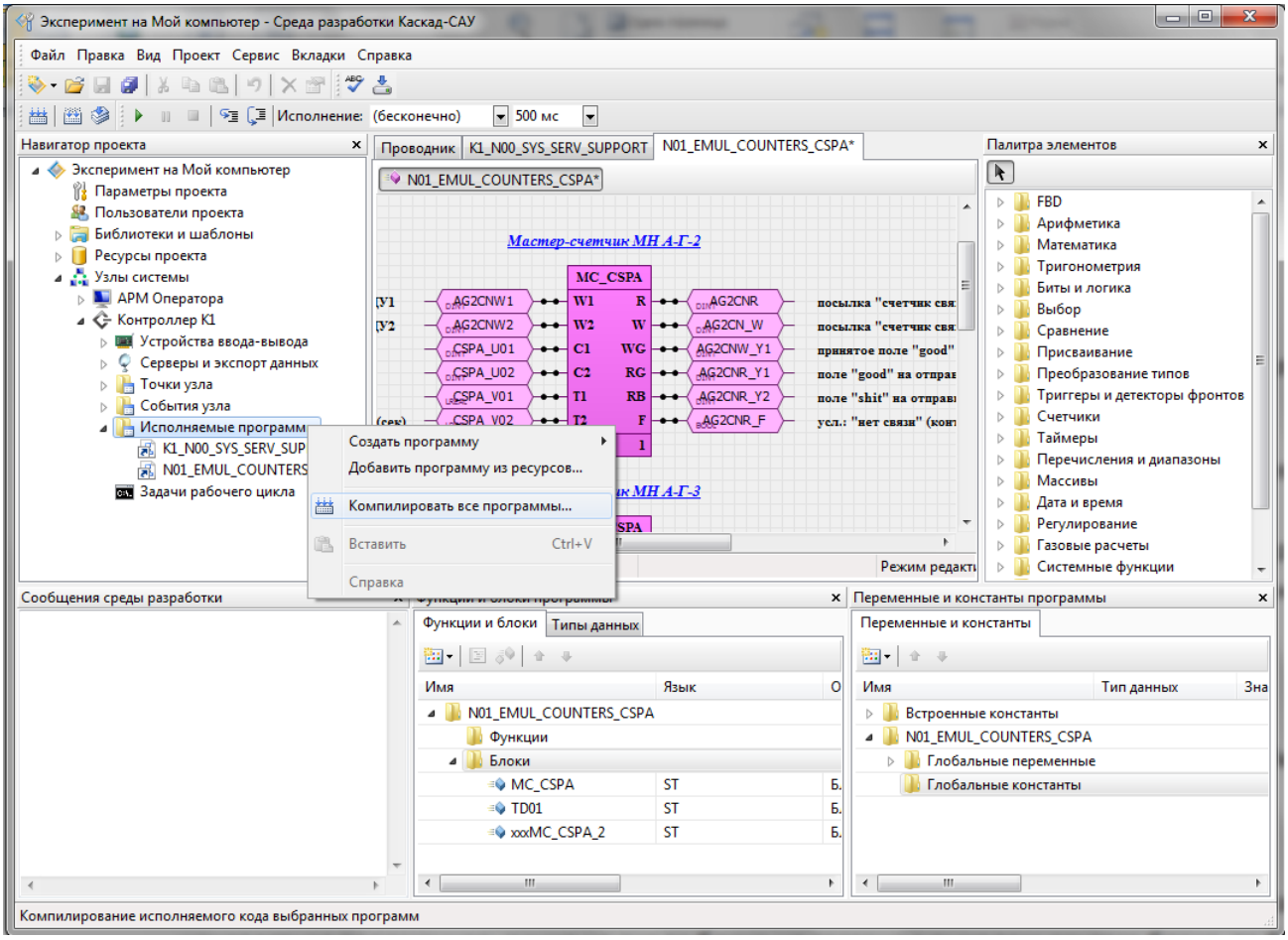
*Внимание! В Каскад-САУ не поддерживается перегрузка функций. Вместо этого стандартные функции представлены в нескольких вариантах для работы с разными типами данных. Такие функции имеют в имени суффикс типа данных, например, **ADD_DINT** и **ADD_LREAL**.*

9.11 Компилирование программы

После изменения программы ее следовать компилировать - получить байт-код для исполнения на узле. Исполнение байт-кода программы на узле выполняет задача *виртуальная машина IEC 61131-3* (см. п. 15.8).

Внимание! После любого изменения программы ее следует компилировать заново, чтобы получить новый исполняемый байт-код.

Для компилирования программы щелкните правой кнопкой мышки на программе в дереве проекта или таблице на вкладке **Проводник** и в меню **Проект** выберите команду **Компилировать программу**. Для компиляции сразу нескольких программ щелкните в дереве проектов правой кнопкой мышки на папке **Исполняемые программы** и в открывшемся окне **Компилировать все программы** отметьте флажками программы для компиляции.



Примечание. Исходный текст программы не хранится на узле. При утере или повреждении проекта текст программы не может быть восстановлен из исполняемого кода программы на узле.

Если в ходе компиляции в программе обнаружены ошибки, например, ошибка синтаксиса, неизвестная переменная, линия на входе блока не связана с выходом другого блока, то будет автоматически открыта панель **Сообщения среды разработки** со списком найденных ошибок. Чтобы переместиться к месту ошибке в программе дважды щелкните на строку с текстом ошибки в панели сообщений.

Помимо ошибок в ходе компиляции могут быть выданы предупреждения и подсказки по исходному тексту программы. Предупреждения и подсказки не являются ошибками с точки зрения языка программирования, но сообщают о подозрительных местах в программе, которые могут привести к ошибкам в процессе ее исполнения.

Внимание. Настоятельно рекомендуется устранить все предупреждения компиляции программы.

9.12 Особенности разработки программ в Каскад-САУ

9.12.1 Использование таймеров в программах

Таймер - переменная программы типа **TIME**, которая используется в программах для отсчета времени.

Отсчет времени с помощью таймеров можно реализовать несколькими способами:

1. Использовать стандартные функции TP, TON и TOF. Таймер включается подачей на вход управляющего сигнала, по окончании работы таймера на выходе будет сформирован управляющий импульс.

2. Использовать функции ручного управления таймером TIMER_START, TIMER_STOP, TIMER_STARTED. Запуск таймера делается функцией TIMER_START. Текущее значение таймера хранится в таймерной переменной и обновляется автоматически. Завершение работы таймера проверяется вручную, сравнивая значение переменной со значением таймаута.

3. Использовать отсчет таймера по времени рабочего цикла. Значение таймера увеличивается вручную каждый цикл расчета программы на величину времени рабочего цикла, полученного с помощью функции **GET_LAST_CYCLE_SIZE**.

4. Использовать функции для работы со временем. Время начала работы таймера фиксируется с помощью функции **NOW**. Значение таймера вычисляется вручную каждый такт как разница между временем начала работы таймера и текущим временем.

Значения всех таймеров, включенных функциями **TP**, **TON, TOF** и **TIMER_START** обновляются один раз за рабочий цикл узла перед расчетом программы после загрузки значений точек в переменные. В течение расчета программы значение таймера не меняется, даже если программа считается долго. Поэтому в программе нельзя включить таймер и затем сделать цикл и ждать в нем, когда таймер сработает.

Минимальное время импульса, которое можно сделать с помощью таймера, равно размеру цикла узла. Если требуется сделать импульс длительностью меньше рабочего цикла узла, то следует использовать внешние устройства, например блок ШИМ.

*Примечание. Поскольку цикл узла может "плавать" +/- 1 мс, таймер, включенный функциями **TP, TON, TOF и TIMER_START,** может увеличиться, например, не на 500 мс, а на 499 мс. Поэтому таймаут в 500 мс может выдать срабатывание таймера на 1 цикл позже, чем надо. Чтобы этого не было следует в функциях указывать таймаут 1-2 мс меньше требуемого.*

Таймерную переменную можно связывать с точкой, как на чтение, так и на запись. В начале выполнения программы в таймер загрузится значение из точки, затем таймер увеличится, по

окончании выполнения значение таймера запишется в точку и так далее по кругу. Таким образом, значение таймера может быть передано для отображения на АРМ оператора, в вышестоящие системы и сохранено в энергонезависимую память.

Примечание. При записи значения таймера в точку не сохраняется признак включенного таймера. Если точка таймера сохраняется в энергонезависимую память, то при перезагрузке узел восстановит только значение таймера, сам таймер не будет запущен. Аналогично на другой узел можно передать в точке только значение таймера. Для сохранения состояния таймера или передачи его на другой узел нужно использовать отдельную точку.

Запуск таймера функцией **TP**, **TON, TOF** и **TIMER_START** сбрасывает значение таймера в 0. Но можно поменять начальное значение таймера вручную присвоением нужного значения таймерной переменной, в том числе и для включенных таймеров.

Внимание! Не поддерживается передача признака включения таймера параметром функции. Функция может включить таймер, переданный во входной переменной, но при выходе из функции признак включения будет потерян, таймер работать не будет. Поэтому таймерами могут быть только глобальные переменные и локальные переменные блоков.

9.12.2 Обработка ошибок исполнения программ

В ходе исполнения программы могут возникать ошибки. Различают критические и некритические ошибки исполнения.

При возникновении критической ошибки программа останавливается, выводится сообщение об ошибке в журнал отладки узла.

К критическим ошибкам относятся:

- ошибки в байт-коде (ошибки компилятора),
- неизвестная команда (разные версии компилятора и виртуальной машины),
- зацикливание программы (см. п. 15.8).

Примечание. Наличие ошибок в работе программ узла может быть диагностировано с помощью параметров обобщенной диагностики виртуального устройства ввода ***Монитор ресурсов****.*

При возникновении некритической ошибки программа продолжает выполняться, никаких сообщений в журнал не выводится.

К некритическим ошибкам относятся:

- математические ошибки, переполнение или потеря точности,

- деление на 0,
- недопустимый номер точки в функциях по работе с точками,
- ошибка чтения или записи в энергонезависимую память,
- выход за границу массива,
- недопустимое значение параметра функции.

Некритические ошибки определяются по коду ошибки исполнения. Код ошибки автоматически сбрасывается перед каждым исполнением программы. Для получения кода ошибки используется функция **GET_LAST_ERROR**. Возможные коды ошибок указаны в описании функций и блоков.

Результат выполнения функции, вызвавшей некритическую ошибку, зависит от вида ошибки:

- Математические ошибки, переполнение, потеря точности - результат не определен.
- Переполнение во время приведения типа - результат не определен.
- Деление на 0 - результат 0.
- Отрицательный параметр сдвига функций **SHL**, **SHR**, **ROL**, **ROR** - результат 0.
- Недопустимый номер или поле точки функций **POINT_XXX** и **NVRAM_XXX** - результат 0. При записи в точку значение и статус не меняется.
- Выход за границу массива - результат 0. При записи значение элемента массива не меняется.

9.12.3 Особенности программирования для распределенных систем

В распределенных системах ввод-вывод и обработка данных может быть разделена на разные узлы.

Например, можно разнести логику управления объектами по разным контроллерам и учитывать в каждом состояние соседнего. Либо вынести логику управления на выделенный сервер, а ввод-вывод выполнять с помощью множества небольших контроллеров.

Разделение логики возможно благодаря автоматической синхронизации данных между узлами одного проекта. Программа на одном узле может использовать для вычислений значения входных точек других узлов, считанные с устройств ввода. Значения точек, которые изменит программа, будут автоматически переданы на другие узлы и записаны в устройства вывода.

Примечание. Точки, которые меняет программа, должны принадлежать узлу, на котором эта программа исполняется. В противном случае эти точки не будут переданы на другие узлы.

При разделении ввода-вывода и обработки данных на разные узлы следует учитывать следующие особенности:

- Между возникновением сигнала и выдачей команды управления на исполнительное устройство может быть задержка в 2 цикла узла (или более). Значение входной точки первого узла будет передано на второй узел только в конце рабочего цикла (1-й цикл). Значение точки, которую изменит программа на втором узле, будет передана обратно на первый узел в конце цикла (2-й цикла). Итого задержка 2 цикла.
- Из-за медленной линии связи на втором узле возможна потеря данных от первого узла, что может сказаться на работе программы (см. п. 16.4).
- Для больших систем требуется большое количество точек, что сильно увеличит расход оперативной памяти на узлах и нагрузку на сеть.

9.12.4 Особенности реализации согласно приложению D из МЭК 61131-3

Стандарт **МЭК 61131-3** реализацию многих параметров языков программирования отдает на усмотрение производителя. Эти параметры перечислены в приложении D стандарта.

Особенности реализации языков программирования стандарта **МЭК 61131-3** в Каскад-САУ приведены в таблице ниже.

П.п.	Особенность	Реализация
	Указание типа после имени функции в языке ST.	В языке ST указание типа после имени функции не поддерживается. Тип функции определяется типом встроенной переменной Result. В теле функции для возврата значения следует использовать переменную Result вместо названия функции.
2.1.2	Максимальная длина идентификатора.	Максимальная длина названия функции/блока 79 символов. Максимальная длина названия переменной 35 символов. Максимальная длина названия константы 35 символов. Максимальная длина названия типа 63 символа. Максимальная длина метки 35 символа.
2.1.5	Максимальная длина комментария.	Длина комментария не ограничена. Поддерживаются однострочные комментарии //.
2.1.6	Синтаксис и семантика прагма.	Прагма ({pragma}) не поддерживаются.

2.2.1	Синтаксис и семантика числовых констант.	Не поддерживается указание типа при определении константы, например, DINT#5, BOOL#TRUE и т.п.
2.2.2	Синтаксис и семантика строковых констант.	Строковые константы и выражения не поддерживаются.
2.3.1	Диапазон и точность значений типов TIME, DATE_AND_TIME.	Диапазон значений типа TIME: -365000-00:00:00.000 до 365000-00:00:00.000. Диапазон значений типа DT: 31.12.0099 00:00:00.000 до 31.12.9999 00:00:00.000. Точность измерения типов TIME и DT – 1 мс. Типы DATE, TIME_OF_DAY не поддерживаются.
2.3.3.1	Поддержка и ограничения перечисляемых типов и массивов.	Поддерживаются производные типы от базовых типов. Уровень вложенности производных типов не ограничен. Максимальное количество элементов перечисляемого типа не ограничено. Максимальное количество элементов в массиве не ограничено. Максимальное количество типов-массивов не ограничено. Структурные типы не поддерживаются.
2.3.3.2	Поддержка и ограничения строковых типов.	Строковые типы не поддерживаются.
2.4.1.1	Определение переменных.	Блоки определений переменных VAR/END_VAR не поддерживаются. Для определения переменных используется таблица в отдельном окне редактора. Адресная нотация переменных %I/%Q/%M не поддерживается. Привязка переменных к точкам делается в таблице переменных.
2.4.2	Инициализация переменных.	Глобальные переменные программы инициализируются начальными значениями один раз перед самым первым исполнением программы. Выходные и локальные переменные функции сбрасываются в значение по умолчанию перед каждым вызовом функции. Выходные и локальные переменные блока сбрасываются в значение по умолчанию один раз вначале перед

		самым первым исполнением программы. Незаданные входные переменные функции/блока (на FBD не привязана ножка, в ST параметр пропущен при вызове с использованием оператора :=) сбрасываются в 0.
2.4.3	Максимальное количество переменных.	Количество глобальных/локальных переменных не ограничено. Ключевое слово RETAIN для включения сохранения значения в энергонезависимую память не поддерживается. Вместо него следует включить сохранение значения точки в энергонезависимую память, либо записывать/читать значения из энергонезависимой памяти вручную с помощью функций NVRAM_XXX.
2.5	Ограничения на количество вызовов функции/блоков.	Не более 16 вложенных вызовов функций/блоков. Общее количество вызовов функций/блоков программы ограничено количеством команд виртуальной машины, исполненных в этой программе за один рабочий цикл (по умолчанию 1000000, задается параметром **CmdLimit** задачи *виртуальной машины IEC 61131-3* (***vmiec61131)***, см. п.).
2.5.1.2	Значение выходных переменных функций при ENO равном FALSE.	Управление исполнением функций (вход EN и выход ENO) не поддерживается.
2.5.1.3	Максимальное количество функций.	Количество функций в программе не ограничено.
2.5.1.4	Таблица перекрытых функций.	Перекрытие функций (автоматический выбор функции подходящего типа по типу входных переменных) не поддерживается. Все функции строго типизированы. Например, вместо функции ADD используются функции ADD_INT, ADD_DINT, ADD_REAL и т.п.
2.5.1.5	Максимальное количество входов/выходов.	Не более 32 входных и не более 32 выходных параметров функции/блока. У функции SEL, MAX, MIN, ADD, MUL в языке ST не может быть более 2

		параметров.
2.5.1.5.1	Точность приведения типов.	Результат приведения типа не определен, если значение на входе выходит за диапазон значений выходного типа (ошибка исполнения 34).
2.5.1.5.2	Точность математических функций.	Результат функции SQRT(-1) не определен (ошибка исполнения 2, код 33). Результат функции LN(-1) не определен (ошибка исполнения 2, код 33). Результат функции LN(0) не определен (ошибка исполнения 2, код 34). Результат функции ASIN(-2) не определен (ошибка исполнения 2, код 33). Функция MUX возвращает 0/FALSE, если селектор выходит за границу. Функции SHL, SHR, ROL, ROR возвращает 0, если количество разрядов больше 15 (ошибка исполнения 2). Функции DIV, MOD возвращает 0 в случае деления на 0 (ошибка исполнения 3). Функции POINT_XXX и NVRAM возвращают 0 при неправильном номере точки (ошибка исполнения 4, 5, 6, 7). Функция ROUND использует банковское округление (к ближайшему четному числу).
2.5.1.5.6	Приведение типов даты и времени к другим типам.	Значение типа TIME и DT может быть приведено к значению типа LREAL и обратно.
2.5.2	Максимальное количество блоков.	Количество функциональных блоков программы не ограничено. Количество переменных типа блок не ограничено.
2.5.2.1	Вызов блоков.	Не поддерживается обращение к переменным блока через точечную нотацию в языке ST.
2.5.2.1 a)	Значение входных переменных при ENO равном FALSE.	Управление исполнением блоков (вход EN и выход ENO) не поддерживается.
2.5.2.3.3	Пределы значений PV блоков таймеров.	Максимальное и минимальное значение параметра PV блоков CTU, CTD зависит от типа блока INT или DINT.
2.5.2.3.4	Изменение параметра PT блоков таймеров при	Разрешается изменять значение входной переменной PT блоков TP, TON, TOF во время

	включенном таймере.	отсчета времени таймером для изменения длительности импульса.
2.5.3	Ограничение на размер программы.	Максимальный размер программы ограничен лимитом на суммарный размер байт-кода всех программ в проекте (см. п.).
2.6.X	Ограничения SFC.	Язык SFC не поддерживается.
2.7.1	Эффект от использования READ_WRITE доступа к выходным переменным блоков.	Разрешается читать и писать значения в выходные переменные блока.
2.7.2	Максимальное количество и интервал исполнения задач.	Понятие задач не поддерживается. Вместо задач используется последовательное исполнение программ. Каждая программа исполняется один раз за рабочий цикл узла. Размер рабочего цикла настраивается свойством **Рабочий цикл** узла (см. п. 6.3). Максимальное количество программ в одном цикле ограничено лимитом на количество программ в проекте.
3.3.1	Максимальная длина выражения в языке ST.	Максимальная длина выражения не ограничена.
3.3.2.3	Максимальное количество условия оператора CASE.	Количество условий оператора CASE не ограничено.
3.3.2.4	Значение переменной цикла FOR по завершении цикла.	Значение переменной цикла FOR при выходе из цикла равно MAX+1.
4.1.1	Ограничение топологии связей в FBD.	Ограничения на порядок следования блоков в FBD нет.
4.1.3	Порядок исполнения при наличии циклических связей FBD.	Порядок исполнения блоков FBD задается автоматически слева направо, сверху вниз либо назначается вручную.

