

**Инструкция по установке и эксплуатации
(пользовательская инструкция)
для программы RailCAD**

Содержание

1	Введение.....	5
1.1	Терминология.....	5
1.2	О системе RailCAD.....	6
1.3	Операционная система и программное обеспечение.....	7
2	Установка и Запуск	7
2.1	Установка приложения	7
2.2	Запуск приложения.....	8
2.3	Установка рабочих директорий	8
2.4	Выбор темы графического оформления.....	8
3	Запуск и выход из системы	9
3.1	Включение компьютера.....	9
3.2	Вход в систему Windows.....	9
3.3	Запуск системы RailCAD.....	10
3.3.1	Автоматический	10
3.3.2	С Рабочего Стола.....	10
3.4	Выход из системы RailCAD	10
3.5	Выход из системы Windows	10
4	Работа с системой RailCAD.....	10
4.1	Основное окно приложения.....	10
4.2	Настройка отображения компонентов основного окна.....	11
4.3	Страница приветствия.....	12
4.4	Главная панель инструментов.....	13
4.5	Панель инструментов настройки проекта.....	13
4.6	Панель инструментов для работы с объектами в графическом редакторе.....	14
4.7	Графический редактор станции	14
4.8	Дерево проекта.....	16
5	Создание и загрузка проекта.....	17

5.1	Создание нового проекта.....	18
5.2	Создание нового проекта на основе существующих данных (импорт).....	22
5.3	Загрузка существующего проекта	23
6	Настройка проекта	24
6.1	Настройка IPU типов (ADMGENTAB)	24
6.2	Компоненты проекта	26
6.3	Типы контроллеров	27
6.3.1	Список типов контроллеров проекта	28
6.3.2	Префикс типа контроллера.....	29
6.3.3	Общие параметры типа: соответствие контактов IPU типам	29
6.3.4	Import number для контактов.....	31
6.3.5	Маски типов контроллеров и IPU объектов	32
6.4	Параметры импорта из object_list	34
6.5	Редактор скриптов валидации	34
7	Добавление станций	35
8	Автоматизированный ввод данных (базовое заполнение IntData, OSdata)	36
8.1	Импорт данных о логических объектах из Visio.....	37
8.2	Импорт данных проектировщиков из object_list.....	37
8.3	Автоматическое связывание IPU объектов с переменными логических объектов.....	39
9	Работа с логическими объектами	42
9.1	Графический редактор графа станции	42
9.1.1	Добавление, редактирование, удаление логических объектов	43
9.1.2	Графическое редактирование объектов (вращение, отражение)	44
9.1.3	Графическое редактирование связей логических объектов	45
9.1.4	Связывание IPU объекта с группой логических объектов.....	46

9.2	Корректировка параметров логических объектов.....	47
9.2.1	Переименование логического объекта.....	48
9.2.2	Биты индивидуализации.....	48
9.2.3	Связывание freewired переменных.....	49
9.2.4	Связывание с IPU объектами (status, order переменные).....	50
9.2.5	Присвоение, удаление COS номеров логическим объектам.....	51
9.2.6	Связывание индикационных переменных с LOGCOS объектами.....	51
9.2.7	Автоматическое связывание переменных с текущим LOGCOS объектом.....	52
9.2.8	Валидация логических объектов.....	52
10	Работа с IPU объектами.....	53
10.1	Добавление, редактирование, удаление IPU объекта.....	54
10.2	Отображение связанных логических объектов в графическом редакторе.....	57
11	Работа с конвертерами.....	58
12	Работа с контроллерами.....	59
12.1	Добавление, корректировка, удаление контроллеров.....	59
12.2	Создание IPU объектов из формы контроллера.....	61
12.3	Автоматическая генерация имен IPU объектов из формы контроллера.....	62
13	Сохранение проекта.....	63

1 Введение

RailCAD (Logic Edition) - Система автоматизированного проектирования железнодорожных станций, предназначена для автоматизации проектирования и подготовки файлов данных и графических схем, используемых для адаптации под конкретную станцию.

RailCAD (Logic Edition) предоставляет пользователю возможность редактирования и автоматизированного формирования файлов адаптации (IntData, OCdata, ADMGENTAB). Редактирование и корректировка всех перечисленных файлов данных осуществляется через интерфейс приложения и не зависит от каких-либо сторонних библиотек и приложений, не входящих в стандартный дистрибутив программы.

1.1 Терминология

Виджет – графический модуль программного приложения.

ЛКМ – левая клавиша мыши.

Маска – общее правило, по которому могут быть определены или должны быть указаны имена различных объектов (например, имена контроллеров, *IPU* объектов, переменных).

ОК – объектный контроллер.

ПКМ – правая клавиша мыши.

Тулбар (панель инструментов) - элемент графического интерфейса пользователя, предназначенный для размещения на нём нескольких других элементов.

IntData – файл Interlocking_data (файл данных уровня логики).

OCData – файл Object_controller_system_data (файл данных уровня контроллеров и конвертеров).

CommandTable - файл Command_table (файл данных уровня команд).

Метаданные – файлы данных, содержащие описание возможных типов объектов и сущностей данных редактируемых и создаваемых в RailCAD:

- **sternol.ste** и **VariableDescription.txt** - содержат описания типов логических объектов, из которых состоит станция.

Расположены в следующей директории *//depot/eqv/adapt/ILL2_RF_StationName/.../implementation/sernal/out*, где *StationName* – имя станции.

- **admgentab** – содержит описания возможных типов IPU-объектов. Данная информация специфична для каждой станции и расположена в следующей директории: *//depot/eqv/adapt/ILL2_RF_StationName/.../interface*, где *StationName* – имя станции.
- **cosgentab** - содержит описания возможных типов команд. Данная информация специфична для каждой станции и расположена в следующей директории: *//depot/eqv/adapt/ILL2_RF_StationName/.../interface*, где *StationName* – имя станции.
- **TelegramDescription.txt**- содержит описание одного типа контроллера, имя которого и версию можно определить, исходя из имени директории, в которой лежит данный файл. Описания контроллеров хранятся в директориях *//depot/eqv/adapt/OC_ControllerName*, где *ControllerName* – имя контроллера. Для каждой версии контроллера существует отдельная директория

1.2 О системе RailCAD

Данное программное обеспечение предназначено для автоматизированного создания файлов данных, используемых для адаптации под конкретную станцию.

Целью внедрения данной разработки является обеспечение программного обеспечения, функционирующего под ОС Windows, позволяющего редактировать все необходимые для адаптации файлы, обладающего интуитивно понятным интерфейсом и не зависящего от каких-либо сторонних библиотек и приложений, не входящих в стандартный дистрибутив программы.

Все данные, используемые для адаптации можно разделить на следующие уровни: Hardware, Logic, Command. Каждый из этих уровней содержит, так называемые, метаданные и данные. К метаданным относятся данные, описывающие по определенным правилам способ построения данных. Так, например, для уровня Logic метаданными являются типы объектов, описанные на языке sternol, а данными – объекты, описываемые в файлах Interlocking_data, и построенные на основании типов объектов.

1.3 Операционная система и программное обеспечение

Программа RailCAD предназначена для работы под операционной системой Windows XP SP3 и более поздних версиях.

В настоящее время, первоначальное создание объектов и связей между ними для уровня Logic, происходит в MSVisio.

2 Установка и Запуск

2.1 Установка приложения

Для установки приложения **RailCAD (Logic Edition)** необходимо скачать последнюю версию дистрибутива по адресу: <http://develop.rusig.ru/>. Запустить исполняемый файл скачанного дистрибутива RailCADSetup.exe. В форме инсталлятора (рисунок 1) проверить и при необходимости откорректировать директорию установки приложения, затем нажать кнопку «Установить».