10 Серверы и экспорт данных

10.1 Передача данных во внешние системы

Узлы могут передавать и принимать данные в/из узлов другого проекта, контроллеров и систем управления других производителей и прочих внешних систем сбора данных.

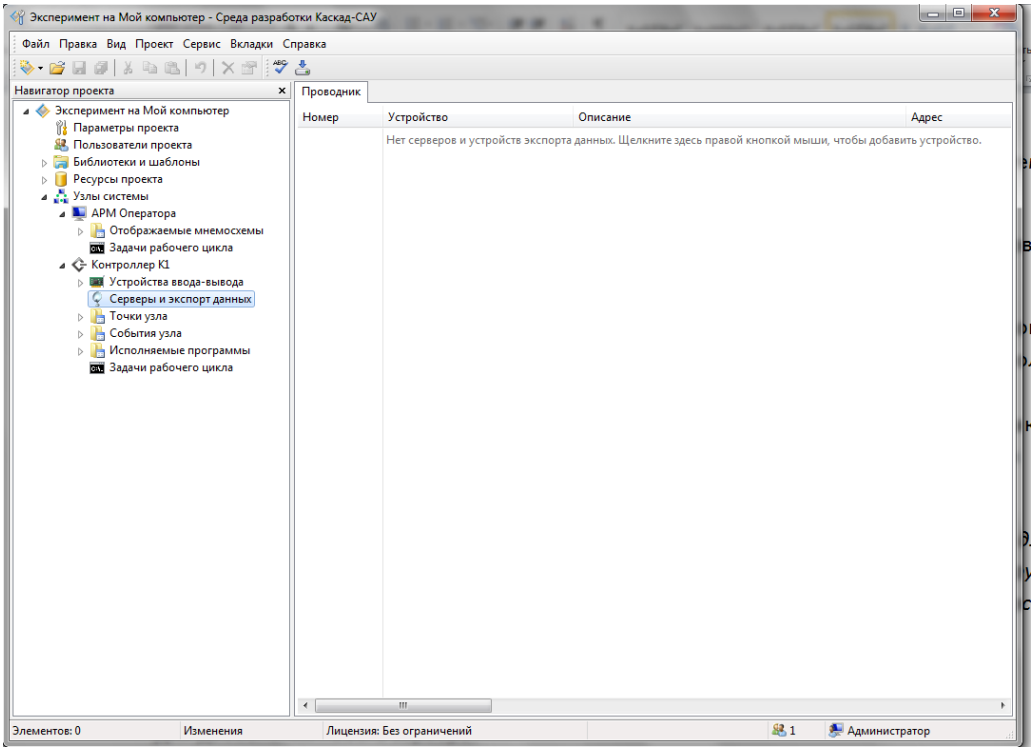
Как правило, соединение узла с внешними системами выполняются через последовательный порт RS-232/RS-485 или по сети Ethernet. Для передачи данных используется один из поддерживаемых протоколов передачи данных или интерфейс OPC DataAccess. Узел может передавать данные во внешние системы одновременно по нескольким протоколам.

При передаче данных узел является для других систем ведомым устройством, то есть внешние системы читают данные узла и записывают в него команды управления. Подробнее о работе серверов данных см. п. 15.13.

Примечание. Передавать и принимать данные внешних систем могут все узлы проекта. Например, в некоторых случаях для связи с внешней системой удобнее отказаться от узла выделенного сервера данных, и передавать их непосредственно с узла контроллера.

10.2 Конфигурация серверов данных

Для настройки конфигурации серверов и протоколов передачи данных во внешние системы используется папка **Серверы и экспорт данных** узла в дереве проекта.



Среда исполнения использует конфигурацию серверов для запуска и настройки серверов передачи данных в процессе работы.

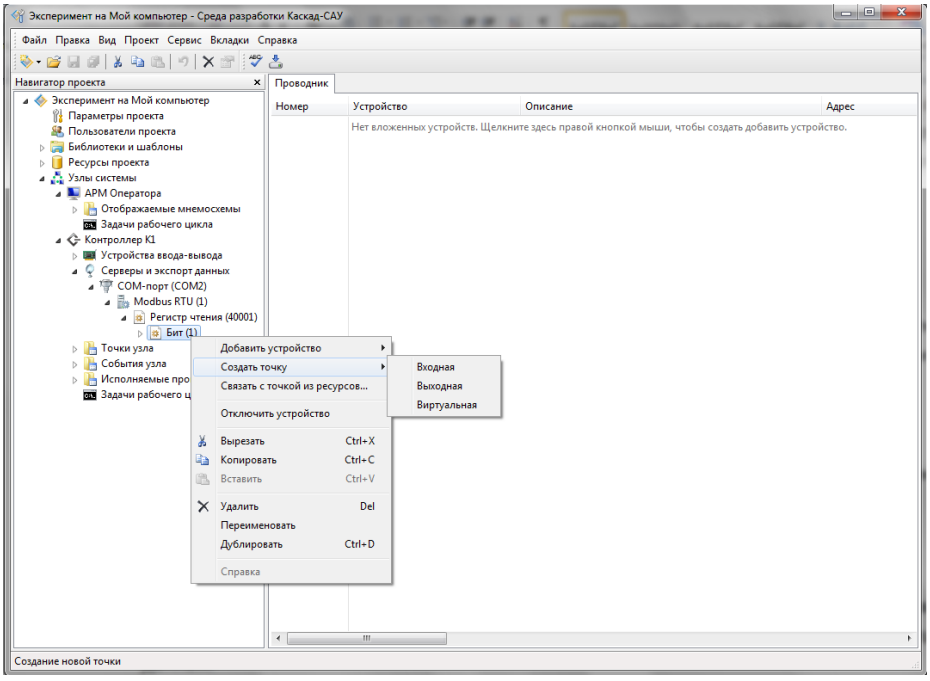
Конфигурация серверов данных отображается в папке в вид дерева в порядке физического подключения внешних систем к узлу. Например, если SCADA система опрашивает контроллер по протоколу *Modbus RTU*, то в дереве **Серверы и экспорт данных** порт *COM1* будет отображаться как устройство первого уровня, сервер *Modbus RTU* будет отображаться как дочернее устройство порта *COM1*, регистры протокола будут отображаться в дереве как дочерние устройства сервера *Modbus RTU*.

*Примечание. По умолчанию папка **Серверы и экспорт данных** отображается только для узлов типа **Контроллер** и **Сервер данных**. Чтобы включить отображение папка на другом узле, щелкните на его названии в дереве проекта и выберите команду Видимые ресурсы узла > Серверы и экспорт данных.*

10.3 Добавление нового сервера

Для добавления нового сервера:

- Щелкните правой кнопкой на папке **Серверы и экспорт данных** узла, в контекстном меню выберите команду **Добавить устройство** и затем выберите *COM-порт* или *Сеть Ethernet*.
- Щелкните правой кнопкой на значке добавленного порта или сети и добавьте подключенные к нему сервер.
- Щелкните правой кнопкой на значке добавленного сервера и добавьте регистры или параметры согласно протоколу передачи данных.



Среда разработки автоматически показывает в контекстном меню только те серверы, регистры или параметры, которые можно подключить к выбранному устройству. Как правило, дерево серверов передачи данных соответствует описанию протокола сервера. Перед добавлением сервера рекомендуется ознакомиться с описанием протокола передачи данных.

Примечание. Шаблоны серверов всех поддерживаемых протоколов передачи данных хранятся в проекте. Список шаблонов заносится в проект в момент создания и определяется версией среды разработки Каскад-САУ, в которой он был создан. Если в новой версии Каскад-САУ появляется поддержка нового протокола, то необходимо обновить версию проекта, чтобы добавить в него шаблон сервера нового протокола и чтобы этот сервер появился в списке устройств, доступных для создания. Подробнее об обновлении версии проекта см. п. 0.

10.4 Изменение адреса сервера

Некоторым серверам или регистрам необходимо указать адрес, который используется для опроса. Этот адрес указывается в дереве проекта в скобках после названия сервера или регистра.

Изменения адреса выполняется аналогично изменению адреса устройства ввода-вывода (см. п. 8.5).

10.5 Свойства сервера

Для настройки свойств сервера дважды щелкните левой кнопкой значок родительского устройства и перейдите в таблицу на вкладке **Проводник**. Выделите строку с сервером или его регистром и введите значения свойств.

Свойства сервера аналогичны свойствам устройств ввода-вывода (см. п. 8.6) за исключением указанных ниже.

Точка ввода - точка, в которую записывается значение команды управления, полученной от внешней системы. Значение команды управления может записываться одновременно в несколько точек. Точкой ввода может быть точка любого типа.

Точка вывода - точка, значение которой записывается в регистр сервера и выдается по запросу во внешнюю систему. Для записи в один регистр может использоваться только одна точка. Точкой вывода может быть точка любого типа.

10.6 Параметры работы сервера

Настройка дополнительных параметров, необходимых для работы сервера, аналогична настройке параметров устройств ввода-вывода (см. п. 8.7).

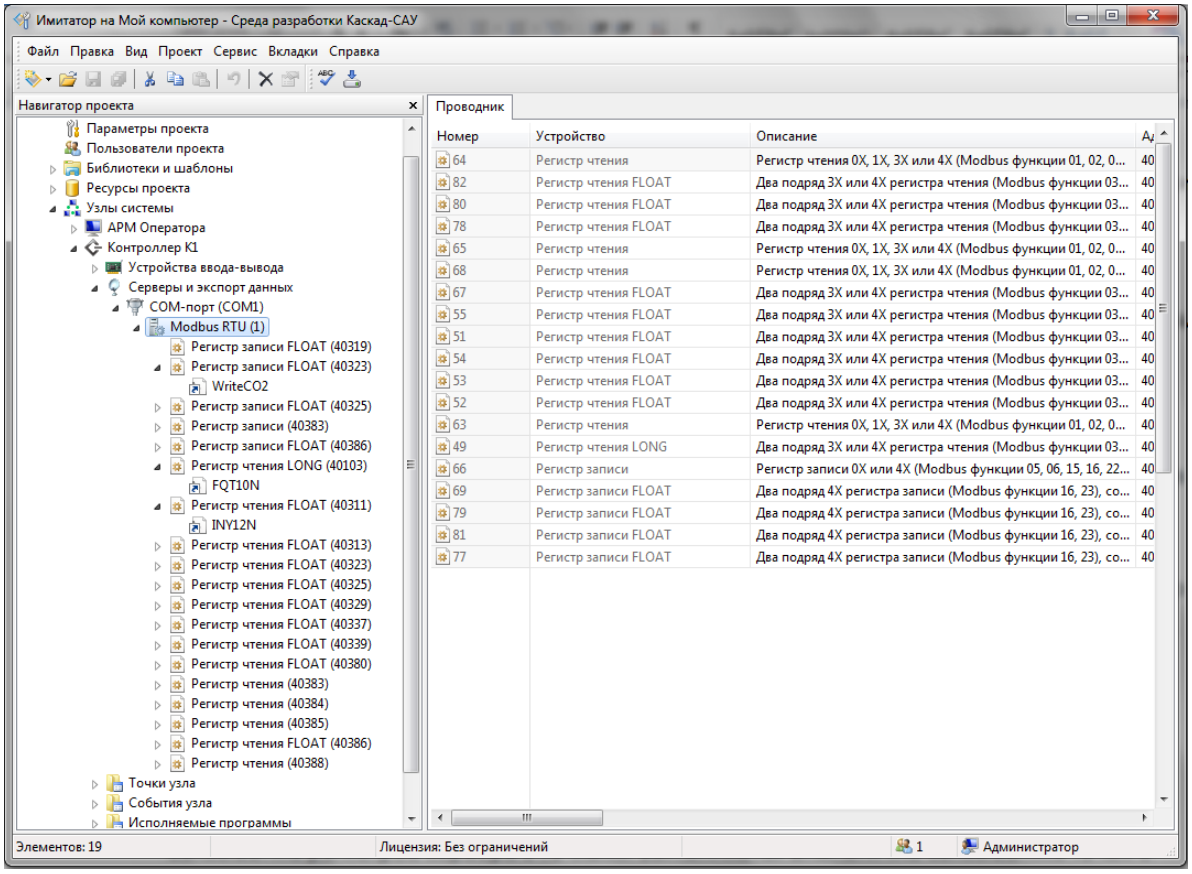
10.7 Чтение и запись значений регистров сервера

Чтобы выполнять чтение или запись значений сервера нужно привязать к регистрам точки:

- при запросе из внешней системы регистра чтения будет передано значение привязанной к нему точки;
- значение команды управления, полученное от внешней системы в *регистр записи,* будет записано в привязанную к нему точку;

В дереве проекта привязанные к регистрам точки отображаются в виде ярлыков:

- Значение одной точки может быть записано одновременно в несколько регистров чтения. В этом случае ярлык точки будет повторяться сразу у нескольких разных регистров.
- Команда управления одного регистра записи может быть записана одновременно в несколько точек. В этом случае у регистра будет несколько ярлыков разных точек.



10.8 Диагностика состояния сервера

Диагностика состояния сервера данных не поддерживается.

10.9 Связывание регистров сервера с точками

Связывание регистров сервера с точками выполняется аналогично связыванию точек с устройствами ввода-вывода (см. п. 8.9).

11 События и тревоги

11.1 Система событий Каскад-САУ

Система событий Каскад-САУ выполняет обнаружение и выдачу уведомлений о различных авариях, отклонениях от нормы, изменениях параметров и прочих событиях, связанных с технологическим процессом или работой среды исполнения.

Система событий позволяет обнаруживать и выдавать уведомления о следующих событиях:

- Изменение значения дискретной точки;
- Выход значения аналоговой точки за заданные пределы;
- Изменение качества и режимов обработки значения точки;
- Действия оператора;
- Изменение состояния среды исполнения.

Обнаружение событий, связанных с технологическим процессом, выполняется по отклонению значений точек от заданных значений. Для контроля отклонения система событий использует набор условий, определяющих, какие сообщения при каких значениях точек следует выводить.

Для уведомления о событиях система событий использует текстовые сообщения различной важности и, при необходимости, звуковые сигналы. Разновидностью сообщений высокой важности, требующих особого внимания оператора, являются предупреждения и сообщения об авариях (тревоги).

Для отображения сообщений событий используются таблицы на мнемосхемах. Сообщения могут быть сохранены в архив и просмотрены позднее с помощью программы просмотра архива событий.

Список событий, текст сообщений, важность, признак сохранения в архиве и прочие параметры событий настраиваются пользователем при настройке проекта.

В проектах с несколькими узлами каждый узел может иметь свой набор событий. События узла будут автоматически рассылаться на другие узлы этого проекта.

11.2 События и условия

Событие - это параметр среды исполнения Каскад-САУ, предназначенный для выдачи текстовых и звуковых сообщений по значениям точек.

Событие связывается с точкой и содержит набор условий. Условие - это правило, с помощью которого указывается возможное значение или диапазон значений точки события и соответствующее ему сообщение.

Система событий выдает сообщение при переходе значения точки в одно из значений или диапазон значений, заданных условиями события. Текст сообщения определяется условием, которое выполняется в данный момент. Если значение точки не удовлетворяет ни одному условию события, то выдается сообщение специального условия **Возврат к норме**.

*Примечание. Каждое событие может иметь только одно условие **Возврат к норме**.*

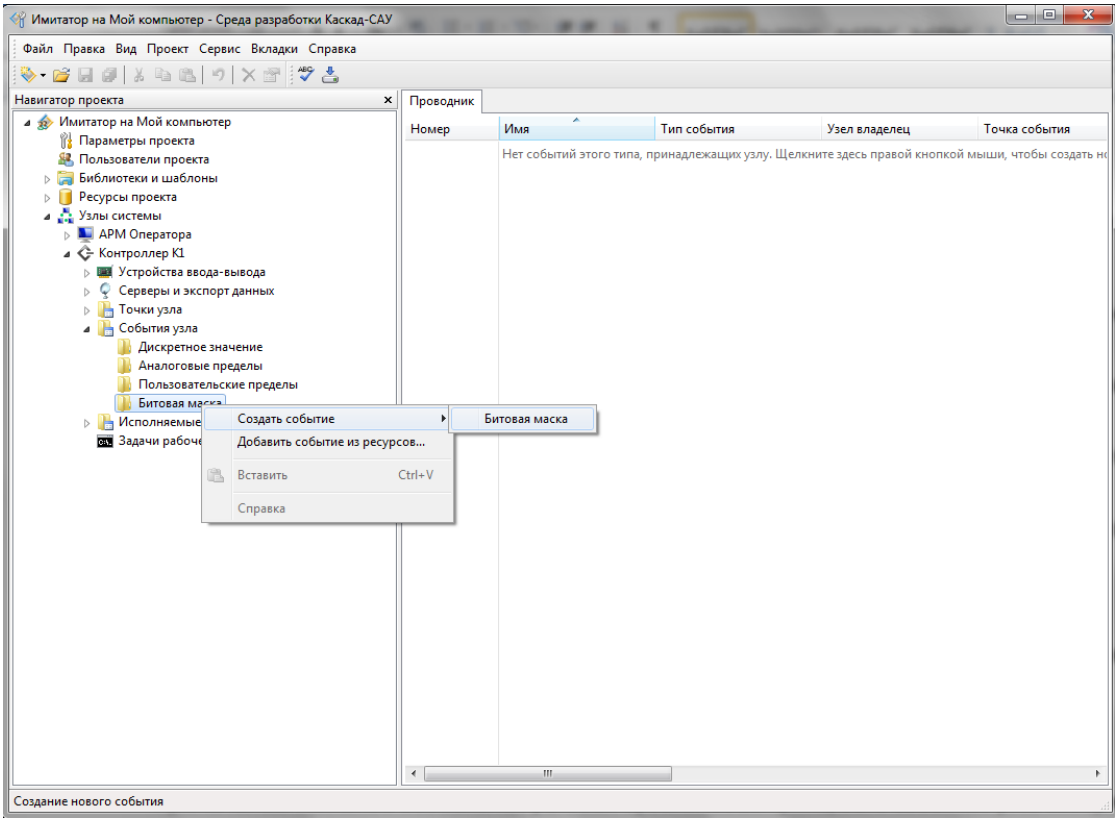
Подробнее о том, как среда исполнения выполняет вычисление состояния и квитирование событий см. п. 15.1.

11.3 Создание нового события

Для создания нового события раскройте в дереве проекта папку узла, щелкните правой кнопкой на папке **События узла**, выберите команду **Создать событие** и затем выберите тип нового события:

- **Дискретное значение** – применяется для выдачи сообщений при равенстве значения точки указанному значению.
- **Аналоговые пределы** – применяется для уведомления о выходе значения точки за пределы пороговой тревоги, заданные в параметрах точки.
- **Пользовательские пределы** – применяется для уведомления о выходе значения точки за пределы, заданные пользователем.
- **Битовая маска** – применяется для выдачи сообщения при наличии указанных битов в значении точки.

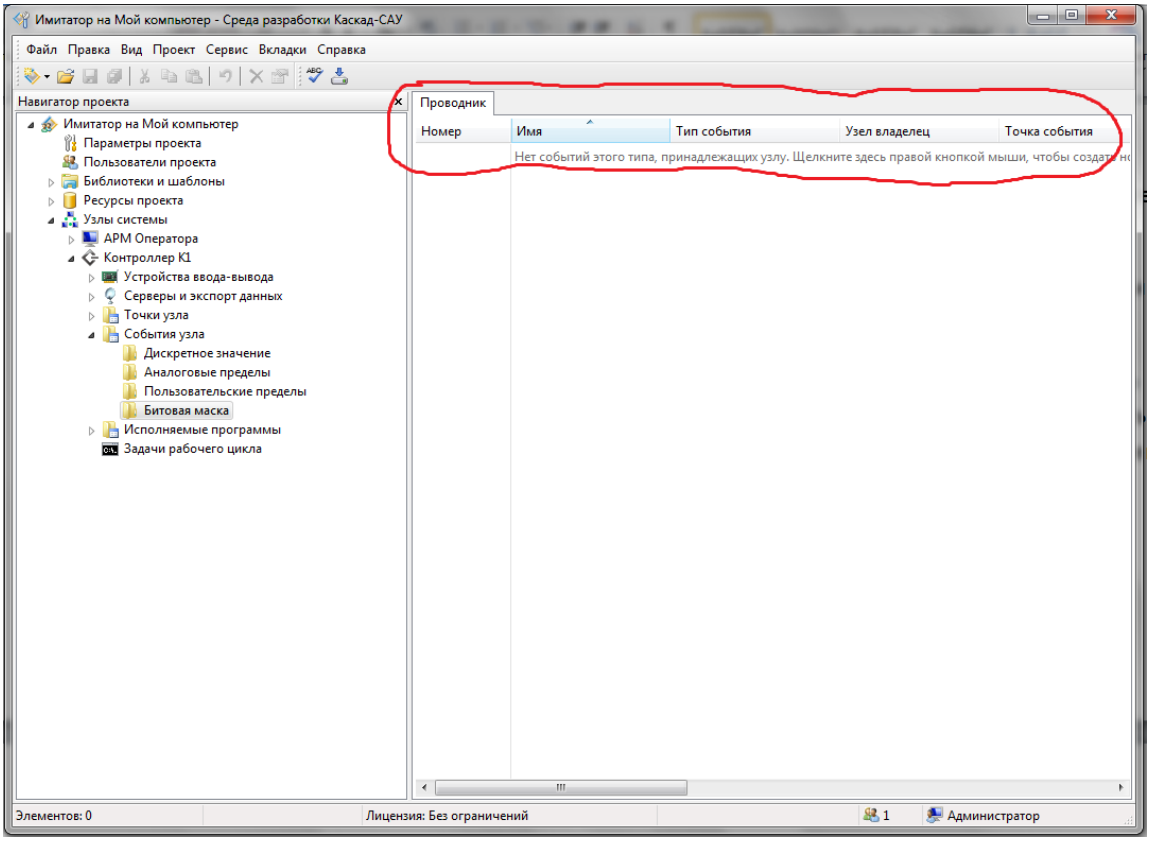
Щелкните в дереве проекта правой кнопкой на созданном событии, выберите команду **Добавить условие** и добавьте необходимые условия для события. Набор доступных условий зависит от типа события.



Примечание. Команда добавления условия будет недоступна, если событие уже содержит выбранное условие и это условие может содержаться только в единственном экземпляре. Например, условие ***Возврат к норме****.*

11.4 Свойства события

Для настройки свойств события дважды щелкните левой кнопкой значок **События узла** или на дочернюю папку типа событий и перейдите в таблицу на вкладке **Проводник**. Выделите строку с событием и введите значения свойств.



Номер - порядковый номер события в таблице событий. Значение свойства выбирается автоматически при создании события и не может быть изменено. При удалении события ее номер может быть назначен следующему новому событию.

ИД - уникальный номер события в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя события, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя не может содержать одновременно русские и латинские буквы. Имя события должно быть уникальным в проекте. Регистр символов имеет значение.

Описание - описание события, не более 127 символов.

Примечание – строка примечания события, не более 247 символов. Используется для ввода произвольной дополнительной информации по событию. Не используется во время исполнения проекта.

Тип события – определяет набора условий, использующихся для вычисления состояния события:

- **Дискретное значение** – применяется для выдачи сообщений при равенстве значения точки указанному значению.

- **Аналоговые пределы** – применяется для уведомления о выходе значения точки за пределы пороговой тревоги, заданные в параметрах точки.
- **Пользовательские пределы** – применяется для уведомления о выходе значения точки за пределы, заданные пользователем.
- **Битовая маска** – применяется для выдачи сообщения при наличии указанных битов в значении точки.

Подробнее о типах событий см. п. 11.5.

Узел-владелец - узел проекта, который меняет состояние события. Только один узел проекта может менять состояние события, остальные узлы будут получать состояние и соответствующее ему текст события от узла-владельца. Если событие не принадлежит ни одному из узлов проекта, то на всех узлах будет выполняться вычисление состояния этого события и, как следствие, состояние одного события может различаться на разных узлах . Подробнее о поддержке распределенных систем см. п. 16.

Точка события - точка, значение которой используется для выбора активного условия и вычисления состояния события. Если значение точки события маскировано, то вычисление состояния события отключается, активное событие становится неактивным, выводится сообщение текущего активного условия события с добавлением слова **МАСКИРОВАНО**.

Общий текст сообщений - текст, который добавляется в начало текста сообщений всех условий события.

Общий звук - звук из библиотеки звуков проекта, который воспроизводится перед воспроизведением звука активного условия события. Общий звук не воспроизводится при возврате события в неактивное состояние.

Гистерезис - допустимое отклонение значения точки события от значения активного условия, в пределах которого условие все еще остается активным. Используется для подавления ложных сообщений при небольших колебаниях значения. Для задания гистерезиса в процентах от диапазона ТЕ следует указать символ "%" после значения.

Задержка включения - не используется, зарезервировано для будущих версий.

Сохранение в архиве - включение и выключение сохранение в архивной базе данных события. Для поддержки архива в проекте должен быть создан и запущен узел типа **Архи**в с назначенным текущим архивом. Подробнее о сохранении событий в архив см. п. 12.

Монопольное событие К1-К2 - не используется, зарезервировано для будущих версий.

Точка состояния - точка, значение которой ставится в **TRUE** (1), если событие активно, в **FALSE** (0), если событие не активно. Одна точка может быть точкой состояния только для одного события или условия.

Уровень передачи события - уровень передачи значения точки с узла-владельца узлы и системы:

- **Локальный узел** - узел-владелец не передает событие на другие узлы проекта.
- **Узлы проекта** – событие передается только на другие узлы проекта. Серверы событий не могут передавать событие в другие системы управления.
- **Вышестоящие системы** – событие может быть передано в другие системы управления.

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства события и квитировать его в мнемосхемах. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства события и квитировать его в мнемосхемах. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

11.5 Свойства условия

Для настройки свойств условия дважды щелкните левой кнопкой значок события и перейдите в таблицу на вкладке **Проводник**. Выделите строку с условием и введите значения свойств.

Номер - порядковый номер условия в таблице условий. Значение свойства выбирается автоматически при создании условия и не может быть изменено. При удалении условия его номер может быть назначен следующему новому условию.

ИД - уникальный номер условия в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Точка условия - точка, значение которой используется для проверки условия. Если точка не задана, то для проверки условия используется точка события.

Условие - тип условия события. Параметр только для чтения. Задается при создании условия.

Значение условия - значение, с которым сравнивается значение точки для вычисления состояния условия.

Текст условия - текст сообщения, которое выдается при выполнении условия (условие становится активным).

Внимание! Если текст сообщение не задан, то условие игнорируется.

Важность - важность сообщения условия:

- обычное сообщение,
- предупреждение
- тревога.

Важность определяет цвет сообщения в таблице событий, системный звук события и приоритет воспроизведения звуковых сообщений. Подробнее о настройке цвета и системного звука для различных уровней важности см. п.

Уровень важности - уровень важности сообщения в диапазоне 0..1000:

- 1-399 - обычные сообщения,
- 400-799 - предупреждения,
- 800-1000 - аварии.

Используется для детального разбиения сообщений по важности, а также для выбора звука для воспроизведения при наличии нескольких не квитированных событий с одинаковой важностью.

Требовать квитирования - Определяет, нуждается ли событие, обнаруженное условием, в квитировании оператором. Требование квитирования не задается для условия возврата в нормальное состояние. Не квитированные события выделяются на мнемосхеме мигающим значком "!" в таблице событий. Для них воспроизводится звуковое сообщение.

Звук условия - звук из библиотеки звуков проекта, который воспроизводится при выполнении условия. Перед звуком условия воспроизводится общий звук события. Если звук условия не указан, то воспроизводится системный звук, соответствующий важности условия. Если в один момент времени активно несколько событий, то воспроизводится звук события с наибольшим уровнем важности. Звук активного события воспроизводится циклически до тех пор, пока событие не будет квитировано оператором.

Порядок вычисления - Порядок вычисления условий для события с несколькими условиями, за исключением условия **Возврат к норме**. Для определения активного условия среда исполнения вычисляет условия события в указанном порядке. Первое по порядку выполняющееся условие становится активным. Если не выполняется ни одного условия, то активным становится условие **Возврат к норме**.

Примечание. Для автоматического пересчета порядка вычисления после изменения значения условий, щелкните правой кнопкой на событии и выберите команду ***Пересчитать порядок*** *условий.*

Точка состояния - точка, значение которой ставится в **TRUE** (1), если условие активно, в **FALSE** (0), если условие не активно. Одна точка может быть точкой состояния только для одного события или условия.

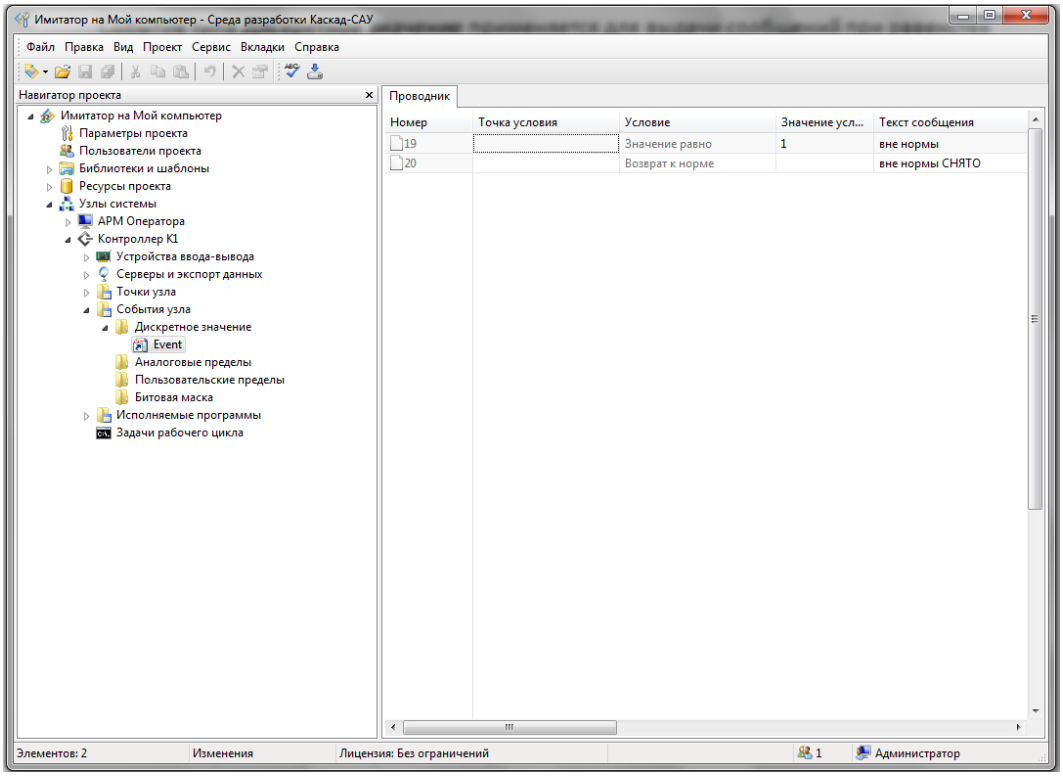
11.6 События типа Дискретное значение

Событие типа **Дискретное значение** применяется для выдачи сообщений при равенстве значения точки указанному значению.

Событие **Дискретное значение** имеет следующий набор условий:

- **Значение равно** - выполняется, если значение точки события равно указанному значению с погрешностью, равной значению свойства **Гистерезис** события.
- **Возврат к норме** - выполняется, если не выполняются другие условия события.

Событие типа **Дискретное значение** может иметь несколько условий **Значение равно** с разными значениями и только одно условие **Возврат к норме**. Такие события можно использовать, например, для выдачи сообщений по параметру, пределы которого контролируются независимыми дискретными датчиками.



*Пример 1 события **Дискретное значение** (условия и их сообщения):*

- *Значение точки равно 1 - Приточный вентилятор включен.*
- *Возврат к норме - Приточный вентилятор отключен.*

*Пример 2 события **Дискретное состояние** с несколькими условиями:*

- *Значение точки D2 равно 1 - Уровень максимальный аварийный (датчик 2).*
- *Значение точки D1 равно 1 - Уровень аварийный (датчик 1).*
- *Возврат к норме - Уровень в норме.*

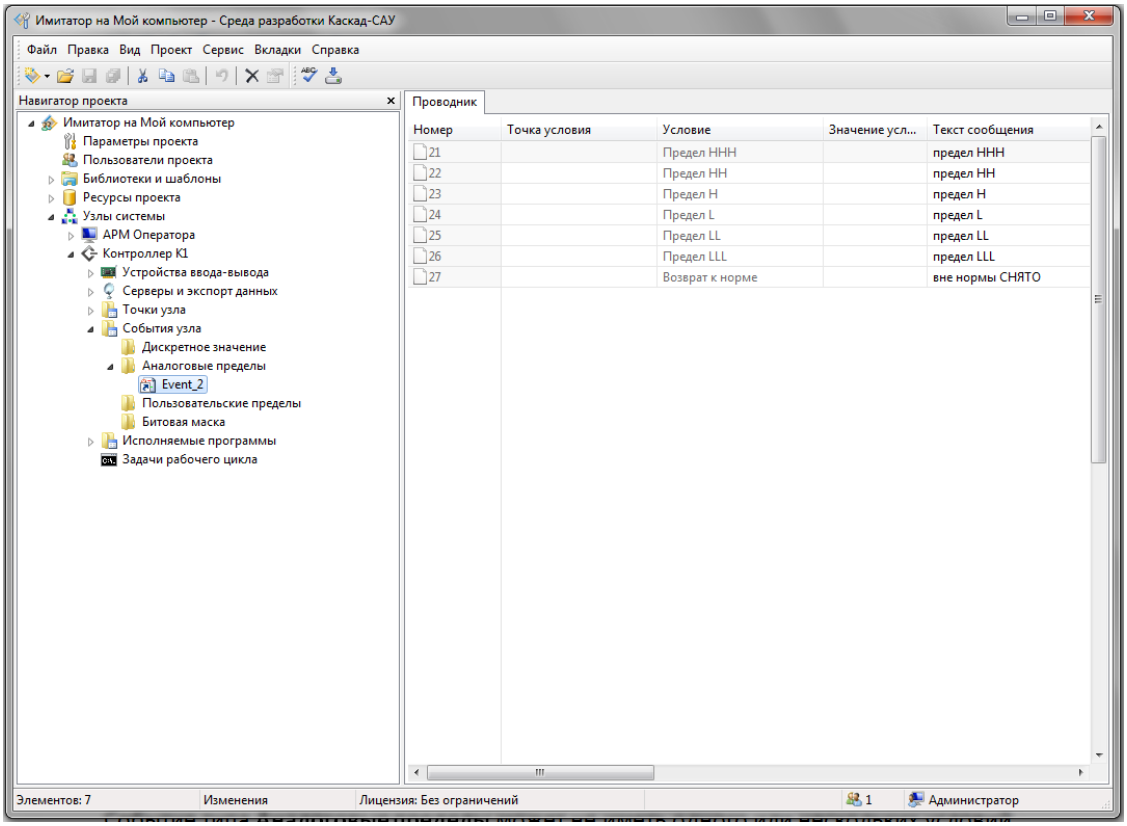
У события типа **Дискретное значение** может отсутствовать условие **Возврат к норме**. Такие события можно использовать, например, для выдачи сообщений о получении команд управления.