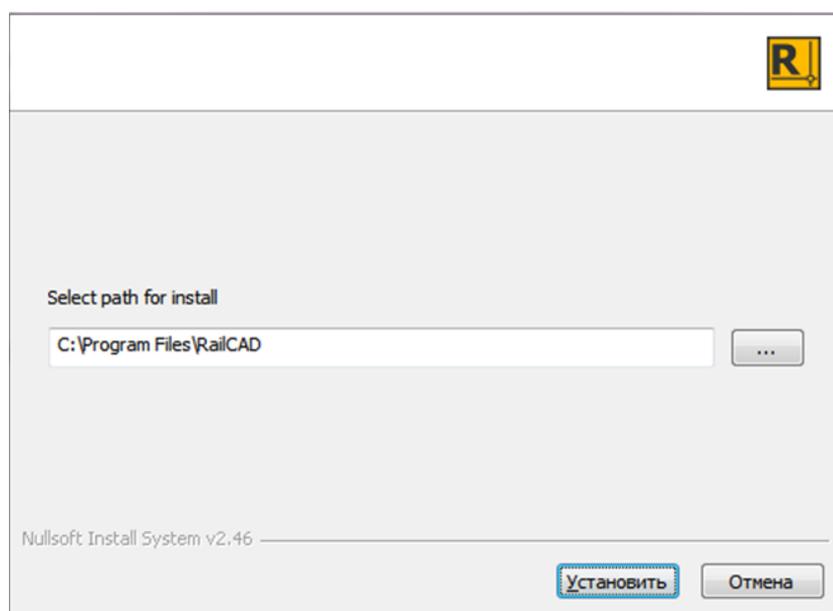


Рисунок 1 Форма инсталлятора

2.2 Запуск приложения

Запуск приложения осуществляется по двойному клику ЛКМ на ярлыке приложения  или по соответствующему файлу проекта (*.rcad) в проводнике. При первом запуске приложения, после его установки, необходимо указать рабочие директории.

2.3 Установка рабочих директорий

Установка рабочих директорий является обязательным условием работы приложения (рисунок 2). Выполняется при первом запуске RailCAD или корректируется пользователем в процессе использования приложения при нажатии кнопки  **Application settings** в главном тулбаре на вкладке **Common**.

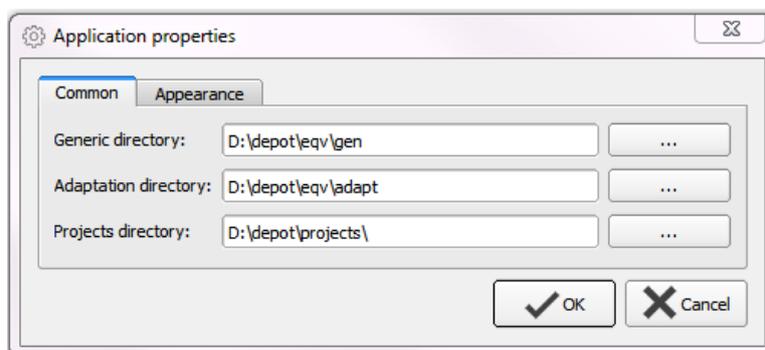


Рисунок 2 Установка рабочих директорий

В полях **Generic directory** и **Adaptation directory** необходимо указать расположение директорий, в которых хранятся файлы данных, необходимые для создания и работы проекта.

Project directory – директория, в которой по умолчанию будет создаваться новый проект и соответствующие ему файлы данных (IntData, OCdata и т.д.)

Имена директорий должны быть обозначены латинскими символами.

Для сохранения внесенных корректировок нажать кнопку **Ок**, для отмены – **Cancel**.

2.4 Выбор темы графического оформления

Выбор темы графического оформления приложения доступно в основном окне при нажатии кнопки **Настройки**  на вкладке

Appearance. Настройки приложения будут применены после его перезагрузки.

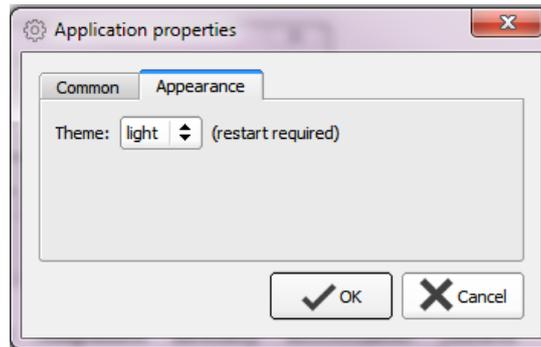


Рисунок 3 Выбор темы графического оформления

После установки рабочих директорий отобразится главное окно приложения и страница приветствия.

При запуске приложения из файла проекта в главном окне, будут отображены все данные загруженного проекта.

При запуске приложения из файла проекта ссылки на файлы данных (ILL, ADMGENTAB, ILS, Telegram Description) должны быть актуальными. Т.е. имена и расположение файлов данных, а также директорий их хранения должны соответствовать ConfigInfo.

В случае, если при запуске приложения файлы данных не найдены, отобразится сообщение с информацией об отсутствующих файлах, проект загружен не будет.

3 Запуск и выход из системы

3.1 Включение компьютера

Включите монитор кнопкой включения питания на передней панели. На включенном мониторе должен гореть светодиод контроля питания. Включите системный блок кнопкой включения питания на системном блоке. Светодиод контроля питания также должен загореться.

3.2 Вход в систему Windows

После непродолжительной загрузки появляется приглашение «Для входа в систему нажмите «Ctrl+Alt+Del», После нажатия этой комбинации клавиш на клавиатуре появляется диалог входа в систему, в котором необходимо ввести логин и пароль, а потом нажать «Enter».

3.3 Запуск системы RailCAD.

3.3.1 Автоматический

Если после входа в систему клиент RailCAD запустился самостоятельно,

значит, он добавлен в папку (директорию) для автоматического запуска, и пользователю ничего не придется делать.

3.3.2 С Рабочего Стола

На экране должен присутствовать ярлык приложения. Чтобы открыть его – щелкните по нему мышкой, произведя двойной щелчок левой кнопкой (быстро два раза нажав на левую кнопку мышки, наведенной на иконку). Программа немедленно запустится.

3.4 Выход из системы RailCAD

В меню главного окна программы нужно выбрать пункт «Выход». Заккрытие всех окон системы по одному стандартными средствами Windows не приведет к выходу из системы – последнее окно закрывается только выбором пункта «Выход».

3.5 Выход из системы Windows

Нажать кнопку «Пуск». В меню выбрать «Завершение работы». В появившемся диалоге выбрать пункт «Выключить компьютер». Нажать «Enter».

4 Работа с системой RailCAD

4.1 Основное окно приложения

После запуска RailCAD на экране отобразится главное окно приложения. Основные рабочие области окна составляют:

- страница приветствия;
- главная панель инструментов;
- панель инструментов настройки проекта;
- панель инструментов для работы с объектами проекта;
- дерево проекта;
- графический редактор станции.