*Пример события **Дискретное значение** без условия **Возврат к норме**:*

- *Значение точки равно 1 - Приточный вентилятор команда ВКЛЮЧИТЬ.*

11.7 События типа Аналоговые пределы

Событие типа **Аналоговые пределы** применяется для уведомления о выходе значения точки за пределы пороговой тревоги, заданные в параметрах точки.



Событие типа **Аналоговые пределы** имеет следующий набор условий:

- **Предел HHH** - выполняется, если значение точки события больше или равно значения предела **HHH** точки. Активное условие **Предел HHH** становится неактивным, если

значение точки события становится меньше значения предела **HHH** точки минус значение свойства **Гистерезис тревоги** этой точки.

- **Предел HH** - работает со значением предела **HH** точки аналогично условию **Предел HHH.**
- **Предел H** - работает со значением предела **H** точки аналогично условию **Предел HHH.**
- **Предел LLL** - выполняется, если значение точки события меньше или равно значения предела **LLL** точки. Активное условие **Предел LLL** становится неактивным, если значение точки события становится больше предела **LLL** точки плюс значение свойства **Гистерезис тревоги** этой точки.
- **Предел LL** - работает со значением предела **LL** точки аналогично условию **Предел LLL.**
- **Предел L** - работает со значением предела **L** точки аналогично условию **Предел LLL.**
- **Возврат к норме** - выполняется, если не выполняются другие условия события.

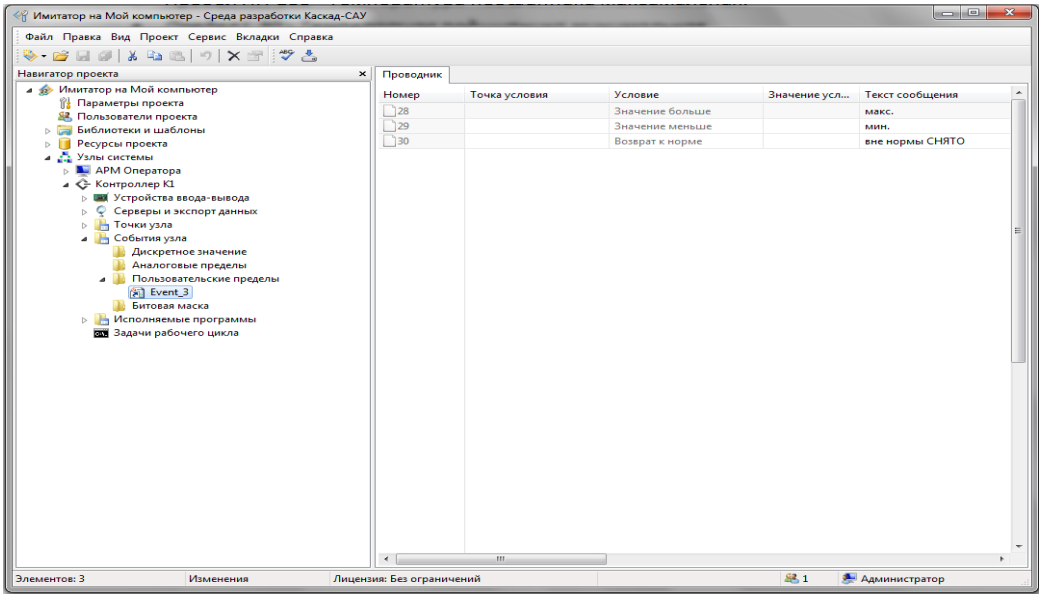
Событие типа **Аналоговые пределы** может не иметь одного или нескольких условий пределов.

*Пример события **Аналоговые пределы** (условия и их сообщения):*

- *Предел HHH 120 - Температура подшипника максимальная аварийная.*
- *Предел HH 100 - Температура подшипника максимальная.*
- *Предел L 30 - Температура подшипника минимальная.*
- *Возврат к норме - Температура подшипника вне нормы СНЯТО.*

11.8 События типа Пользовательские пределы

Событие типа **Пользовательские пределы** применяется для уведомления о выходе значения точки за пределы, заданные пользователем.



Событие типа **Пользовательские пределы** имеет следующий набор условий:

- **Значение больше** - выполняется, если значение точки события больше или равно значения условия. Активное условие **Значение больше** становится неактивным, если значение точки события становится меньше значения условия минус значение свойства **Гистерезис** события.
- **Значение меньше** - выполняется, если значение точки события меньше или равно значения условия. Активное условие **Значение меньше** становится неактивным, если значение точки события становится больше значения условия плюс значение свойства **Гистерезис** события.
- **Возврат к норме** - выполняется, если не выполняются другие условия события.

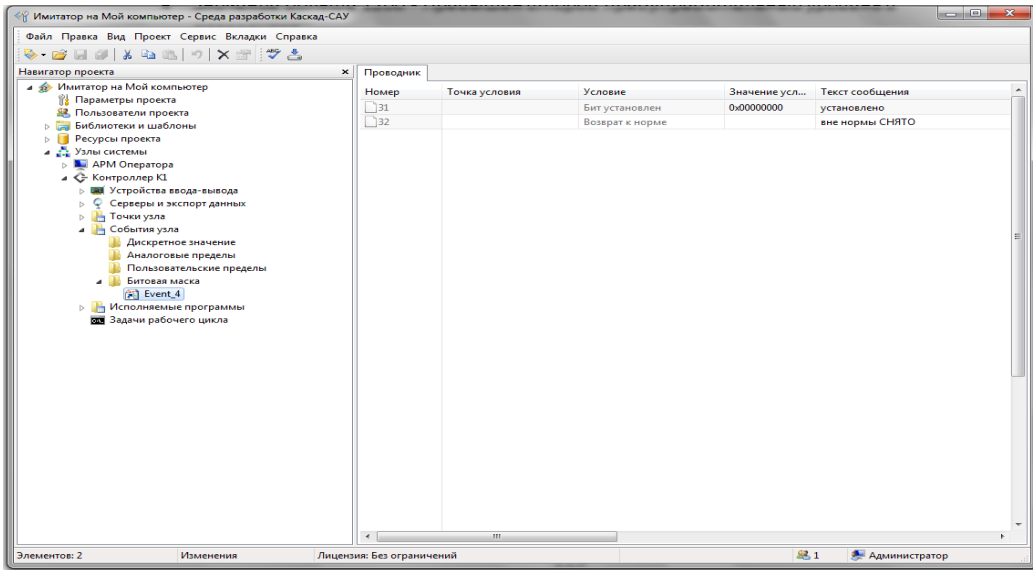
Событие **Пользовательские пределы** может иметь несколько условий **Значение больше** и **Значение меньше** с разными значениями и только одно условие **Возврат к норме**.

*Пример события **Пользовательские пределы** (условия и их сообщения):*

- *Значение больше 1400 - Превышен второй аварийный уровень в резервуаре.*
- *Значение больше 1300 - Превышен первый аварийный уровень в резервуаре.*
- *Значение больше 1200 - Превышен второй предупредительный уровень в резервуаре.*
- *Значение больше 1100 - Превышен первый предупредительный уровень в резервуаре.*
- *Значение больше 1000 - Превышен технологический уровень в резервуаре.*
- *Возврат к норме - Уровень в резервуаре в норме.*

11.9 События типа Битовая маска

Событие типа **Битовая маска** применяется для выдачи сообщения при наличии указанных битов в значении точки.



Событие типа **Битовая маска** имеет следующий набор условий:

- **Бит установлен** - выполняется, если в значении точки установлен хотя бы один из битов, указанных в значении условия.
- **Возврат к норме** - выполняется, если не выполняются другие условия события.

Событие типа **Бит установлен** может иметь несколько условий **Бит установлен** с разными значениями и только одно условие **Возврат к норме**.

11.10 Системные события

Системные события - специальные события, предназначенные для уведомления об изменениях внутреннего состояния среды исполнения Каскад-САУ.

Сообщения о системных событиях выдаются автоматически, их нельзя отключить. Текст сообщений системных событий формируется средой исполнения. Эти события не требуют квитирования и всегда сохраняются в архиве.

11.11 События о действиях пользователя

События о действиях пользователя используются для уведомления о действиях, которые производит оператор в мнемосхемах.

Примеры событий о действиях пользователя:

- *Пользователь вошел в систему.*
- *Пользователь вышел из системы.*
- *Пользователь изменил значение точки.*
- *Пользователь квитировал событие.*

11.12 События об изменении статуса точки

События об изменении статуса уведомляют об установке и снятии битов статуса точки. По умолчанию эти события используются для уведомления об установке и снятии недостоверности и изменении режима обработки значения точек.

Текст сообщений событий об изменении статуса точки, формируется автоматически средой исполнения. При установке бита выдается сообщение с названием бита с добавлением слова **УСТАНОВЛЕНО**. При снятии бита выдается сообщение с названием бита с добавлением слова **СНЯТО**.

Список битов, для которых выдаются сообщения, можно изменить с помощью диалога **Биты статуса точки** системных параметров проекта (см. п.). По умолчанию выдаются сообщения для следующих битов статуса:

- Ошибка конфигурации;
- Ввод-вывод отключен;
- Ошибка ввода-вывода;
- Обрыв датчика;
- Короткое замыкание датчика;
- Маскирование параметра;
- Имитация значения;
- Испытательный режим;
- Калибровка параметра.

События об изменении статуса выдаются для всех точек проекта без исключения.

Примеры событий об изменении статуса точек:

- *Ошибка ввода-вывода УСТАНОВЛЕНО*
- *Имитация значения СНЯТО*

12 Архивирование данных и событий

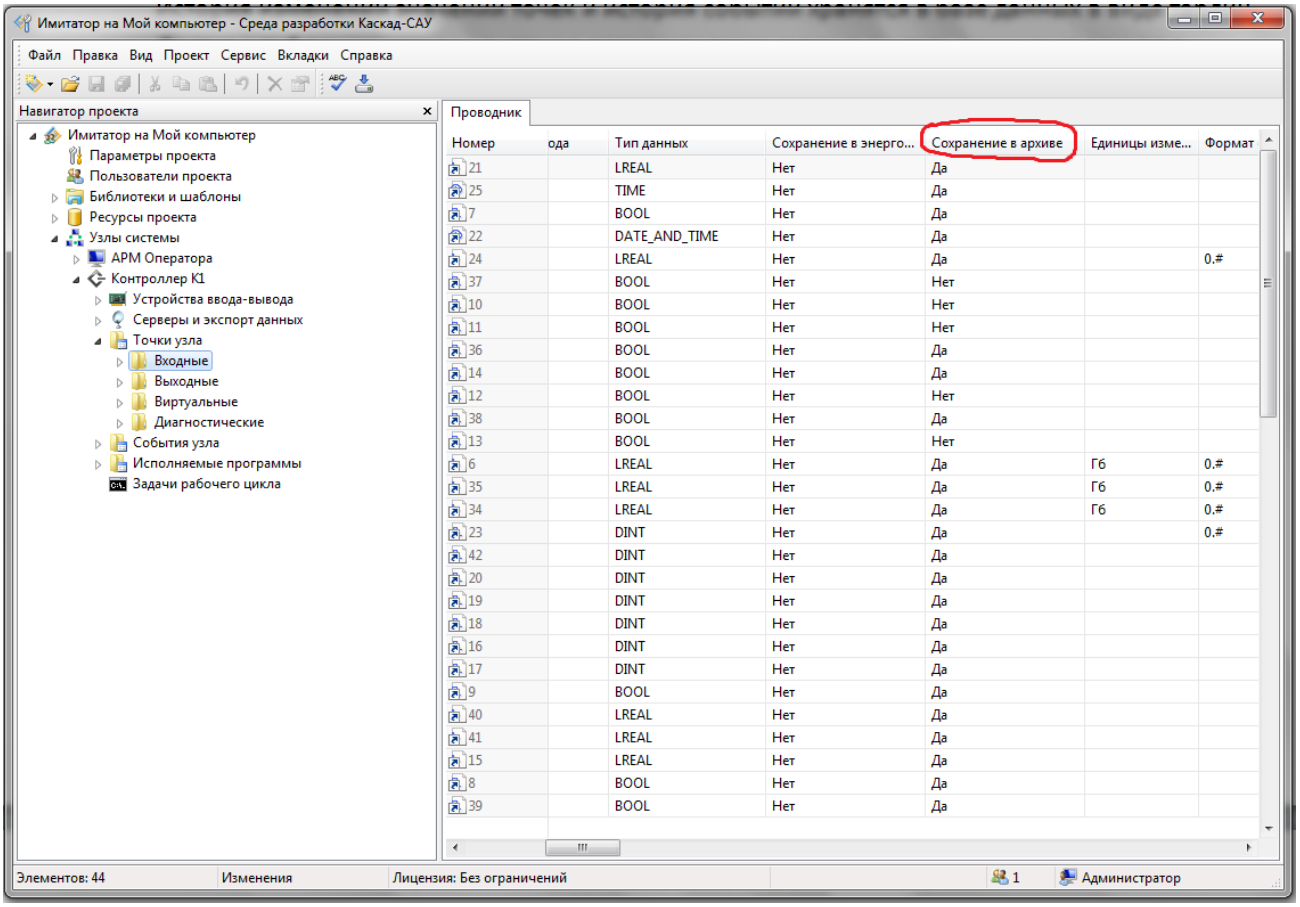
12.1 Архивы Каскад-САУ

Архив - это база данных, в которых хранится история изменений значений точек и история событий всех узлов проекта. Доступ к архивам обеспечивает сервер проектов Каскад-САУ.

История изменений значений точек и история событий хранятся в базе данных в виде таблиц. Доступ к таблицам архива на чтение разрешен всем, можно считать данных из архива с помощью сторонних программ, например, из MES-систем.

12.2 Условия записи точек и событий в архив

Чтобы точки и события записывались в архив нужно явно включить их сохранение в архиве с помощью свойства **Сохранение в архиве**.



Точки записываются в архив *по изменению значения или статуса*. Значение точки считается изменившимся, если оно отличается от предыдущего изменения более чем на величину, заданную свойством **Зона нечувствительности**. Можно принудительно включить запись точки в архив, установи бит статуса **АРХ**, например, по команде с мнемосхемы или из программы.

События записываются в архив по мере возникновения или изменения состояния (квитирования). Системные события записываются в архив всегда.

12.3 Архивируемые данные

Для точек в архив записывается значение, статус и метка времени. Метка времени содержит время узла, который зафиксировал изменение значения или статуса, или время, устройства, если устройство поддерживает выдачу данных с меткой времени. Метки времени имеют точность до миллисекунд.

Примечание. В архиве не хранятся никакие другие параметры точки, кроме значения и статуса. При просмотре архива отображается название и описание точки, считанные из проекта.

Для событий в архив записывается текстовое сообщение, значение и статус точки, вызвавшее это событие (если есть) и метка времени. Метка времени содержит время узла на момент возникновения или изменения состояния события с точностью до миллисекунд.

При удалении точки или события из проекта ее данные из архива не удаляются.

12.4 Архивный сервер

Узел проекта, на котором ведется архивирование, называется *архивным сервером*. Узел архивирования принимает от других узлов проекта значения точек и события и записывает их в свой архив.

В одном проекте может быть несколько архивных серверов. Каждый из них при запуске будет вести архивирование в свой архив независимо от других узлов.

Внимание! На компьютере, на котором запускается узел архивирования, должен быть установлен сервер проектов Каскад-САУ. Поскольку сервер проектов Каскад-САУ поддерживает только операционную систему Windows, узел архивирования должен запускаться только в операционной системе Windows. При запуске узла архивирования в операционной системе Linux ошибки запуска не произойдет, но и архивирование работать не будет.

12.5 Текущий архив узла

Каждый узел, на котором ведется архивирование, может иметь несколько архивов. Однако в один момент времени запись данных и событий может вестись только в один и них.

Архив, в который в данный момент ведется запись, называется *текущим архивом*. Чтобы сделать архив текущим щелкните на нем правой кнопкой мыши и в контекстном меню выберите команду **Назначить текущим архивом**. После изменения текущего архива загрузите изменение конфигурации на узлы.

Примечание. Выделять отдельный узел проекта для архивирования рекомендуется, но не обязательно. Чтобы включить архивирование на произвольном узле создайте на нем архив и назначьте его текущем архивом.

12.6 Сроки хранения архивов

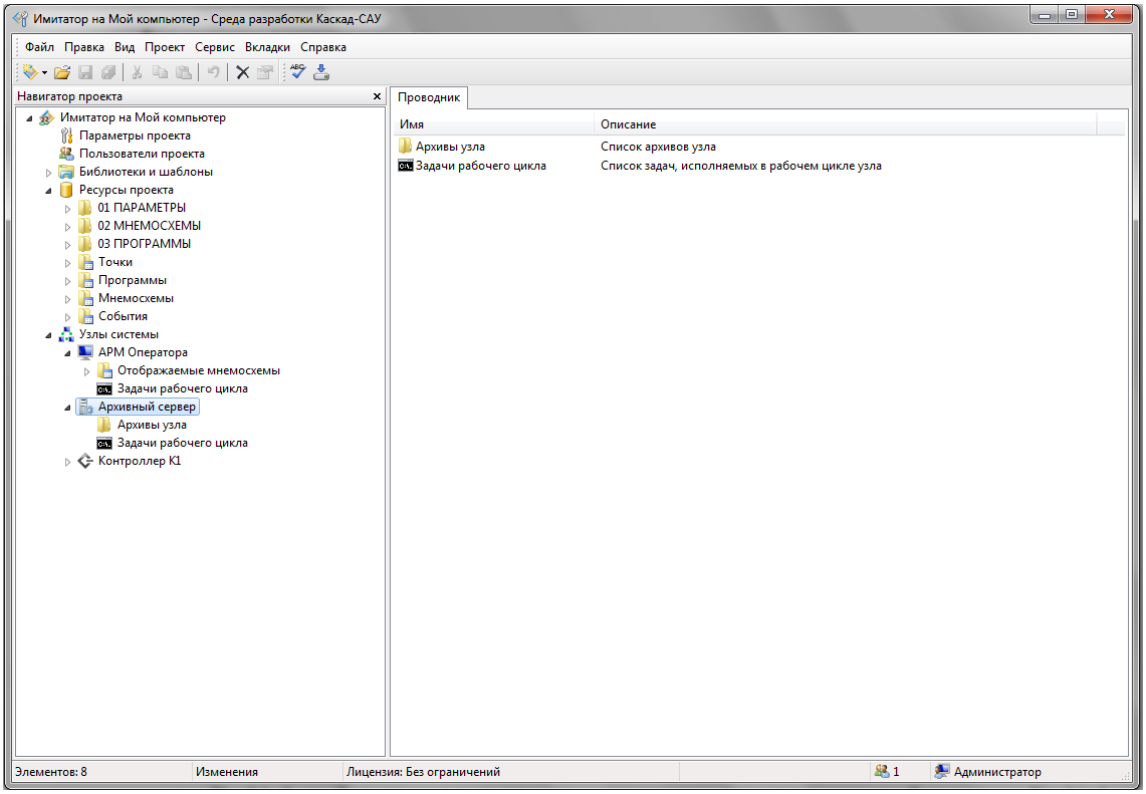
Срок хранения архивов в Каскад-САУ не ограничивается.

Длительная запись в один и тот же архив увеличивает его размер, что приводит к увеличению времени резервного копирования и повышает риск потери данных при сбоях жесткого диска. Рекомендуется не реже 1 раза в год вручную создавать новый архив и назначать его текущим архивом.

12.7 Создание нового архива

Для создания нового архива:

1. Щелкните правой кнопкой на папке **Архивы узла**, в контекстно меню выберите команду **Создать архив**. Откроется окно мастера создания нового архива.



*Примечание. По умолчанию папка **Архивы узла** отображается только для узлов типа **Архивный сервер**. Чтобы включить отображение папка на другом узле, щелкните на его названии в дереве проекта и выберите команду Видимые ресурсы узла > Архивы узла.*

2. Введите в поле **Архив** имя нового архива. Это имя будет отображаться в программах просмотр архивных трендов и архива событий при выборе архива. Имя архива не может состоять только из пробелов, а также содержать символы \/[]:|<>+=;,?*"

3. Введите в поле **База данных** полный путь и имя файла базы данных архива. Имя файла может отличаться от имени архива. Нажмите кнопку **Обзор**, чтобы выбрать другое имя файла либо папку архива.

Примечание. Если не указать путь к файлу, то архив будет создан в папке проектов по умолчанию. По умолчанию проекты и архивы Каскад-САУ хранятся в следующей папке:

Документы > Общие документы > Cascade 4.0 > projects

4. Выберите в списке **Драйвер** тип драйвера базы данных архива:

- **Firebird** - выберите, если на вашем компьютере установлен сервер баз данных Firebird. Этот драйвер рекомендуется для больших архивов и архивов, которые предполагается просматривать с других компьютеров одновременно несколькими пользователями.
- **InterBase** - выберите, если на вашем компьютере установлен сервер баз данных InterBase.
- **Firebird Embeded** - выберите, если на вашем компьютере не установлено никаких серверов баз данных. В этом случае будет использован встроенный сервер баз данных.

5. Нажмите кнопку **Создать** для создания архива. Дождитесь окончания создания.

6. В случае ошибки создания архива установить флажок **Показать журнал**, чтобы открыть журнал. В журнале указываются действия, выполняемые в ходе создания архива, и возникающие ошибки. Журнал создания будет полезен системным администраторам для поиска и устранения ошибок создания.

Примечание. Если у узла нет архивов, то новый архив сразу назначается текущим архивом.

12.8 Свойства архива

Для настройки свойств архива дважды щелкните левой кнопкой значок **Архивы узла** и перейдите в таблицу на вкладке **Проводник**. Выделите строку с архивом и введите значения свойств.

Номер - порядковый номер архива в таблице архивов. Значение свойства выбирается автоматически при создании архива и не может быть изменено. При удалении архива его номер может быть назначен следующему новому архиву.

ИД - уникальный номер архива в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя архива, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя архива точки должно быть уникальным в проекте. Регистр символов имеет значение.

Описание - описание архива, не более 127 символов.

Текущий архив - признак текущего архива узла. Текущий архив - архив, в который в текущий момент ведется запись данных и событий. Только один из архивов узла может быть текущим архивом.

База данных - путь к файлу базы данных архива на узле.

Драйвер - типа используемого драйвера базы данных архива (см. создание архива выше).

Параметры - дополнительные параметры подключения к базе данных, например, имя пользователя и пароль. Значения параметров задаются в формате *Параметр=Значение*, параметры разделяются точкой с запятой.

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства архива. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства архива. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

12.9 Просмотр и экспорт архивов

Для просмотра архивов в Каскад-САУ используются:

- программа *Архивные тренды*,
- программа *Архив событий*,
- компоненты просмотра архивов мнемосхем.

Примечание. Просмотр архивных данных поддерживается только для операционной системы Windows.

Программа *Архивные тренды* предназначена для просмотра архива изменений значений точек в виде графиков (трендов) или таблиц. Программа позволяет:

- открыть несколько окон просмотра одного или разных архивов,
- выбрать точки для просмотра архива,
- выбрать диапазон просмотра архива,
- экспортировать данные архива в текстовый файл с разделителями, книгу Excel, документ Word или файл Adobe PDF,
- сохранить изображение тренда в файл формата BPM, WMF, GIF, PNG, JPEG или Adobe PDF,
- распечатать тренд или таблицу значений.

Для запуска программы *Архивные тренды* нажмите кнопку Пуск, затем выберите Программы > Каскад-САУ 4.0 > Просмотр архивов > Архивные тренды.

Примечание. Для быстрого открытия архивных трендов из среды разработки щелкните правой кнопкой на архиве и выберите команду Архивные тренды.

Программа *Архив событий* предназначена для просмотра архива событий проекта в виде таблиц. Программа позволяет:

Программа позволяет:

- открыть несколько окон просмотра одного или разных архивов,
- выбрать диапазон просмотра архива,
- фильтровать события по ключевым словам, точкам и уровням важности,
- экспортировать события в текстовый файл с разделителями, книгу Excel, документ Word или файл Adobe PDF,
- распечатать таблицу событий.

Для запуска программы Архив событий нажмите кнопку Пуск, затем выберите Программы > Каскад-САУ 4.0 > Просмотр архивов > Архив событий.

Примечание. Для быстрого открытия архива событий из среды разработки щелкните правой кнопкой на архиве и выберите команду Архив событий.

Компоненты мнемосхем для просмотра архивов *Тренд архивный* и *События архивные* позволяют организовать просмотр архивных данных проекта на мнемосхемах АРМ оператора без использования дополнительных программ. Компоненты полностью повторяют функционал окон программ *Архивные тренды* и *Архив событий*.

Для использования компонентов в среде разработки:

- дважды щелкните на мнемосхему, чтобы открыть ее в редакторе,
- на панели *Палитра элементов* выберите в группе *Тренды и события* компонент *Тренд архивный* или *События архивные* и добавьте его на мнемосхему,
- на панели *Инспектор свойст*в с помощью свойства *Архив* выберите архив и диапазон просмотра,
- с помощью свойства *Каналы* настройте список точек для просмотра архива значений (только для компонента *Тренд архивный*),
- с помощью свойства *Фильтр событий* настройте фильтр для просмотра архива событий (только для компонента *События архивные*),
- при необходимости с помощью свойства *Панель инструментов* скройте у компонента панель инструментов или настройте кнопки панели, доступные оператору.

12.10 Резервное копирование архивов

12.11 Автоматическое резервное копирование архивов

12.12 Восстановление архива из резервной копии

13 Задачи рабочего цикла

13.1 Задачи и рабочий цикл среды исполнения

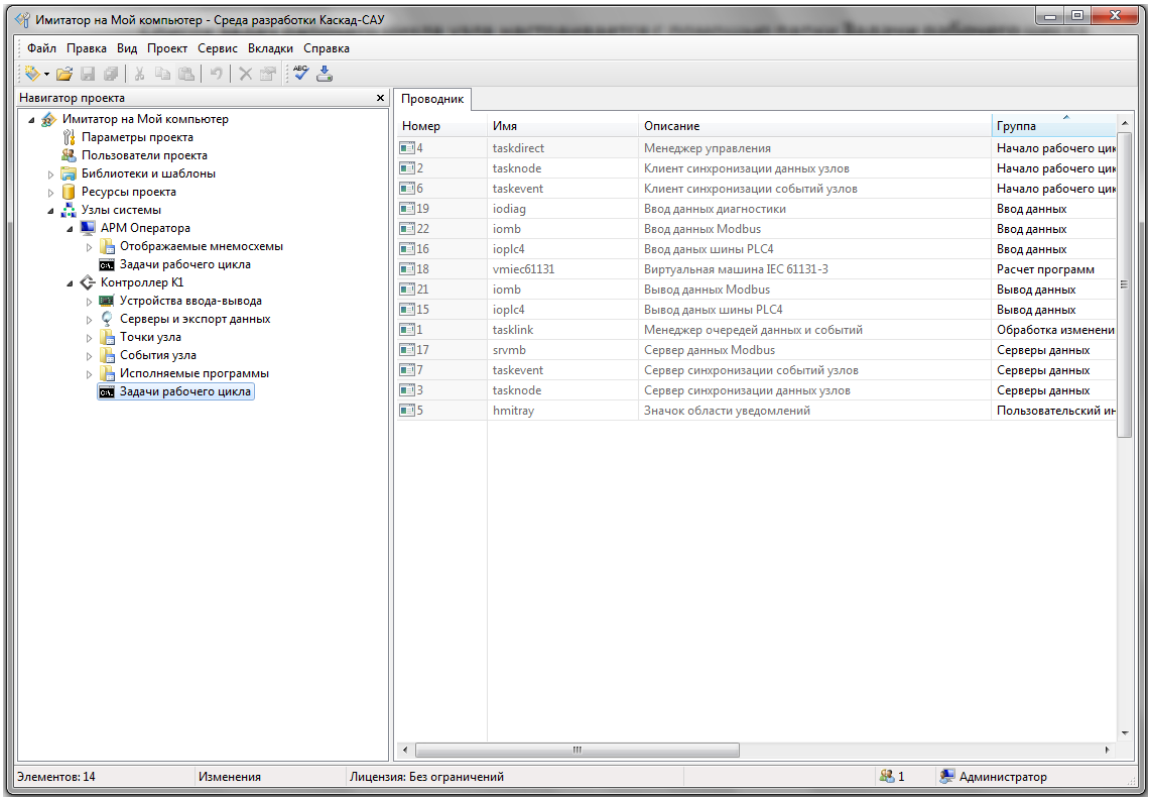
Задача – это процесс (поток) среды исполнения Каскад-САУ. Последовательная работа задач среды исполнения обеспечивает исполнение проекта на узле.

Задачи среды исполнения работают циклически. Цикл исполнения задач называется *рабочим циклом* узла.

Набор задач, работающих в рабочем цикле, определяют функции, которые выполняет узел. Например, на узле типа **Контроллер** за один цикл выполняется чтение данных с устройств ввода, обработка данных в программах, запись данных на устройства вывода, передача данных на другие узлы системы (на АРМ, на архивный сервер).

13.2 Добавление задач в рабочий цикл

Список задач рабочего цикла узла настраивается с помощью папки **Задачи рабочего цикла**.



Как правило, среда разработки формирует список задач автоматически:

- при создании узла добавляются обязательные служебные задачи,
- при добавлении устройства добавляется соответствующая задача ввода-вывода,
- при добавлении протокола передачи данных добавляется сервер данных,
- при добавлении мнемосхемы добавляется задача мнемосхем,
- при добавлении программы в список программ узла добавляется виртуальная машина алгоритмов МЭК 61131-3.
- при добавлении архива добавляются задачи архивирования.

При удалении устройства удалять задачу из списка задач нужно вручную.

Внимание! В большинстве случаев список задач рабочего цикла формируется автоматически. Изменение списка задач и порядка их исполнения вручную не рекомендуется.

Для добавления задачи в список задач рабочего цикла раскройте в дереве проекта папку узла, щелкните правой кнопкой на папке **Задачи рабочего цикла**, выберите команду **Добавить задачу** и затем выберите нужную задачу.

13.3 Свойства задачи

Для настройки свойств задачи дважды щелкните левой кнопкой значок **Задачи рабочего цикла** и перейдите в таблицу на вкладке **Проводник**. Выделите строку с задачей и введите значения свойств.

Номер - порядковый номер задачи в таблице задач. Значение свойства выбирается автоматически при создании задачи и не может быть изменено. При удалении задачи ее номер может быть назначен следующей новой задачи.

ИД - уникальный номер задачи. Значение не может быть изменено.

Имя - имя задачи. Соответствует названию исполняемого файла среды исполнения. Значение не может быть изменено.

Описание - описание задачи. Значение не может быть изменено.

Группа - группа задач в рабочем цикле. Определяет порядок исполнения задач в ходе рабочего цикла узла. Подробнее о порядке исполнения задач см. п. 15.5

Индекс в группе - определяет порядок выполнения задач в пределах одной группы. Задачи с одинаковым индексом выполняются согласно порядковому номеру.

Режим тактирования - режим ожидания завершения работы задачи. Подробнее о тактировании задач см. п. 15.4.

Контрольное время - контрольное время работы задачи в цикле в миллисекундах.

Параметры - строка дополнительных параметров задачи (см. ниже).

Пользовательские разрешения - список номеров пользовательских разрешений или диапазонов номеров, разделенный точкой с запятой. Только те пользователи, у которых есть хотя бы одно из указанных разрешений, могут менять свойства задачи. Если разрешения не указаны, то контроль разрешений пользователя не выполняется.

Уровень доступа - минимальный уровень доступа, который должен быть у пользователя, чтобы иметь возможность изменять свойства задачи. Если уровень доступа не указан, то контроль уровня доступа пользователя не выполняется.

13.4 Параметры работы задачи

Свойство **Параметры** задачи содержит строку дополнительных параметров, необходимых для настройки работы задачи. Значения параметров задаются в формате *Параметр=Значение*, параметры разделяются точкой с запятой.

Список параметров зависит от задачи. Например, для задач архивирования задается время буферизации данных для сохранения в архив. Полный список параметров приведен в описании задачи.

*Примечание. Все параметры, необходимые для настройки работы задач, настраиваются с помощью свойства **Параметры** непосредственно в проекте. Никаких дополнительных настроек в конфигурационных файлах на узлах не требуется.*

14 Пользователи и безопасность

14.1 Система безопасности Каскад-САУ

В основе системы безопасности Каскад-САУ лежит концепция учетных записей пользователей, разрешений и уровней доступа.

Учетная запись - запись с данными о пользователе системы в таблице пользователей проекта, необходимая для его идентификации и авторизации. Учетная запись хранит имя пользователя, пароль для входа в систему, список разрешений, уровень доступа и разрешенное время доступа пользователя к системе.

Все учетные записи в Каскад-САУ бывают двух типов:

- **Администраторы** - пользователи, которым разрешено все и всегда, включая изменение списка и разрешений других пользователей. Имеет полный доступ к системе. Параметры администратора нельзя изменить.

- **Обычные пользователи** - пользователи, доступ к системе которых можно ограничить с разрешений, уровня и времени доступа.

Примечание. Если выдать обычному пользователю все возможные разрешения и не ограничивать время доступа, то он по возможностям будет равен администратору, за исключением разрешения на изменение разрешений пользователей. Поэтому обычный пользователь никогда не сможет заблокировать учетную запись администратора.

Разрешение - число в списке разрешений, при наличии которого пользователь может выполнять определенную функцию в системе. Список разрешений указывается для каждого пользователя в виде строки чисел, разделенных точкой с запятой.

Различают *системные* и *пользовательские* разрешения пользователя.

Системные разрешения определяют разрешенные действия с системой:

- **1** - резервное копирование, восстановление, обновление версии и изменение лимитов проектов
- **2** - изменение конфигурации,
- **3** - изменение архивов, включая добавление, изменение, удаление, резервное копирование и восстановление,
- **5** - управление с мнемосхем,
- **6** - квитирование тревог с мнемосхем,
- **7** - завершение работы узла (**АРМ оператора**).

Пользовательские разрешения - список из 30 разрешений, которыми пользователь может ограничить доступ к ресурсам проекта (точкам, программам, мнемосхемам и т.п.) по своему усмотрению. У каждого ресурса проекта, к которому требуется ограничить доступ, указываются номера пользовательских разрешений, которые должен иметь пользователь в своем списке пользовательских разрешений для получения доступа к ресурсу. Если у пользователя есть хотя бы одно из указанных разрешений ресурса, то он может изменять этот ресурс, выполнять команды управления по нему (для точек) и квитировать тревоги (для событий).

Уровень доступа пользователя используется аналогично *пользовательским разрешениям*, но в отличие от них задается одним числом. Чтобы получить доступ к изменению ресурса или выполнению команды управления (для точек) *уровень доступа* пользователя должен быть таким же и выше, чем уровень доступа у этого ресурса.