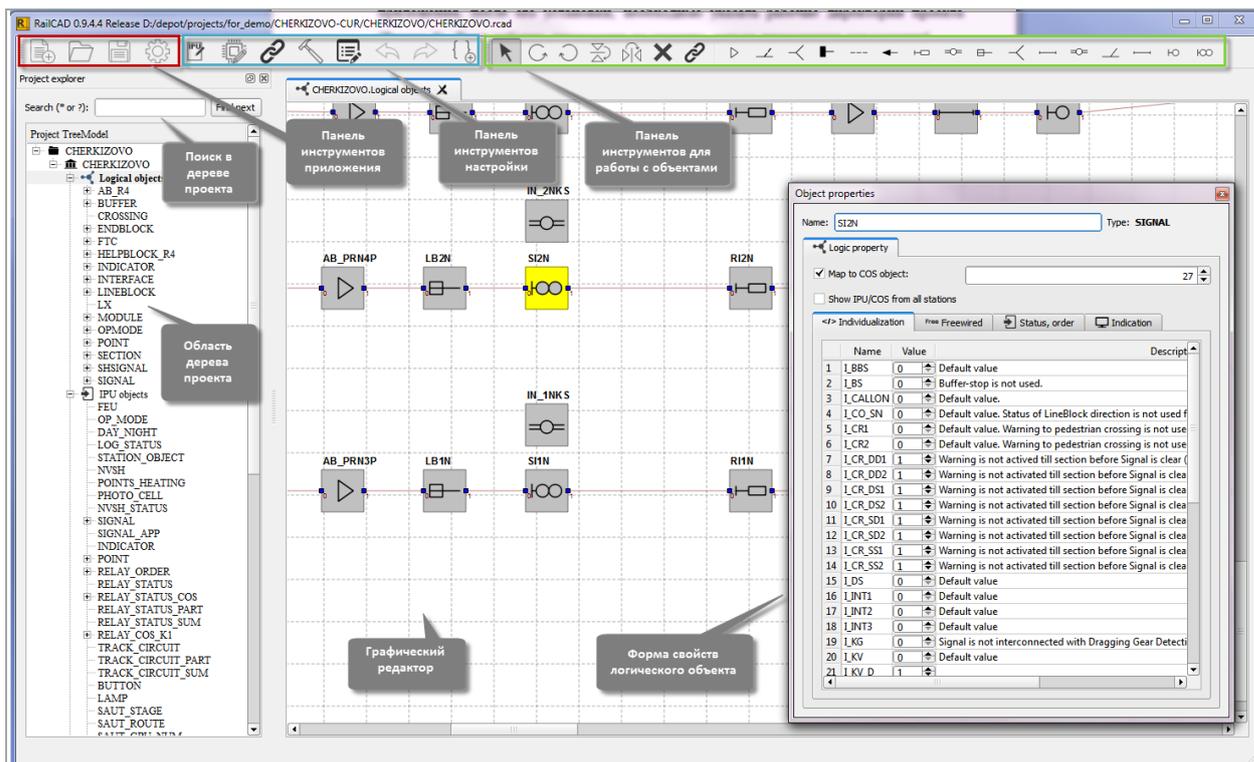


Рисунок 1 Основное окно приложения

4.2 Настройка отображения компонентов основного окна

Отображение компонентов основного окна доступно в контекстном меню любой панели инструментов (вызывается кликом правой клавиши мыши в области панели инструментов).

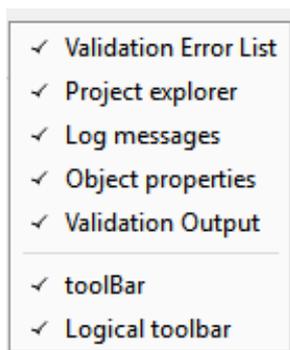


Рисунок 2 Контекстное меню настройки отображаемых компонентов

В контекстном меню отметить флагами те компоненты, которые необходимо отобразить; снять флаги, для тех элементов, которые отображать не нужно.

4.3 Страница приветствия

Страница приветствия отображается при каждом запуске приложения из ярлыка. Позволяет быстро начать (**Start project**) или продолжить (**Recent project**) работу с проектом. Также содержит ссылку на краткое описание по работе с приложением (**Quick guide**). Функции кнопок по созданию проекта описаны в разделе **Главная панель инструментов**.

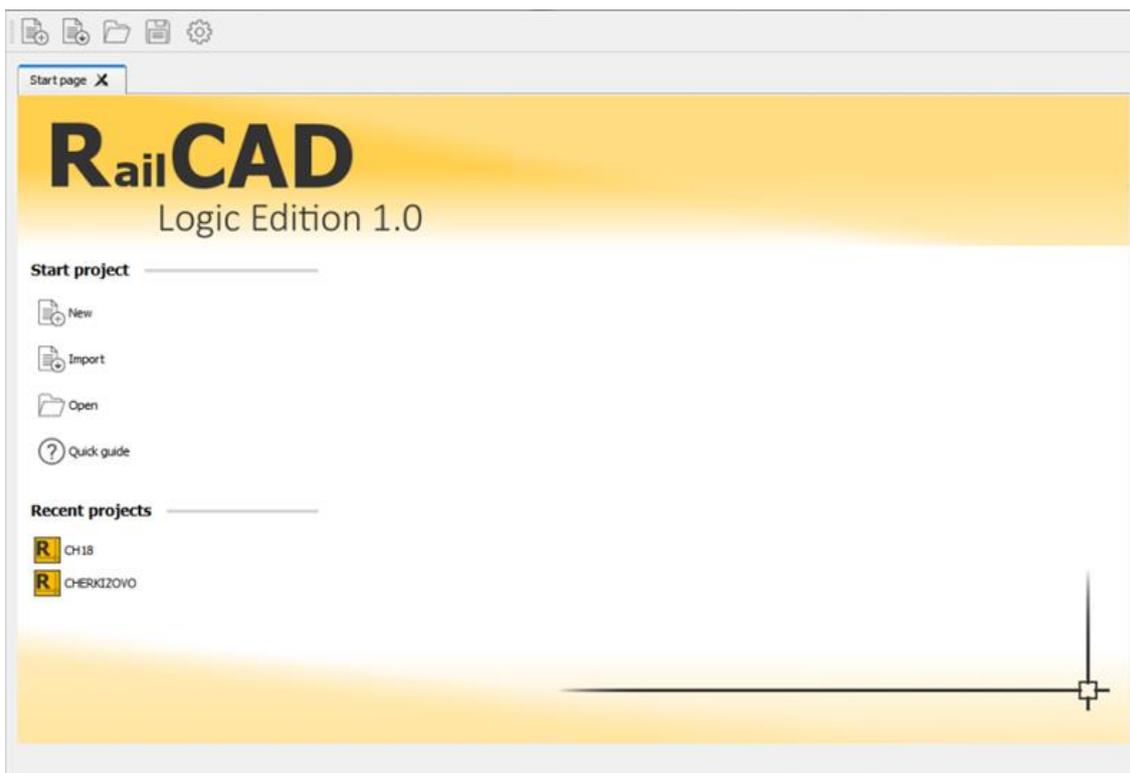


Рисунок 3 Страница приветствия

В перечне проектов в разделе **Recent project** можно удалить проект по действию **Remove** из контекстного меню, также можно перейти к месту расположения файла проекта (**Open Containing Folder**).

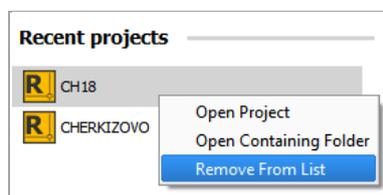


Рисунок 4 Недавно открытые проекты

4.4 Главная панель инструментов

Верхняя панель инструментов включает в себя функции по созданию и загрузке проекта, а также настройке приложения. Слева направо кнопки обозначают:



- создать новый проект;



- создать проект на основе существующих данных;



- открыть проект;



- сохранить проект;



- настройки приложения (рабочие директории).