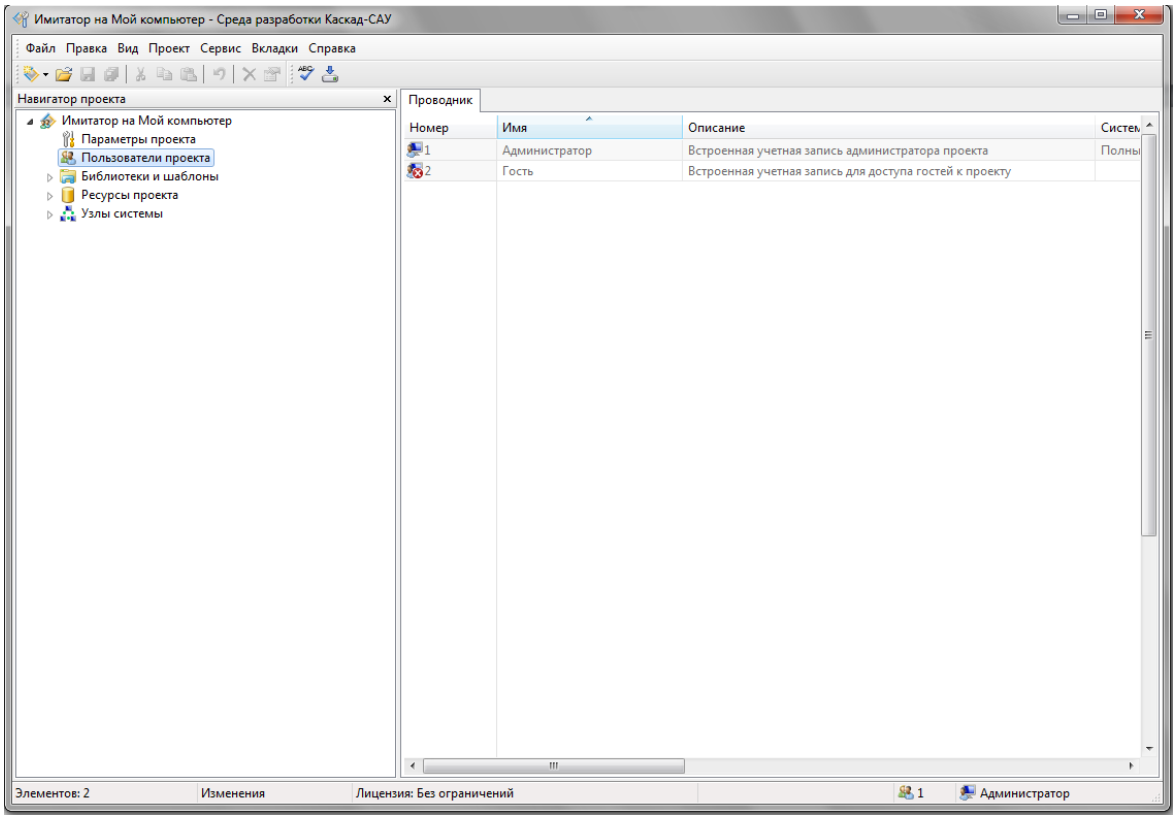
Время доступа ограничивает часы в сутках, когда пользователь может входить в систему для исполнения команд управления и квитирования тревог.

Примечание. Время доступа используется только для ограничения к управлению при работе с мнемосхемами. Время доступа в среду разработки не ограничивается.

14.2 Встроенные учетные записи

Для удобства работы каждый проект по умолчанию имеет две встроенные учетные записи пользователей:

- *Администратор* - учетная запись тип **Администратор**.
- *Гость* - учетная запись типа **Обычный пользователь**.



Встроенные учетные записи нельзя переименовать и нельзя удалить, но можно отключить.

Примечание. В целях повышения безопасности рекомендуется создать еще одну учетную запись администратора с другим именем и отключить встроенную учетную запись Администратор.

14.3 Добавление нового пользователя

Для создания нового узла щелкните в дереве проекта правой кнопкой на папке **Пользователи проекта**, выберите команду **Создать пользователя** и затем выберите тип нового пользователя (см. п. 14.1):

- **Администратор** - имеет доступ без ограничений, может изменять список пользователей проекта.

- **Обычный пользователь** - все остальные пользователи.

14.4 Свойства пользователя

Для настройки свойств пользователя дважды щелкните в дереве проекта на папке **Пользователя проекта** и перейдите в таблицу на вкладке **Проводник**. Выделите строку с пользователем и введите значения свойств.

Номер - порядковый номер пользователя в таблице узлов. Значение свойства выбирается автоматически при создании пользователя и не может быть изменено. При удалении пользователя его номер может быть назначен следующему новому пользователю.

ИД - уникальный номер пользователя в проекте. Значение свойства выбирается автоматически при создании и не может быть изменено.

Имя - имя пользователя, строка не более 31 символа. Имя не может быть пустым или состоять только из пробелов, а также содержать символы "/\[]:;|=,+*?<> . Имя пользователя должно быть уникальным в проекте. Регистр символов имеет значение.

Полное имя - полное имя пользователя, строка не более 63 символа. Используется, как правило, для хранения фамилии и имени пользователя.

Описание - описание пользователя, строка не более 127 символов.

Тип пользователя - тип пользователя, выбранный при создании. Не может быть изменен.

Состояние - состояние учетной записи пользователя:

- **Отключен** - учетная запись отключена, пользователь не может войти в систему.
- **Включен** - учетная запись включена.

Системные разрешения - список номеров или диапазонов номеров, разделенный точкой с запятой. Системные разрешения определяют разрешенные действия с системой. Подробнее о системных разрешениях см. п. 14.1.

Пользовательские разрешения - список номеров или диапазонов номеров, разделенный точкой с запятой. Чтобы получить доступ к изменению ресурса проекта или выполнению команды управления (для точек) у пользователя должно хотя бы одно из пользовательских разрешений, указанных у этого ресурса. Если пользовательские разрешения у ресурса не указаны, то контроль разрешений пользователя не выполняется, пользователь имеет полный доступ к ресурсу.

Уровень доступа - значение уровня доступа пользователя, целое число. Чтобы получить доступ к изменению ресурса проекта или выполнению команды управления (для точек) уровень доступа пользователя должен быть таким же и выше, чем уровень доступа у этого

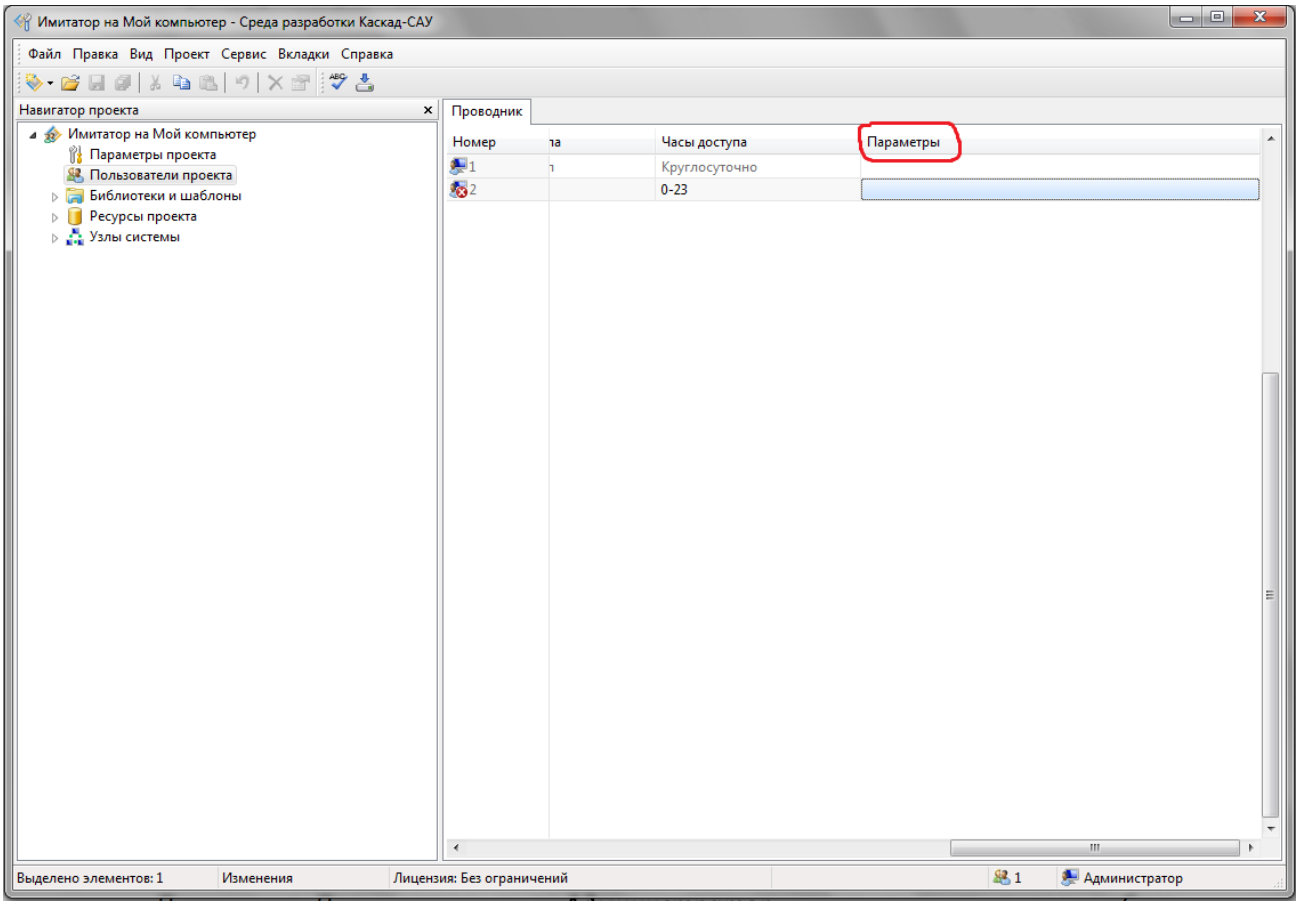
ресурса. Если уровень доступа у ресурса не указан, то контроль уровня пользователя не выполняется, пользователь имеет полный доступ к ресурсу.

Часы доступа - список часов доступа или диапазонов часов, разделенные точкой с запятой. Только в указанные часы пользователь может войти в систему при работе в мнемосхемах. По истечении указанного диапазона зарегистрированный пользователь автоматически выводится из системы. Время доступа к среде разработки не ограничивается.

Параметры - строка дополнительных параметров пользователя (см. ниже).

14.5 Параметры пользователя

Свойство **Параметры** пользователя содержит строку дополнительных параметров, необходимых для настройки работы задачи. Значения параметров задаются в формате *Параметр=Значение*, параметры разделяются точкой с запятой.



Список параметров приведен в таблице ниже.

Параметр	**Описание**
IdleTimeout	Максимальное допустимое время бездействия зарегистрированного пользователя, в минутах. Если в течение указанного времени пользователь не нажимает клавиш и двигает мышкой в мнемосхемах, то он принудительно выводится из системы.

14.6 Изменение пароля пользователя

Для изменения пароля пользователя щелкните правой кнопкой мышки на его учетной записи и в открывшемся меню выберите команду **Задать пароль**. Введите новый пароль в окне **Изменение пароля** и нажмите кнопку **ОК**.

*Примечание. Пользователь типа **Администратор** может изменить пароль любого другого пользователя в проекте.*

14.7 Отключение учетной записи пользователя

Учетную запись пользователя можно отключить. Отключенные пользователи не могут открыть проект в среде разработки, не могут отправлять команды управления и квитировать события на мнемосхемах.

Для отключения пользователя щелкните правой кнопкой мышки на его учетной записи и в открывшемся меню выберите команду **Отключить пользователя**. Для включения пользователя щелкните правой кнопкой мышки на его учетной записи и в открывшемся меню выберите команду **Включить пользователя**.

*Внимание! Чтобы иметь возможность управлять пользователями, в проекте всегда должен быть хотя бы один включенный администратор. При отключении учетной записи типа **Администратор** убедитесь, что в проекте есть как минимум еще одна запись администратора, и вы помните ее пароль.*

15 Устройство и работа среды исполнения

15.1 Среда исполнения Каскад-САУ

Среда исполнения Каскад-САУ – это набор исполняемых файлов и библиотек, используемых для запуска задач рабочего цикла узла.

Среда исполнения включает в себя:

- Задачи ввода-вывода;
- Драйверы различных протоколов и устройств;
- Виртуальную машину расчета алгоритмов МЭК 61131-3;
- Задачи пользовательского интерфейса;
- Задачи для архивирования данных;
- Серверы данных;
- Служебные задачи.

15.2 Рабочий цикл среды исполнения

Задачи среды исполнения работают циклически. Цикл исполнения задач называется *рабочим циклом* узла. Список задач рабочего цикла настраивается для каждого узла по отдельности, в большинстве случаев среда разработки делает это автоматически (см. п. 13.2).

Рабочим циклом среды исполнения управляет служебная задача *администратор среды исполнения* ***taskadm***.

15.3 Запуск и остановка задач рабочего цикла

Запуск и остановку задач рабочего цикла выполняет *администратор среды исполнения* во время загрузки на узел новой конфигурации. Администратор среды исполнения автоматически запускает новые задачи, добавленные в список задач рабочего цикла, и останавливает задачи, которые убраны из списка задач рабочего цикла.

Внимание! Запуск и остановка задач может занять продолжительное время. После добавления новой задачи в список задач рабочего цикла рекомендуется контролировать загрузку конфигурации на этот узел.

Если во время запуска задачи произошла ошибка, то администратор отключает исполнение задачи и продолжает работу рабочего цикла без нее.

Некоторые задачи могут не поддерживаться в некоторых операционных системах. Такие задачи администратор также отключает от исполнения в рабочем цикле.

15.4 Тактирование задач рабочего цикла

В ходе рабочего цикла администратор среды исполнения по очереди передает управление задачам рабочего цикла:

- администратор посылает задаче сигнал на начало работы,
- задача выполняет свою работу и возвращает назад сигнал готовности
- администратор переходит к следующей задаче.

После последней задачи администратор делается паузу до начала следующего цикла и так далее. Пауза делается такая, чтобы выдержать заданную длину рабочего цикла.

Передача управления задаче рабочего цикла называется *тактированием* задачи (от слова такт, цикл). *Режим тактирования* определяет, что администратор будет делать, если задача не вернула сигнал готовности (то есть не выполнила работу) за контрольное время.

Администратор среды исполнения поддерживает 4 режима тактирования.

Жесткое тактирование - администратор будет ждать задачу неограниченно долго. Используется для задач ввода-вывода и расчета алгоритмов, для которых важна последовательность выполнения. Например, не имеет смысла выполнять обработку данных в алгоритмах раньше, чем эти данные будут записаны в память узла, а отправку команд управления на устройства следует выполнять только после окончания расчета алгоритмов.

Задержка в работе задач жесткого тактирования, например задачи вывода при записи данных в устройство, может привести к задержке рабочего цикла.

Мягкое тактирование - по истечении контрольного времени администратор перейдет к следующей задаче. Используется для серверов данных и архивирования, задержка в работе которых не должна приводить к задержке работы задач ввода-вывода и снижению общего времени реакции системы на аварийные сигналы и команды управления.

Задержка в работе задач мягкого тактирования может привести к задержке рабочего цикла не более чем на контрольное время задачи.

Только тактирование - администратор вообще не будет ждать готовности задачи. Используется для сброса сторожевого таймера узла.

Нет тактирования - задача не будет получать сигналов о начале работы.

15.5 Порядок исполнения задач рабочего цикла

Порядок исполнения задач в рабочем цикле определяется свойствами **Группа** и **Индексом в группе** задачи. Сначала исполняются задачи одной группы, затем другой и так далее.

Группа задает стадию рабочего цикла, в которой будет исполняться задача. Стадии рабочего цикла жестко заданы, их порядок и не может быть изменен:

- **Начало рабочего цикла** - обработка команд управления и приема данных и событий от других узлов проекта.
- **Начало рабочего цикла** - обработка команд управления, синхронизация данных.
- **Ввод данных** - чтение данных с устройств.
- **Расчет программ** - исполнение программ обработки данных.
- **Вывод данных** - запись данных в устройства.
- **Обработка изменений** - обработка изменений данных и формирования событий.
- **Серверы данных** - передача данных и событий на узлы проекта и в другие системы.

- **Пользовательский интерфейс** - пользовательский интерфейс.
- **Архивирование** - загрузка данных и событий в архив.
- **Конец рабочего цикла** - сторожевой таймер.

Задачи одной группы исполняются последовательно согласно индексу в группе. Задачи с одинаковым индексом в группе выполняются согласно порядковому номеру.

15.6 Обработка команд управления

Команда управления - это команда установки значения или статуса точки узла. Команды управления посылают мнемосхемы и серверы данных.

Обработку команд управления выполняет служебная задача *Менеджер управления (**taskdirect**)* узла.

Менеджер управления принимает команды управления от мнемосхем и серверов своего узла или других узлов проекта и ставит их в очередь команд. Команды из очереди команд выполняются в начале рабочего цикла узла по сигналу администратора среды исполнения.

Команда установки значения - это команда на изменение значения точки в памяти узла. Из нескольких команд установки значения по одной точке, полученных подряд в течение одного цикла, выполняется последняя команда.

Команды установки значения могут выполняться в двух режимах: **уровнем** или **импульсом**. Настройка режима выполняется в параметрах команды (для мнемосхем) или параметрах устройства (для серверов).

Команды управления **уровнем** выполняются как простое изменение значения точки на значение из команды.

Команда управления **импульсом** выполняется в следующем порядке:

- Значение точки меняется на значение команды.
- По истечении времени импульса (по умолчанию 2 секунды или 4 такта при рабочем цикле 500 мс) значение точки меняется на значение точки, заданное свойством **Начальное значение**.
- По истечении времени сброса (по умолчанию 1 секунда или 2 такта при рабочем цикле 500 мс) значение точки снова меняется на начальное значение точки, заданное свойством **Начальное значение**.

Длительность импульса и длительность сброса настраиваются параметрами **ImpulseTime** и **ImpulseReset** задачи *менеджер управления*. Сброс импульса выполняется в начале рабочего цикла узла, следующего по истечении времени импульса.

Внимание! Все другие команды управления импульсом по одной точке, полученные во время исполнения команды импульса по этой точке, игнорируются.

*Примечание. Команда изменения значения точки **уровнем** прерывает выполнение команды **импульсом** по этой точке.*

Команда установки статуса - это команда на установку или снятие битов в статусе точки. Несколько команд установки статуса по одной точке, полученные в течение одного цикла, суммируются.

Команды установки статуса может отправить только оператор с мнемосхем. Оператору доступны для изменения биты МАСК, НДСТ, ИМИТ, ИСП, ЗАП, АРХ. Можно ограничить доступные оператору биты отдельно по каждой точке с помощью свойства точки **Биты статуса, доступные оператору**.

Команду управления по точке может выполнять только узел, который является ее владельцем. В распределенных системах менеджер управления автоматически перенаправляет команду на узел-владельца точки (см. п. 16.5).

15.7 Ввод данных из устройств в точки

Каждый узел проекта может считывать данные из подключенных к нему устройств ввода во входные точки. Список устройств определяется в конфигурации устройств вывода-вывода узла.

Чтение данных из устройств выполняют *задачи ввода (**taskxxx**)* с помощью драйверов. Каждый драйвер выполняет опрос устройств в отдельном потоке асинхронно с рабочим циклом узла, чтобы не задерживать работу задачи ввода.

Все драйверы поддерживают одновременный опрос нескольких устройств, подключенных к разным линиям связи (одна линия связи - это один COM-порт или TCP/IP соединение). Устройства на одной линии связи опрашиваются параллельно с опросом устройств на других линиях. Устройства на одной линии связи опрашиваются последовательно. После опроса всех устройств на лини драйвер выдерживает паузу, заданную параметром **ScanDelay** в параметрах линии связи. Длительность паузы определяет время цикла опроса устройств на линии.

Внимание! Одно медленное устройство на линии может задерживать опрос всех других устройств на этой линии. Для ускорения опроса медленных устройств следует подключить их к разным линиям связи, например, к разным COM-портам.

Ответ устройства на команду опроса является для драйвера признаком наличия связи с этим устройством. Признак наличия связи драйвера используется задачей ввода для формирования значения **Диагностических** точек (см. п. 15.8).

Драйвер помещает считанные с устройства данные в промежуточный буфер. Задача ввода записывает данные из буфера драйвера в точки один раз за рабочий цикл по команде администратора среды исполнения.

Задача ввода записывает в точку последнее известное значение устройства на момент начала рабочего цикла. Если значение устройства меняется несколько раз за цикл узла, то эти изменения будут потеряны. Чтобы не потерять значение следует уменьшить размер цикла узла и паузу между циклами опроса драйвера (параметр **ScanDelay**).

*Примечание. Рекомендуется назначать значение параметра **ScanDelay** равным половине длительности рабочего цикла узла.*

*Внимание! На некоторых контроллерах слишком маленькое значение **ScanDelay** может привести к зависанию системы из-за того, что высокоприоритетный поток опроса устройства в драйвере не даст работать другим задачам.*

Задача ввода записывает значение в свои точки каждый такт, независимо от того, поменялись эти значения на устройстве или нет. Если значение входной точки изменить в цикле, например, в программе, то в начале следующего цикла оно снова будет перезаписано задачей ввода значением из устройства.

Перед записью значения в точку задача ввода выполняет над значением устройства несколько преобразований. В общем случае ввод значения устройства в значение точки выполняется в следующей последовательности:

1. Драйвер считывает значение с устройства. Если от устройства нет ответа, то драйвер сбрасывает признак наличия связи с устройством.

2. Задача получает у драйвера значение устройства и признак наличия связи.

3. Если у точки включен режим имитации (установлен бит **ИМИТ** в статусе), то задача игнорирует значение драйвера, не меняет значение точки, сбрасывает в статусе точки бит ошибки ввода-вывода **ВВ**, биты аппаратных ошибок **ОБ**, **КЗ**, **ГР** и бит недостоверности **НДСТ**, и на этом завершает ввод значения устройства в точку.

Примечание. При отключении режима имитации задача ввода запишет в точку последнее считанное с устройства значение и восстановит биты аппаратных ошибок в соответствии с текущим состоянием связи с устройством (см. ниже).

4. Если связи с устройством нет, то задача игнорирует значение драйвера и не меняет значение точки, устанавливает в статусе точки бит ошибки ввода-вывода **ВВ** и на этом завершает ввод значения устройства в точку. Если у точки включен режим маскирования (установлен бит **МСК** в статусе), то текущее состояние бита ошибки ввода-вывода **ВВ** не меняется («замораживается»).

Примечание. Для случаев частых коротких потерь связи с устройством в задаче ввода предусмотрена защита от дребезга изменения бита ошибки ввода-вывода. Бит ошибки ввода-вывода устанавливается с задержкой одновременно с изменением значения точки диагностики связи с устройством (см. п. 15.8).

5. Значение преобразовывается к типу **LREAL**.

6. К значению применяется калибровочное преобразование устройства. Вид преобразования задается свойством **Калибровочное преобразование** устройства.

7. К значению применяется АЦП-ТЕ преобразование точки. Вид преобразования задается свойством **АЦП-ТЕ преобразование** точки:

- для аналоговых точек - преобразование из диапазона АЦП устройства в диапазон значений технических единиц точки,
- для дискретных точек - инверсия значения устройства.

8. Если точка имеет тип **Входная**, то к значению применяется калибровочное преобразование точки. Вид преобразования задается свойством **Калибровочное преобразование** точки.

9. Значение преобразовывается из типа **LREAL** к типу данных точки. При этом используются следующие правила:

- для преобразования в тип **BOOL**: 0 = FALSE, не 0 = TRUE.
- для преобразования в тип **INT** и **DINT** используется банковское округление (к ближайшему четному целому числу), затем приведение к диапазону целого с переполнением.
- для преобразования в тип **REAL** используется приведение с переполнением.

10. Для входных аналоговых точек выполняется первичный анализ качества значения:

- определяется выход за нижнюю границу диапазона ТЕ (обрыв),
- определяется выход за верхнюю границу ТЕ (короткое замыкание),
- выполняется контроль скорости изменения значения (градиент),
- определяется недостоверность значения.

Результатом первичной обработки являются набор битов качества значения **ОБ**, **КЗ**, **ГР**, **НДСТ** в статусе точки. Подробнее о битах качества значения см. п. 7.7.2.

Установка и снятие недостоверности значения (бит **НДСТ**) выполняется по правилам, заданным в свойствах точки:

- Недостоверность устанавливается при установке хотя бы одного из битов, перечисленных в свойстве **Причина установки недостоверности** точки.

- Если не установлено ни одного из битов, перечисленных в свойстве **Причина установки недостоверности**, то недостоверность снимается.

Примечание. Оператор не может вручную установить недостоверность для входных и выходных точек командой с мнемосхемы. Даже если он и подаст команду на установку бита недостоверности, а причин для недостоверности нет, то задача ввода на следующем цикле узла сбросит недостоверность.

11. Задача записывает значение в точку, сбрасывает в статусе точки бит ошибки ввода-вывода **ВВ** и обновляет биты аппаратных ошибок **ОБ**, **КЗ**, **ГР**. Если у точки включен режим маскирования (установлен бит **МСК** в статусе), то текущее состояние битов аппаратных ошибок **ОБ**, **КЗ**, **ГР** не меняется («замораживается»).

Внимание! После всех преобразований значение точки может выйти за диапазон ТЕ точки.

15.8 Диагностика связи с устройствами

Диагностику связи с устройствами выполняют драйверы.

Драйвер определяет наличие связи с устройством во время периодического чтения его данных. Ответ устройства на команду опроса является для драйвера признаком наличия связи с этим устройством.

Если устройство не отвечает на команду опроса, либо отвечает поврежденным или неправильным ответом, то драйвер повторяет запрос столько раз, сколько задано параметром **RetryCount** в параметрах линии связи, к которой подключено устройство. Если после **RetryCount** запросов драйвер так и не получит правильного ответа, то но считает, что связь с устройством потеряна.

Примечание. Чтобы диагностика связи заработала, у устройства в проекте должен быть хотя бы один параметр для чтения. Например, у устройства Modbus должен быть хотя бы один регистр чтения, если регистров нет вообще или есть только регистры записи, то диагностика работать не будет.

Для ввода состояния связи с устройством используются **Диагностические** точки. Значение диагностической точки, привязанной к устройству, устанавливает задача ввода на каждом цикле узла. Значение точки диагностики не зависит от типа устройства и может быть одним из следующих:

- **TRUE** - устройство работает (связь с устройством установлена),
- **FALSE** - устройство не отвечает (связь с устройством потеряна).

Для случаев частых коротких потерь связи с устройством в задаче ввода встроена защита от дребезга значения точки диагностики. Значение точки диагностики будет установлено в **FALSE**, только если устройство не отвечает в течение нескольких рабочих циклов узла. Количество рабочих циклов, на которое будет задержана изменение точки диагностики, задается параметром **OfflineFilter** устройства ввода. По умолчанию значение **OfflineFilter** установлено в 3 цикла для устройств, подключенных к COM-порту, и в 1 цикл для устройств Ethernet.

При потере связи с устройством одновременно с установкой в точку диагностики значения **FALSE** задача ввода для всех точек ввода, привязанных к этому устройству, устанавливает бит ошибки ввода-вывода **BB**. При восстановлении связи с устройством бит ошибки ввода-вывода **BB** сбрасывается.

*Примечание. Так как бит ошибки ввода-вывода меняется одновременно с изменением значения точки диагностики, то для него также работает защита от дребезга, которая регулируется параметром **OfflineFilter** устройства.*

15.9 Исполнение программ обработки данных

Исполнение программ на узле выполняет задача *виртуальная машина IEC 61131-3* (***vmiec61131***).

За один рабочий цикл виртуальная машина исполняет все программы, назначенные на исполнение на узле (см. п. 9.9). При этом каждая программа за один рабочий цикл исполняется полностью от начала до конца.

Если количество исполненных команд программы за один цикл превышает установленный лимит, то виртуальная машина считает, что произошло зацикливание программы и снимает эту программу с исполнения. Чтобы запустить программу снова, нужно исправить ошибку, приводящую к зацикливанию, перекомпилировать программу и загрузить конфигурацию на узел. Лимит команд программы настраивается параметром **CmdLimit** задачи *виртульная машина IEC 61131* (по умолчанию 1000000).

Все программы узла исполняются последовательно. Последовательность исполнения программ задается в таблице программ узла с помощью свойства **Порядок исполнения**. Программы исполняются в порядке от меньшего порядкового номера к большему. Порядок исполнения программ с одинаковым порядковым номером не определен.

Все программы исполняются изолированно друг от друга. Локальные переменные одной программы изолированы от локальных переменных других программ. Для передачи результатов расчета из программы в программу следует использовать значение точки в памяти узла либо в энергонезависимой памяти.

Остановка исполнения программы из-за критической ошибки, или обновление исполняемого кода программы влияет на работу других программ.

15.10 Вывод данных из точек в устройства

Каждый узел проекта может записывать значения из выходных точек в подключенные к нему устройства вывода. Список устройств определяется в конфигурации устройств вывода-вывода узла.

Запись данных в устройства выполняют *задачи ввода (**taskxxx**)* с помощью драйверов. Драйвер выполняет запись в устройство по команде задачи вывода и не возвращает управление задаче до тех пор, пока не закончит запись.

Внимание! Длительные операции записи в устройство может привести к задержкам рабочего цикла узла.

На время записи команды в устройство драйвер прерывает цикл опроса устройств на линии связи. До завершения записи в устройство опрос других устройств на той же линии связи не выполняться не будет. При этом опрос устройств на других линиях связи продолжается.

Внимание! Частые команды записи (каждый цикл узла) в устройство приведет к задержкам либо полностью прервет опрос этого и других устройств на той же линии связи.

Для линий связи плохого качества (определяется, например, по ошибкам контрольной суммы) драйвер повторяет команду записи столько раз, сколько задано параметром **RetryCount** в параметрах линии связи.

Примечание. Чтобы не задерживать рабочий цикл, ожидая ответ на команду от заведомо неработающего устройства, драйвер не делает запись в устройство, с которым нет связи. Поскольку наличие связи с устройством определяется периодическим чтением с него данных (см. п. 15.8), то у устройства, в которое нужно записывать данные, в проекте обязательно должны быть заданы и параметры для чтения. Например, если у устройства Modbus добавить только регистры записи, то запись работать не будет, так как драйвер ничего не читает из устройства и считает, что с ним нет связи.

Задача ввода посылает драйверу команду на запись в устройство при изменении значения точки. Если значение точки не изменилось с предыдущего цикла, то запись не выполняется. Сравнение значений точки выполняется *без учета зоны нечувствительности* точки.

Дополнительно при восстановлении связи с устройством, а также после обновления конфигурации узла задача вывода выполняет принудительную запись значений всех

выходных точек в это устройство. Для отключения принудительной записи связи следует установить параметр устройства **ForceWrite** в 0.

В любой момент можно принудительно записать текущее значение точки в устройство, установив в статусе точки бит **ЗАП**. Установить бит можно в программе или по кнопке мнемосхемы. Бит сбрасывается автоматически в начале цикла узла. Принудительная запись используется для повторной записи одного и того же значения в точку.

Перед записью значения в устройство задача вывода выполняет над значением точки несколько преобразований. Эти преобразования выполняются так же, как и для входных точек, но в обратном порядке:

1. Задача получает из памяти узла значение точки в технических единицах.

2. Значение точки преобразовывается к типу **LREAL**.

3. Если точка имеет тип **Выходная**, то к значению применяется калибровочное преобразование точки. Вид преобразования задается свойством **Калибровочное преобразование** точки.

4. К значению применяется ТЕ-АЦП преобразование точки. Вид преобразования задается свойством **АЦП-ТЕ преобразование** точки:

- для аналоговых точек - преобразование из диапазона значений технических единиц точки в диапазон АЦП устройства,
- для дискретных точек - инверсия значения точки.

5. К значению применяется калибровочное преобразование устройства. Вид преобразования задается свойством **Калибровочное преобразование** устройства.

6. Значение обрезается по диапазону АЦП устройства. Если значение больше свойства **АЦП макс** устройства, то приравнивается к **АЦП макс**. Если меньше свойства **АЦП мин** устройства, то приравнивается к **АЦП мин**.

Примечание. Значение для записи в устройство никогда не будет выходить за диапазон АЦП устройства. Чтобы отключить такое поведение, следует оставить пустыми значения диапазона АЦП у устройства.

7. Значение преобразовывается из типа **LREAL** к типу данных устройства. При этом используются такие же правила, что и при преобразовании значения входного устройства к типу точки (см. выше).

8. Задача передает значение в драйвер. Драйвер записывает значение в устройство.

9. Если запись в устройство не выполнена из-за отсутствия связи с устройством или из-за превышения количества повторов при плохом качестве линии, то задача на 1 цикл узла выставляет в статусе точки бит ошибки ввода-вывода (бит **BB**).

15.11 Обработка изменений и очередь изменившихся данных

Все значения точек, изменившиеся на очередном цикле системы, попадают в *очередь изменившихся данных*. Данные из этой очереди рассылаются на другие узлы проекта и передаются серверами данных в вышестоящие системы.

В очереди изменившиеся данные хранятся в виде пакетов. Каждый пакет хранит значения точек, изменившиеся в течение одного цикла. Каждое значение хранится вместе со статусом и меткой времени.

Формирование пакетов изменившихся данных выполняет служебная задача *Менеджер очередей данных и событий (**tasklink**)*. Эта задача включается в список задач рабочего цикла узла автоматически и является обязательной.

Внимание! Удаление менеджера очередей из списка задач рабочего цикла приведет к нарушению обмена данными между узлами проекта и работы серверов данных.

Каждый цикл менеджер очередей пробегает по всем точкам узла, сравнивает значение точки со значением на предыдущем цикле и, если оно изменилось более чем на величину, заданную свойством **Зона нечувствительности**, добавляет это значение в очередь изменившихся данных. Если изменился статус точки, то значение также добавляется в очередь. Как только все точки проверены, пакет добавляется в *очередь изменившихся данных.*

Внимание! В пакет изменившихся данных добавляются только точки, владельцем которых он является. Изменения точек, не принадлежащих ни одному узлу, не обрабатываются и не передаются на другие узлы.

Размер очереди изменившихся данных ограничен и зависит от размера доступной оперативной памяти. При заполнении очереди старые пакеты стираются.

Например, для контроллера ВСЕ-5 с 64 МБ оперативной памяти размер очереди составляет 8Мб или 6000 пакетов, этого хватает на хранение предыстории изменений за 10 минут при цикле узла в 100 миллисекунд. Для узла Windows с 2 ГБ размером оперативной памяти размер очереди составляет 48Мб или 36000 пакетов, этого хватает на хранение предыстории изменений за 1 час при цикле узла в 100 миллисекунд. При увеличении количества изменений за один цикл узла время хранения предыстории в очереди сокращается.

Внимание! Если очередь заполняется быстрее, чем данные из нее успевают передаваться на другие узлы, возможна потеря данных. В этом случае на другие узлы вместо предыстории изменений будут переданы последние известные значения точек узла.

15.12 Вычисление и квитирование событий

Каждое событие Каскад-САУ может находиться в одном из двух состояний: активном и неактивном. Если значение точки удовлетворяет хотя бы одному условию события, то событие находится в активном состоянии. Если значение точки не удовлетворяет ни одному из условий события, то событие находится в неактивном состоянии.

Вычисление состояния событий производится каждый рабочий цикл узла после того, как будет завершено чтение данных с устройств ввода, выполнены программы пользователя, записаны новые данные в устройства вывода и обработаны изменения значений точек.

Вычисление состояния и выдачу сообщений выполняет *Менеджер очередей данных и событий (**tasklink**)*. Менеджер вычисляет состояние события в следующем порядке:

- Если событие неактивно и выполняется одно из его условий, то это условие становится активным, событие переходит в активное состояние, и выдается сообщение, заданное для условия.

  Примечание. Если у события выполняется сразу несколько условий, то активным становится первое условие согласно по порядку вычисления.

- Если событие активно, но выполняется другое его условие, то событие остается активным, и выдается сообщение нового активного условия.
- Если событие активно и ни одно его условие не выполняется, то событие переходит в неактивное состояние и выдается сообщение специального условия "Возврат к норме".
- Если у события отсутствует условие "Возврат к норме" или у этого условия не задан текст сообщения, то при переходе события в неактивное состояние сообщение не выдается.

Для вычисления состояния события используется значение точки, с которой связано это событие. Точка события может быть переопределена для каждого условия события по отдельности. В этом случае для проверки условия будет использовано значение точки условия вместо точки события.

Маскирование точки события отключает вычисление состояния события, активное событие становится неактивным, выводится сообщение активного условия события с добавлением слова **МАСКИРОВАНО**. Если в условии переопределена точка события, то маскирование этой точки отключает вычисление этого условия.

Текущее состояние события может быть записано в связанную с ним точку состояния. Значение точки будет равно **TRUE** (1), если событие активно и **FALSE** (0), если не активно. Аналогичным образом в другую точку может быть записано состояние условия события.

На мнемосхемах текущие активные события выделяются значком "!" в таблице событий.

Активные события могут требовать *квитирования* – подтверждения события оператором. Не квитированные события выделяются на мнемосхемах мигающим значком "!" в таблице событий. Для таких событий воспроизводится звуковое сообщение.

15.13 Серверы передачи данных в вышестоящие системы

Передачу данных узла в вышестоящие системы выполняют *серверы данных (**srvxxx**)*. Список протоколов, для которых запускаются серверы на узле, определяется в конфигурации серверов данных узла.

Сервер передает значения точек как значения регистров/параметров, с которыми связаны эти точки. Сервер переводит значения точек в значения регистров/параметров так же, как и задачи вывода перед записью в устройство, включая применение калибровочного преобразования и преобразования ТЕ-АЦП (см. п. 15.8). При этом, в отличие от задач ввода, сервер получает значение точек не из памяти узла, а из очереди изменившихся данных, а, следовательно, значения в регистрах/параметрах сервера меняются *с учетом зоны нечувствительности*.