4.5 Панель инструментов настройки проекта

Панель инструментов настройки проекта отображается после загрузки проекта. Включает в себя функции по настройке параметров проекта. Слева направо кнопки обозначают:



- редактор IPU типов (возможность редактировать ADMGENTAB);



- редактор перечня типов контроллеров;



- автоматическое связывание IPU объектов с переменными логических объектов;



- валидация логических проектов;



- редактор скриптов валидации;



- отменить выполнение действия;



- повторить выполнение действия;



- редактор компонентов проектов (возможность редактировать ConfigInfo).

4.6 Панель инструментов для работы с объектами в графическом редакторе

Панель инструментов для работы с объектами в графическом редакторе отображается после загрузки проекта и включает функции по работе с выбранными логическими объектами в графическом редакторе.

 - выбрать объект;

 - вращать выбранный объект направо (R);

 - вращать выбранный объект налево (Shift+R);

 - отразить объект по оси X (F);

 - отразить объект по оси Y (Shift+F);

 - удалить выбранный объект (Del);

 - связать выбранные логические объекты с IPU объектом.

4.7 Графический редактор станции

Графический редактор станции занимает большую часть рабочей области основного окна приложения. Визуально располагается в центральной части главного окна. Для каждой станции проекта отображается свой графический редактор, который можно закрыть при нажатии кнопки крестик в заголовке закладки редактора.

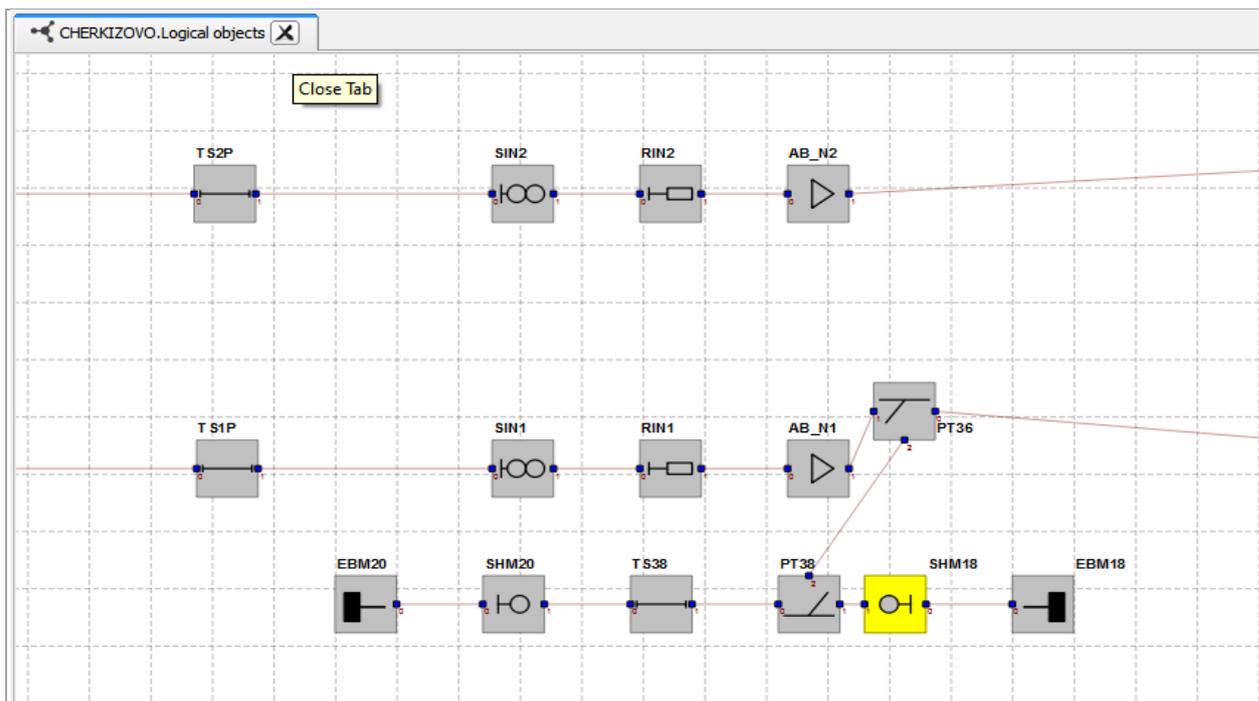


Рисунок 5 Графический редактор станции

Добавление логических объектов в область графического редактора выполняется при помощи перетаскивания иконок соответствующих типов с панели инструментов проекта в область редактора.

На панели инструментов станции отображаются иконки соответствующие типам логических объектов, которые могут быть добавлены в проект.

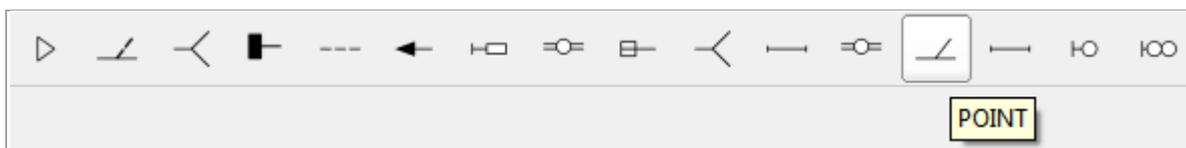


Рисунок 6 Панель инструментов станции

При клике правой кнопкой на логическом объекте станции отображается контекстное меню действий с логическим объектом. Перечень действий в контекстном меню аналогичен действиям на панели инструментов проекта.

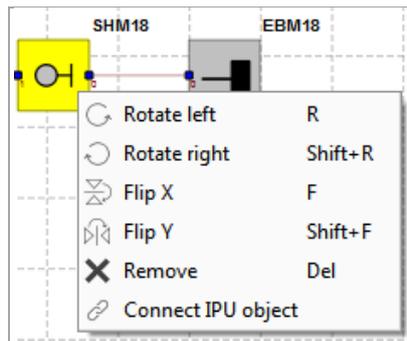


Рисунок 7 Контекстное меню объекта графического редактора

4.8 Дерево проекта

В дереве проекта отображается информация об объектах станции:

- **Logical objects** – логические объекты;
- **IPU objects** – IPU объекты;
- **Controllers** – контроллеры;
- **Converters** – конвертеры.

Каждый из вышеперечисленных уровней разделен на типы, в каждом типе отображаются отдельные экземпляры объектов.

Добавление, редактирование, удаление экземпляров объектов доступно на уровне типа объекта. Исключение - уровень логических объектов, где добавление объектов доступно в области графического редактора.

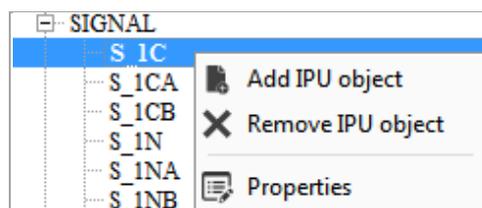


Рисунок 8 Контекстное меню в дереве проекта

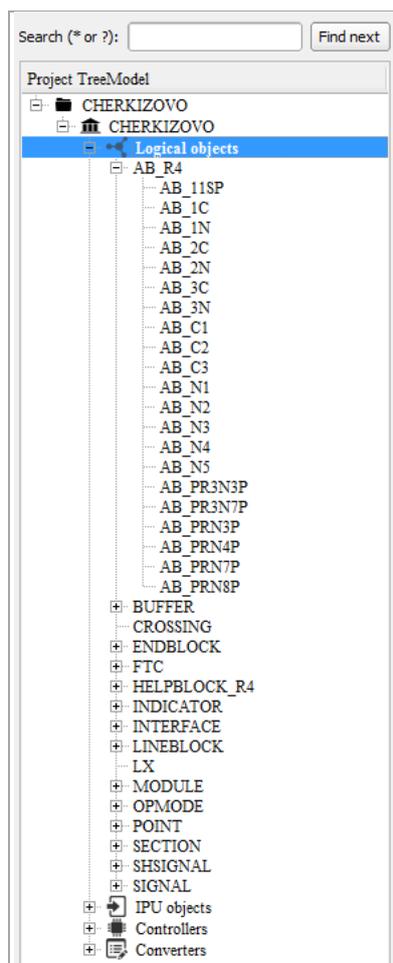


Рисунок 9 Дерево проекта

При двойном клике на экземпляр объекта открывается форма для его редактирования.

В дереве проекта предусмотрена возможность поиска объектов, контроллеров, конвертеров, всех возможных названий типов и имен станций с возможностью использования специальных символов (* и ?). Найденный в дереве проекта элемент выделяется жирным шрифтом.

5 Создание и загрузка проекта

После успешной установки приложения и рабочих директорий можно приступить к работе с программой: создавать новые проекты, загружать и редактировать существующие.

При создании проектов и последующей работе с ними необходима корректная и синхронизированная с **Perforce** структура файлов данных в рабочих директориях проекта.

Имена и расположение файлов данных (ILL, ADMGENTAB, ILS, Telegram Description) должны соответствовать ConfigInfo, иначе создание и загрузка проекта могут быть завершены с ошибкой.

5.1 Создание нового проекта

Для создания нового проекта необходимо нажать кнопку  **New project** в главном тулбаре приложения или на странице приветствия.

Создание проекта происходит в несколько этапов.

Этап 1. Выбор шаблона и базовых параметров проекта

В форме создания нового проекта New project выбрать шаблон, по которому будет создан проект; указать имя создаваемого проекта (Name) и директорию его расположения (Location) (по умолчанию подставится базовая директория из настроек приложения). Нажать кнопку Next >.

Шаблоны создания проекта отличаются базовым набором информации по проекту: перечень типов контроллеров, ILL, СВИ, скриптами валидации для логических объектов и т.д.

Для формирования новых и корректировки используемых шаблонов необходимо обратиться к разработчикам RailCAD.

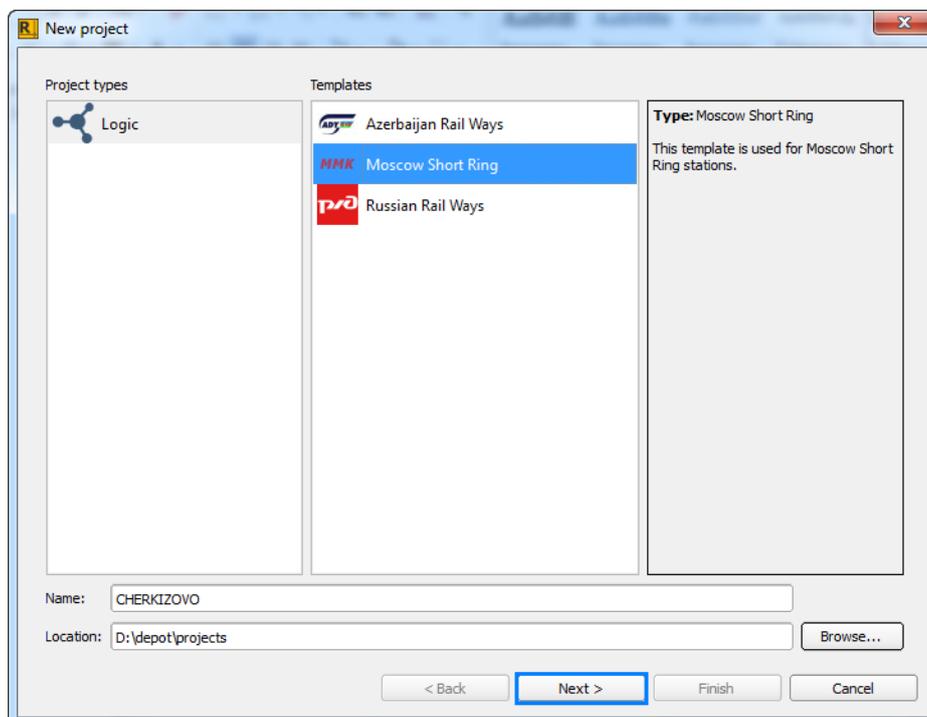
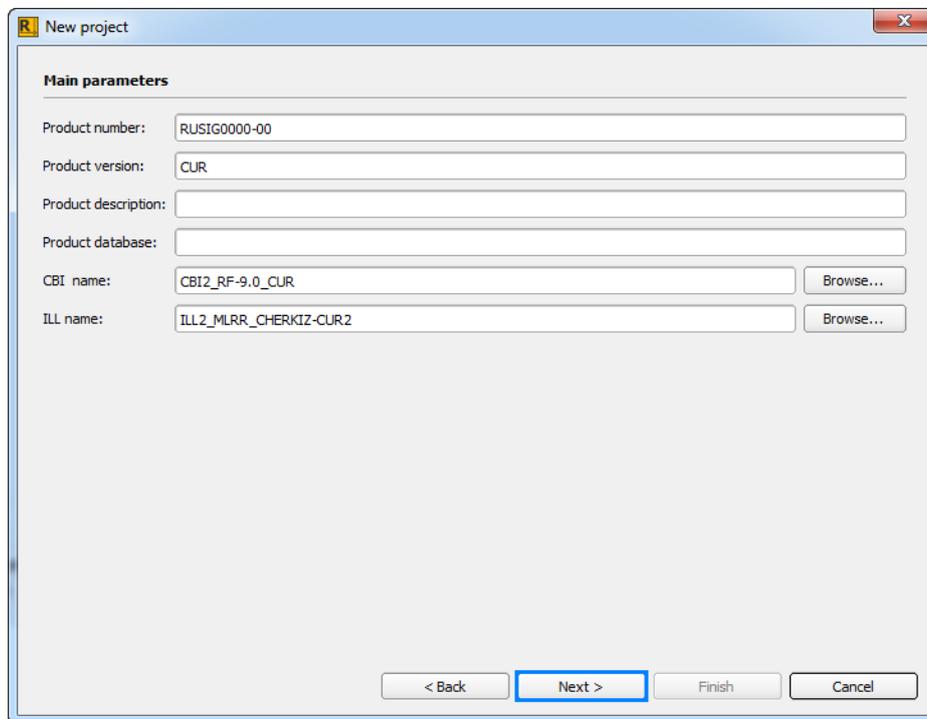


Рисунок 14 Выбор шаблона создания проекта

Этап 2. Main parameters

В форме «New project - Main parameters»: заполнить и/или откорректировать общие данные проекта; CBI name, ILL name, выбрав соответствующей ConfigInfo. Нажать кнопку Next >.