Если протокол сервера поддерживает передачу предыстории, то сервер передает предысторию изменений из очереди изменившихся данных узла. Однако ее емкость ограничена (см. п. 15.11) и при низкой скорости передачи или задержках на линии связи возможна потеря данных.

При получении команды управления или команды записи регистра/параметра сервер меняет значение точки, связанной с этим регистром/параметром. Перед выполнением команды сервер выполняет со значением команды такие же преобразования, как и задача ввода перед записью в точку, включая преобразование АЦП-ТЕ и калибровочное преобразование (см. п. 15.6). Однако в отличие от задач ввода, сервер никогда не записывает значения непосредственно в точку, а отправляет команду на установку значения точки в очередь команд управления узла (см. п. 15.6).

Если у точки включен режим имитации (в статусе точки установлен бит **ИМИТ**), то сервер при получении команды управления не меняет значение точки. При этом в журнале узла будет сообщение о том, что команда игнорирована из-за включенного режима имитации точки. При отключении режима имитации сервер не меняет значение точки до получения следующей команды управления по этой точке.

Сервер может выполнить команду управления как команду управления уровнем или импульсом (см. выше). По умолчанию все команды обрабатываются как команды уровнем. Режим исполнения команд можно изменить для каждого регистра/параметра сервера по отдельности (см. параметр **Impulse** в параметрах устройства).

15.14 Архивирование

Архивирование данных и событий выполняют служебные задачи *Клиент синхронизации данных узлов (**tasknode**, см. п. 16.4), Клиент синхронизации событий (**taskevent**, см. п. 16.6)* и *Загрузчик данных в архив (**taskarc**)*.

Внимание! Загрузчик данных в архив включен в состав среды исполнения только для ОС Windows. Загрузка данных в архив из других ОС не поддерживается.

Архивирование данных и событий выполняется в следующем порядке:

- Изменившиеся данные и новые события записываются в файлы оперативного архива.
- По мере накопления данные из оперативного архива загружаются в текущий архив проекта.

Оперативный архив - это набор текстовых файлов определенного формата, в котором хранятся значения точек и события. Назначение оперативного архива - ускорить запись данных в архив за счет буферизации данных, минимизировать нагрузку на СУБД при массовой записи данных в архив и обеспечить сохранность данных при повреждении архива.

Примечание. По умолчанию файлы оперативного архива хранятся в папке данных узла (см. п. 15.16).

Запись данных и событий в файлы оперативного архива выполняют *клиент синхронизации данных* и *клиент синхронизации событий* соответственно. Запись в оперативный архив включается автоматически для всех узлов типа ***Архивный сервер***. Чтобы вручную включить запись данных в оперативный архив установите параметр ***ArchiveMode*** задач и ***tasknode taskevent*** в значение 1.

Один файл оперативного архива содержит значения точек или событий за 10 минут или 100000 записей. По истечении 10 минут файл закрывается и открывается следующий файл и так далее. Время наполнения одного файла оперативного архива в минутах задается параметром ***ArchiveTime***, максимальное количество записей в одном файле задается параметром ***ArchiveRecs*** задач *клиент синхронизации данных* и *клиент синхронизации событий*.

Примечание. Не рекомендуется устанавливать время записи менее 2 минут, так как это может снизить производительность работы архива.

Внимание! Изменившиеся значения и события попадают в архив не сразу, а с задержкой в 10 минут (по умолчанию).

Загрузку данных из оперативных архивов в архив узла выполняет *загрузчик данных в архив*. Загрузка выполняется автоматически по мере готовности файлов оперативного архива. После успешной записи в архив файл оперативного архива удаляется и в архив загружается следующий готовый файл и так далее.

Если во время записи в архив произошла ошибка (например, если архив недоступен из-за повреждения файла базы данных после неправильного выключения питания), то файл оперативного архива не удаляется, загрузка данных в архив приостанавливается, чтобы позднее попытаться загрузить файл еще раз. Таким образом, до тех пор, пока архив недоступен, значения точек и события будут накапливаться в оперативном архиве. Как только архив будет восстановлен, все накопленные данные тут же будут записаны в него, потери архивных данных не произойдет.

Если запись в архив не выполнена из-за ошибки в файле оперативного архива (например, если значение точки равно NAN или INF), то расширение файла будет изменено на *.err* и файл удален не будет. Такой файл можно позднее проанализировать вручную, устранить ошибку и вернуть первоначальное расширение, чтобы он потом загрузился его снова.

Если при удалении файла, успешно загруженного в архив, произошла ошибка, то расширение файла будет изменено на *.del*. Такой файл следует удалить вручную.

Внимание! По умолчанию в ОС Windows папка данных с файлами оперативного архива расположена на диске C:. Если на диске закончится свободное место, то произойдет потеря данных из-за ошибки записи данных в файлы оперативного архива. В нормальном режиме работы в папке данных должен находиться один файл оперативного архива значений точек (.pnt), один файл текущих значений (.cur) и один файл оперативного архива событий (.evt).

Загрузчик данных загружает данные в архив узла, помеченный как *текущий*. При этом загрузчик данных не обращается непосредственно к архиву, но работает с ним через сервер проектов. Сервер проектов знает о внутреннем устройстве архива (базы данных) и порядок записи данных в нее.

На компьютере, на котором запускается архивный узел, требуется наличие установленного сервера проектов Каскад-САУ. В противном случае загрузка данных в архив выполняться не будет, на узле будут копиться файлы оперативного архива. Чтобы такая ситуация не привела к переполнению диска, количество файлов оперативного архива ограничено. По умолчанию это ограничение установлено в 4000 файлов (архив приблизительно за 30 дней). Ограничение на количество файлов оперативного архива можно изменить параметром

ArchiveMode задач ***tasknode*** и ***taskevent*.** По достижении указанного ограничения старые файлы будут удаляться, что приведет к потере архивных данных.

Для случаев, когда требуется обеспечить архивирование в ОС, отличных от Windows, например, на котроллерах Linux, в Каскад-САУ предусмотрена возможность ручной загрузки файлов оперативного архива в архив:

- Откройте папку данных узла, перейдите в родительскую папку на 3 уровня вверх (в папку с ***GUID*** проекта, см. п. 5.3) и скопируйте ее на сменный носитель.
- В среде разработки щелкните правой кнопкой на архиве, в который необходимо загрузить данные и выберите команду **Загрузить архивные данные**.
- В окне мастера загрузки архивных данных нажмите кнопку **Обзор** и выберите диск сменного носителя. Мастер автоматически найдет на носителе папку с оперативным архивом текущего проекта.

Примечание. Допускается хранить на одном сменном носителе одновременно несколько папки оперативных архивов с разными GUID. При загрузке данных в архив мастер автоматически выберет папку с оперативным архивом текущего проекта.

15.15 Поддержка энергонезависимой памяти

Энергонезависимая память используется для сохранения значения и статуса точек, которые требуется восстанавливать после перезапуска узла. Запись и чтение точек из энергонезависимой памяти выполняет *администратор среды исполнения* ***taskadm***.

В качестве энергонезависимой памяти на узле администратор использует микросхему памяти FRAM либо файл на жестком диске. Тип используемой памяти (FRAM или файл) выбирается автоматически в зависимости от типа узла. В текущей версии Каскад-САУ поддержка памяти FRAM реализована для контроллеров BCE-5 и OptiLogic L. В остальных случаев используется файл на жестком диске.

Примечание. Так как в контроллерах BCE-5 и OptiLogic L микросхема энергонезависимой памяти распаяна на плате, при замене вышедшего из строя контроллера на новый все уставки и режимы работы точек будут потеряны и их нужно будет вводить заново.

Микросхема памяти FRAM может хранить точки только одного проекта. При запуске на контроллере узла из другого проекта (например, если бывший в употреблении контроллер из ЗИП настраивается на работу с новым проектом) старое содержимое микросхемы FRAM автоматически очищается.

Файл с содержимым энергонезависимой памяти хранится в папке данных узла (см. п. 15.16) и имеет имя *dboper.bin*.

Восстановление значений из энергонезависимой памяти выполняется администратором при старте узла непосредственно перед запуском первого рабочего цикла. Начальное значение точек, для которых включено сохранение в энергонезависимую память, будет таким, какое оно было в момент завершения работы (перезагрузки) узла, а не таким, какое задано в свойствах точки.

Примечание. Для точек, у которых сохранение в энергонезависимую память отключено, значение при старте узла устанавливается в значение, определенное свойством ***Начальное значение*** *(см. п. 7.4).*

Запись точек в энергонезависимую память выполняется администратором в конце рабочего цикла. Значения точек, записанные вручную из программы (см. п. 7.7) также записываются в конце рабочего цикла, но не в момент вызова функции записи.

Для защиты от преждевременного выхода из строя Flash-дисков и SD-карт с ограниченным количеством операций записи (для случаев, когда в качестве энергонезависимой памяти используется файл на диске) администратор кэширует значения и статус точек в оперативной памяти. По умолчанию запись точек из кэша в энергонезависимую память выполняется каждые 10 рабочих циклов, период записи можно изменить параметром **NvramUpdate** узла.

Запись точек из кэша в энергонезависимую память выполняется синхронно. Пока запись не закончится, новый цикл не начинается. Например, на контроллере BCE-5 запись 1000 точек в энергонезависимую память занимает около 250 мс. При завершении работы узла выполняется принудительная запись кэша в энергонезависимую память.

15.16 Папка данных узла

Папка данных – это папка со служебными файлами среды исполнения. В папке данных хранится журнал событий, журнал запуска, последняя удачно загруженная конфигурация, файл энергонезависимой памяти.

Расположение паки данных зависит от используемой операционной системы:

- Windows: C:\Users\<UserName>\AppData\Local\Tersy\Cascade\4.0\data\<GUID>\<node>
- Linux: /home/<UserName>/.cascade4/data

В операционной системе Windows допускается одновременный запуск нескольких узлов. Каждый запущенный узел будет иметь свою папку данных, в имени которой указывается GUID проекта (см. п. 5.3) и уникальный номер узла в проекте (см. п. 6.3).

Примечание. Чтобы быстро открыть папку данных узла в операционной системе Windows нажмите клавишу SHIFT, щелкните правой кнопкой на иконке узла в области уведомлений и в открывшемся меню выберите команду ***Папка данных****.*

16 Поддержка распределенных систем

16.1 Распределенные системы управления

Системы управления, в которых логика работы распределена на два или более узлов проекта, называются *распределенными системами* управления.

Распределять логику управления на несколько узлов рекомендуется в больших системах: функции управления - на узел контроллера, отображение данных - на узел АРМ оператора, архивирование и связь с другими системами - на узлы серверов.

Примечание. Небольшие системы управления, в которых всего два узла - АРМ оператора и контроллер, также являются распределенными, даже если оба узла физически запускаются на одном компьютере.

16.2 Синхронизация данных между узлами

В основе поддержки распределенных систем Каскад-САУ лежит механизм автоматической синхронизации данных между узлами одного проекта:

- Все узлы одного проекта содержат в памяти один и тот же список точек.
- Каждый узел рассылает на другие узлы значения точек, владельцем которых он является.
- Команды управления с других узлов автоматически пересылаются для исполнения на узел, который является владельцем точки.

Примечание. Автоматическая синхронизация данных работает только для узлов одного проекта. Для приема данных с узлов другого проекта нужно вручную настраивать ввод и вывод, как из обычных устройств ввода-вывода.

Внимание! Если в результате ошибки конфигурации соединить узлы разных проектов, то синхронизация данных между этими узлами работать не будет.

16.3 Узел-владелец точки

Все задачи рабочего цикла работают со значением точек из памяти узла. Значение точки в памяти может менять только один узел из всех узлов проекта. Узел, который меняет значение точки, является владельцем точки.

Узел-владелец автоматически рассылает значения своих точек на другие узлы. Таким образом, в памяти всех узлов в каждый момент времени находится одни и те же значения точек. Узлы могут использовать значения других узлов из своей памяти, например, для отображения на мнемосхеме или для передачи в другие системы.

Как правило, владельцем точки является узел, к которому подключено устройство ввода точки или выполняется алгоритм, рассчитывающий значение этой точки. Задание узла-владельца называется *привязкой точки к узлу*. Каскад-САУ автоматически привязывает входные точки к узлу при привязке ее к устройству ввода узла.

Внимание! Виртуальные точки, значение которых вычисляется в программе, следует привязать к узлу вручную, иначе они не будут передаваться на другие узлы. Точно также следует вручную привязывать к узлу виртуальные точки, которые являются входными параметрами программы и меняются по команде с мнемосхемы, иначе они не будут переданы на узел, на котором вычисляется программа.

16.4 Рассылка точек на другие узлы проекта

Рассылку точек узла выполняют служебная задача *Сервер синхронизации данных узлов (**tasknode**)*, прием выполняет *Клиент синхронизации данных узлов (**tasknode**)*.

В конце рабочего цикла после обработки изменений *сервер синхронизации данных* рассылает изменившиеся значения точек на другие узла. Сервер рассылает на другие узлы значения всех точек независимо от типа ввода-вывода и тапа данных. Значения точек рассылаются при изменении значения больше, чем на величину **Зоны нечувствительности** (см. п. 7.4) или при изменении статуса.

Клиент синхронизации выкладывает принятые от других узлов значения в свою память в начале рабочего цикла. В случае отсутствия связи с сервером какого-либо узла более чем на 10 секунд клиент синхронизации устанавливает бит статуса **СВЗ** в точках этого узла в своей памяти, чтобы пометить, что значения точек устарели и не обновляются.

Передача данных между узлами выполняется по протоколу TCP/IP. Поэтому рассылать значения могут только те узлы, для которых в проекте указан IP-адрес, например, контроллер. У узлов, не рассылающие данные, IP-адрес можно не указывать, например АРМ оператора. IP-адрес узла задается с помощью свойства **IP-адрес К1**, порт сервера данных узла задается параметром **DataPort**.

Примечание. У нескольких узлов на одном компьютере должны быть разные порты сервера данных. В противном случае рассылка данных будет работать только у одного узла.

Каждое изменение помечается меткой времени узла с точностью до миллисекунд. Эта метка времени используется, в частности, для отображения значения на трендах, и при сохранении в архив.

Внимание! Настоятельно рекомендуется обеспечить синхронизацию времени узлов. В противном случае на трендах значения разных узлов, изменившиеся в один момент

времени, будут сдвинуты относительно друг друга. Синхронизация времени должна выполняться средствами операционной системы.

Архивный узел при синхронизации данных дополнительно получает от других узлов предысторию изменения значений из очереди изменившихся данных. Размер очереди изменившихся данных на узлах ограничен (см. п. 15.11). При использовании медленной линии связи и задержках в сети возможна задержка синхронизации и потеря данных из-за переполнения очереди.

16.5 Рассылка команд управления на другие узлы проекта

Обработку команд управления на узле узла выполняет служебная задача *Менеджер управления (**taskdirect**)*.

Менеджер управления принимает команды управления от мнемосхем и серверов данных. Если точка команды не принадлежит текущему узлу, то менеджер автоматически пересылает команду на узел-владельца точки.

Внимание! Пересылка команды управления на другой узел выполняется синхронно и может задержать рабочий цикл при задержках в сети.

Менеджер управления также является и *сервером управления* и принимает команды от других узлов. Прием команд выполняется по протоколу TCP/IP, поэтому принимать команды с других узлов могут только те узлы, для которых в проекте указан IP-адрес. IP-адрес узла задается с помощью свойства **IP-адрес К1**, порт сервера управления узла задается параметром **DirectPort**.

Примечание. У нескольких узлов на одном компьютере должны быть разные порты сервера управления. В противном случае передача команд управления будет работать только на один узел.

16.6 Рассылка событий на другие узлы проекта

Аналогично с рассылкой данных узлы рассылают на другие узлы проекта новые события, которые они сформировали. Рассылку событий узла выполняет служебная задача *Клиент синхронизации событий узлов (**taskevent**)*, прием выполняет *Сервер синхронизации событий узлов (**taskevent**)*.

Сервер синхронизации событий рассылает новые события в конце рабочего цикла после обработки изменений.

Передача событий между узлами выполняется по протоколу TCP/IP, поэтому рассылать события могут только те узлы, для которых в проекте указан IP-адрес. IP-адрес узла задается с помощью свойства **IP-адрес К1**, порт сервера событий узла задается параметром **EventPort**.

Примечание. У нескольких узлов на одном компьютере должны быть разные порты сервера событий. В противном случае рассылка данных будет работать только у одного узла.

События всегда передаются на другие узлы с предысторией изменения. При использовании медленной линии связи и задержках в сети узлы могут не успеть передать всю предысторию, что приведет к потере части событий.

17 OPC сервер Каскад-САУ

Сервер OPC Data Access включен в дистрибутив Каскад-САУ. Сервер устанавливается автоматически при установке среды исполнения.

Примечание. Сервер OPC может работать только в операционной системе Windows.

В списке серверов OPC сервер отображается под именем **Cascade.OPC.DataServer.4**.

Сервер не требует настройки и автоматически отображает точки всех узлов, запущенных на одном с ним компьютере, например точки **АРМ оператора** или **Архивный сервер**. Поскольку в распределенных системах каждый узел проекта имеет один и тот же набор точек и автоматически синхронизирует свои данные с другими узла (см. п. 16), с помощью сервера через эти узлы можно читать и записывать значения точек любого узла проекта, даже если они запущены на других компьютерах или контроллерах.

*Примечание. Рекомендуется добавить в проект узел **Сервер данных** и запустить его на одном компьютере с OPC сервером. Значения точек будут автоматически передаваться с контроллеров (включая контроллеры ВСЕ-5 и контроллера Linux) на узел **Сервер данных**, а с него в OPC сервер. Команды управления будут автоматически передаваться в обратном порядке.*

Для каждой точки каждого узла сервер показывает тег, имя которого имеет формат

```
<Проект> - <Узел>\<Точка>.
```

Внимание. При завершении работы узла точки этого узла удаляются из списка тегов сервера и становятся недоступными.

По умолчанию сервер показывает только теги для тех точек узла, уровень передачи которых, заданный свойством **Уровень передачи сигнала**, больше или равен уровню, заданному параметром **InfoLevel** в файле *srvopcda.ini* в папке установки Каскад-САУ.