The screenshot shows a window titled "New project" with a "Main parameters" section. The fields are as follows:

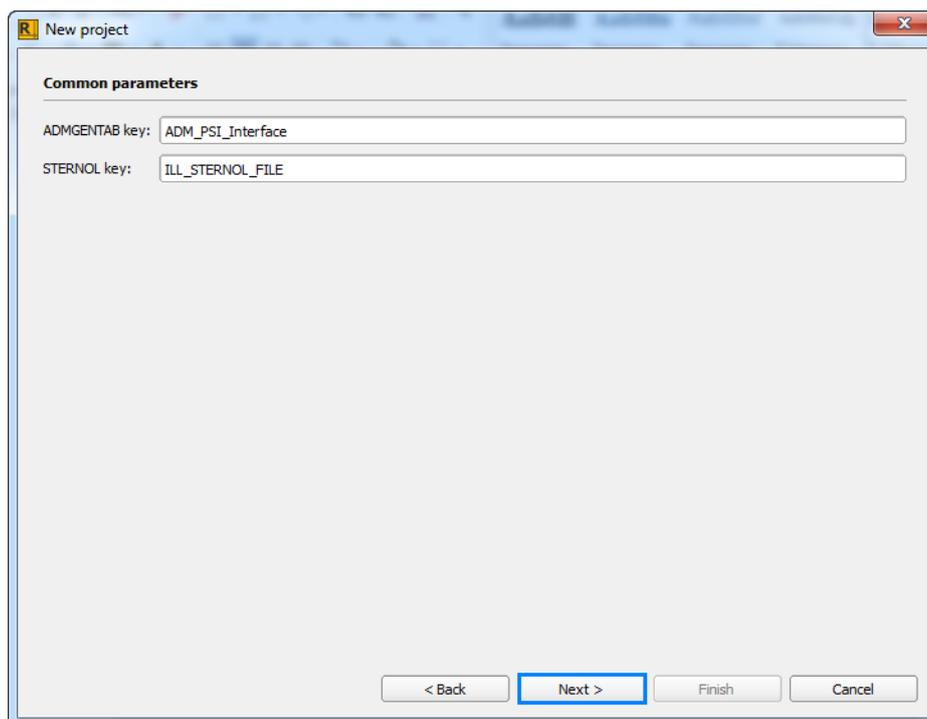
Field	Value
Product number:	RUSIG0000-00
Product version:	CUR
Product description:	
Product database:	
CBI name:	CBI2_RF-9.0_CUR
ILL name:	ILL2_MLRR_CHERKIZ-CUR2

Buttons at the bottom: "< Back", "Next >" (highlighted with a blue border), "Finish", and "Cancel".

Рисунок 105 Создание проекта. Основные параметры

Этапы 3 – 5. Заполнение общей информации файлов данных

Оставить шаблонные данные без изменения или откорректировать данные вручную. На формах 3 и 4 нажать кнопку Next >. На форме 5 нажать кнопку Finish для завершения создания проекта.

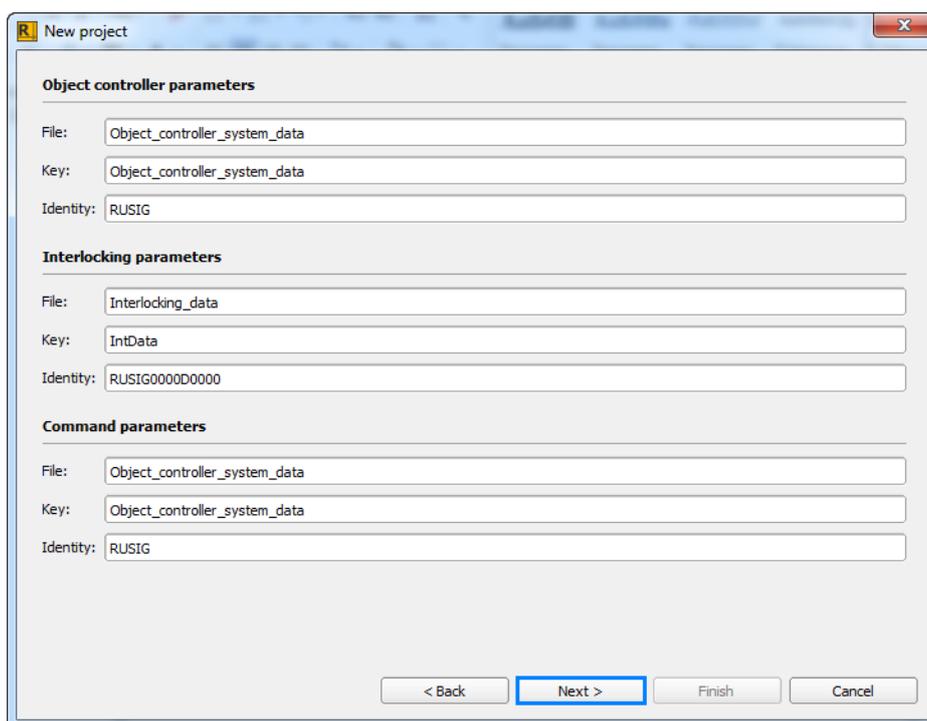


The screenshot shows a 'New project' dialog box with the following fields:

- Common parameters**
- ADMGENTAB key: ADM_PSI_Interface
- STERNOL key: ILL_STERNOL_FILE

Navigation buttons at the bottom: < Back, Next > (highlighted), Finish, Cancel.

Рисунок 116 Создание проекта. Общие параметры



The screenshot shows a 'New project' dialog box with the following sections and fields:

- Object controller parameters**
 - File: Object_controller_system_data
 - Key: Object_controller_system_data
 - Identity: RUSIG
- Interlocking parameters**
 - File: Interlocking_data
 - Key: IntData
 - Identity: RUSIG0000D0000
- Command parameters**
 - File: Object_controller_system_data
 - Key: Object_controller_system_data
 - Identity: RUSIG

Navigation buttons at the bottom: < Back, Next > (highlighted), Finish, Cancel.

Рисунок 127 Создание проекта. Параметры файлов данных

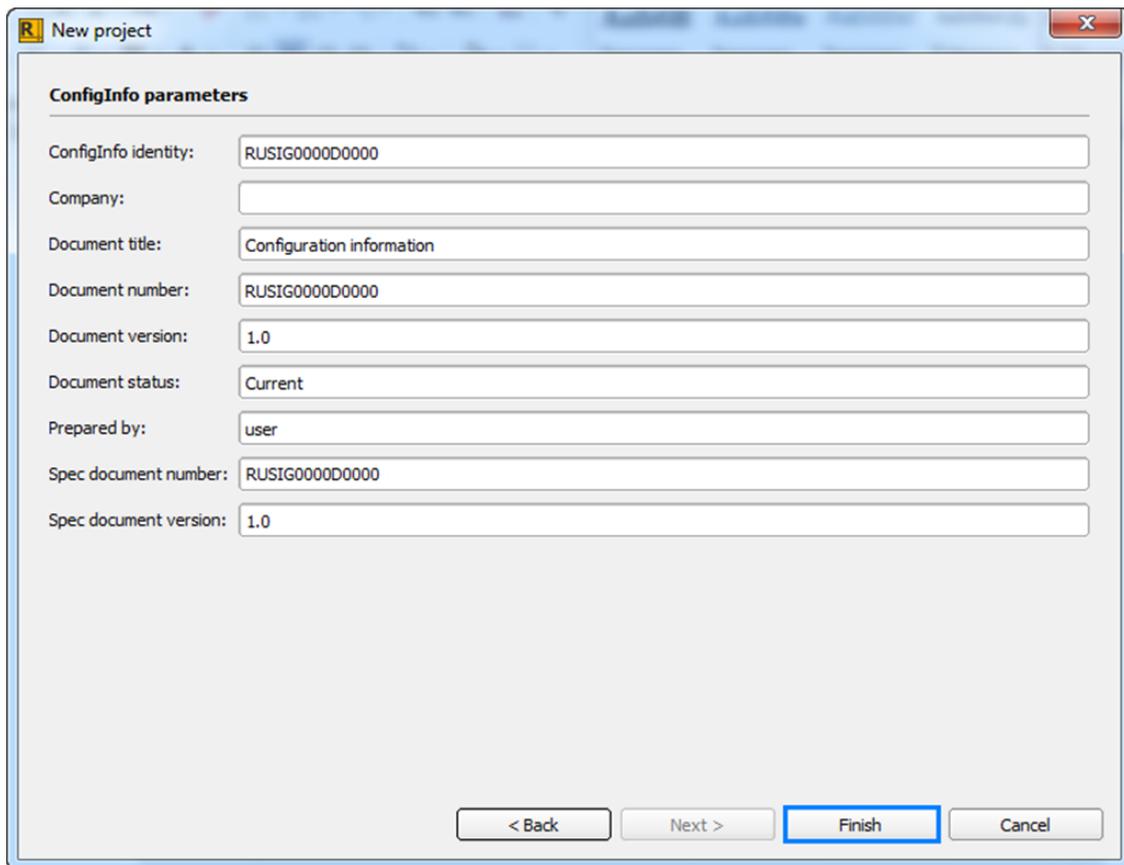


Рисунок 138 Создание проекта. Параметры ConfigInfo

Если все файлы данных, указанные при создании проекта найдены в рабочих директориях приложения, создание проекта будет успешно завершено:

в рабочей директории проекта создадутся файлы данных: файл проекта (*.rcad), INTdata, OCdata, ConfigInfo, файл со скриптами валидации;

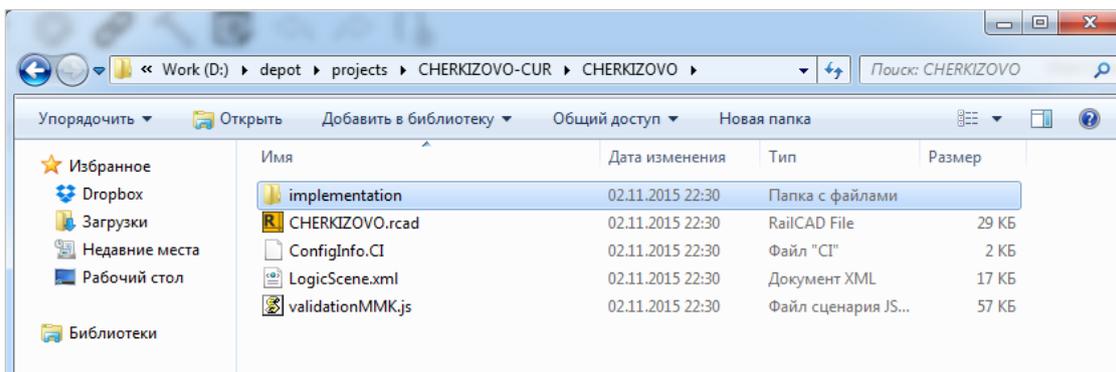


Рисунок 19 Файлы данных, сгенерированные в результате создания проекта

- в главном окне приложения в области дерева проекта отобразится имя созданного проекта;

- пользователю доступно редактирование настроек проекта, можно добавлять станции и вносить данные.

5.1.1 Создание нового проекта на основе существующих данных (импорт)

Для создания проекта на основе существующих данных необходимо нажать кнопку  **Import** в главном тулбаре приложения или на странице приветствия.

1. В форме импорта проекта выбрать шаблон для импорта и **ConfigInfo file**, по которому автоматически определятся все файлы данных для импорта: IntData, OCdata, ADMGENTAB, sternol. Файл проекта будет создан в директории выбранного ConfigInfo. Нажать кнопку **Next>**.

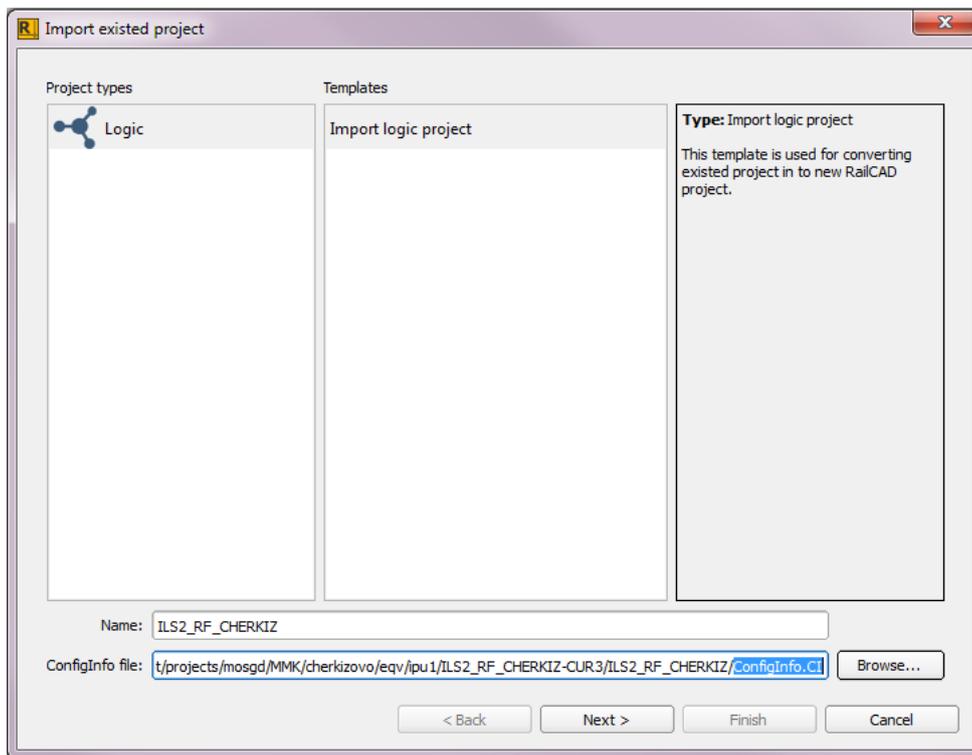


Рисунок 140 Создание проекта на основе существующих данных

2. На следующих этапах импорта можно откорректировать базовые параметры (ключи фалов данных, используемых в проекте). Нажать кнопку **Next >**.
3. На завершающем этапе указать одну или несколько станций, для распределения по ним импортируемых данных.

В проекте может быть создана одна станция без постфикса или множество станций с постфиксами.

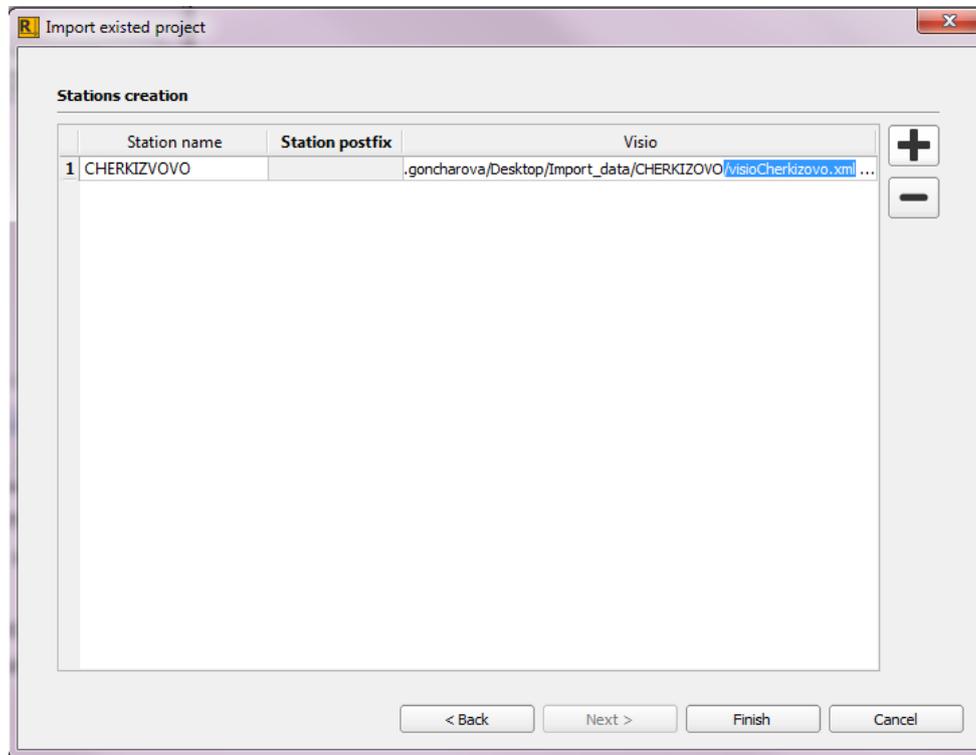


Рисунок 151 Выбор файла для импорта координат графа станции

Если для импортируемого проекта существует файл **Visio** с планом станции, его также можно импортировать в **RailCAD**: для добавленной станции выбрать файл *.xml, выгруженный из Visio при помощи Visio-Plugin. Нажать кнопку **Next>**.

Если при импорте проекта файл **Visio** не указан, то данные будут импортированы по базовым координатам проекта.

5.2 Загрузка существующего проекта

Загрузка существующего проекта выполняется:

- при нажатии кнопки  **Open** на главной панели инструментов или на стартовой странице приложения;
- при выборе недавно открытых проектов на стартовой странице приложения в списке **Recent projects**;
- двойным кликом левой клавишей мыши на файле проекта в директории его хранения.

Условия необходимые для корректной загрузки проекта аналогичны созданию проекта.

6 Настройка проекта

В типовом случае все базовые параметры проекта заполняются автоматически при создании проекта на основе выбранного шаблона.

Пользователь может самостоятельно откорректировать параметры проекта:

- IPU типы (данные из ADMGENTAB);
- типы контроллеров;
- компоненты проекта;
- скрипты валидации;
- правила связывания IPU объектов с переменными логических объектов.

Также, в проекте можно изменить базовые параметры импорта, необходимые для автоматизированного ввода данных.

6.1 Настройка IPU типов (ADMGENTAB)

Настройка IPU типов выполняется в редакторе и включает возможности:

- редактирования перечня и свойств IPU типов (ADMGENTAB) (1 и 2);
- редактирования префиксов IPU типов (3) (для импорта из object_list).

Редактор IPU типов открывается по кнопке  **IPU types** в тулбаре проекта. В отобразившейся форме редактора перечень IPU типов и общая информация загружается автоматически из файла данных ADMGENTAB.

Если ADMGENTAB, используемый в проекте недоступен для редактирования (read only), то в редакторе IPU типов будет доступна только корректировка префиксов.

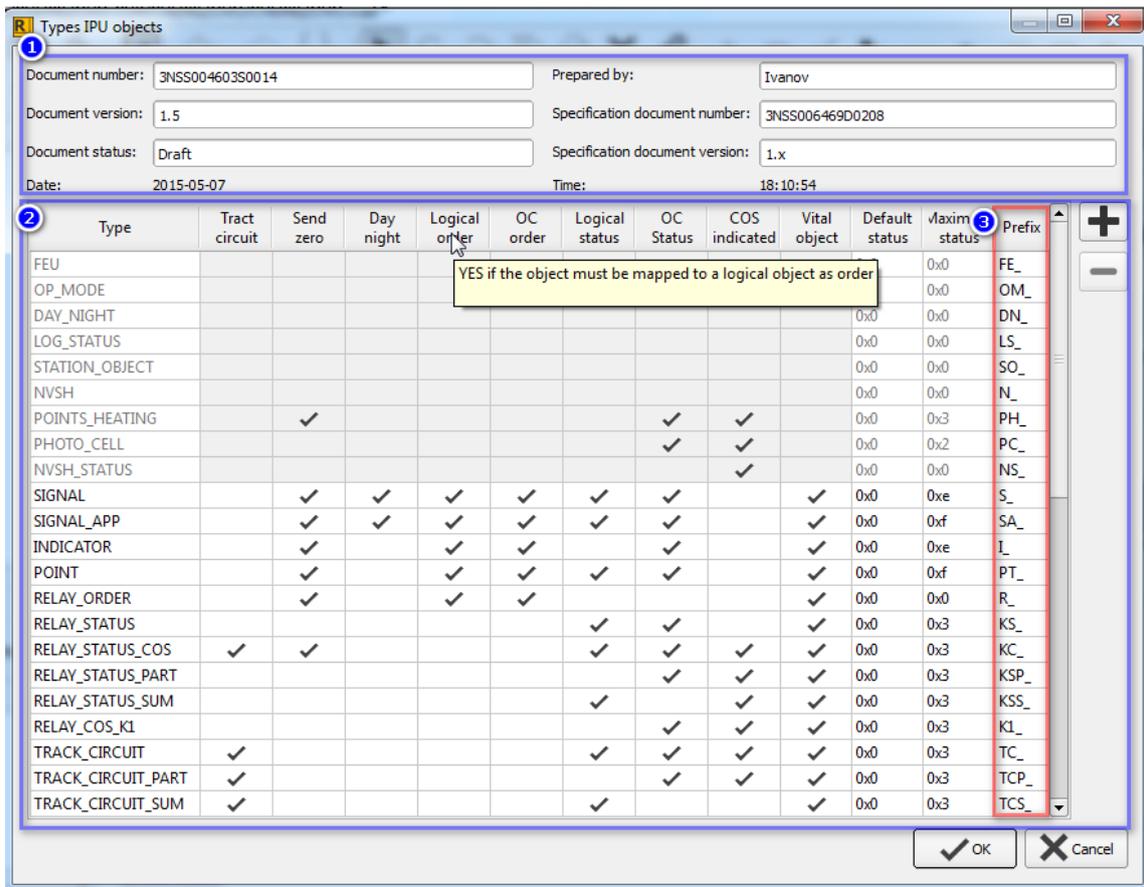


Рисунок 162 Редактор IPU типов

В редакторе IPU типов можно корректировать:

- общую информацию о IPU типах (1);
- перечень типов: добавить/ удалить/ переименовать IPU тип (2);
- свойства IPU типа (2) - выставить флаг, соответствующий значению YES; снять, если значение свойства NO; при наведении курсора на заголовок колонки свойства, отображается всплывающая подсказка с описанием свойства.

Если для IPU типа указано свойство «COS indicated» - YES, то в форме IPU объекта данного типа доступно заполнение IPUCOS номера.

Флаг «COS indicated» снять невозможно, если к IPU типу привязаны IPU объекты.

Корректировка и удаление IPU типов связанных с IPU объектами и типами контроллеров невозможна.

- **префиксы IPU типов (3)** - выбрать тип, в колонке **Prefix** указать или удалить префикс типа. Префиксы базовых типов

определяются автоматически при создании проекта (на основе шаблона).

Import design data

Префиксы IPU типов используются при импорте данных из object_list.

Префикс IPU типа должен соответствовать префиксу IPU объекта, импортируемого из object_list.

Для сохранения внесенных корректировок в редакторе IPU типов (Types IPU objects) нажать кнопку Ok, для отмены – Cancel.

6.2 Компоненты проекта

Перечень компонентов проекта определяется списком "**~/reference**" в **ConfigInfo** проекта.

Редактор компонентов открывается по кнопке  (**ConfigInfo's component editor**) в тулбаре проекта. В форме редактирования компонентов можно удалить используемые компоненты или добавить новые.

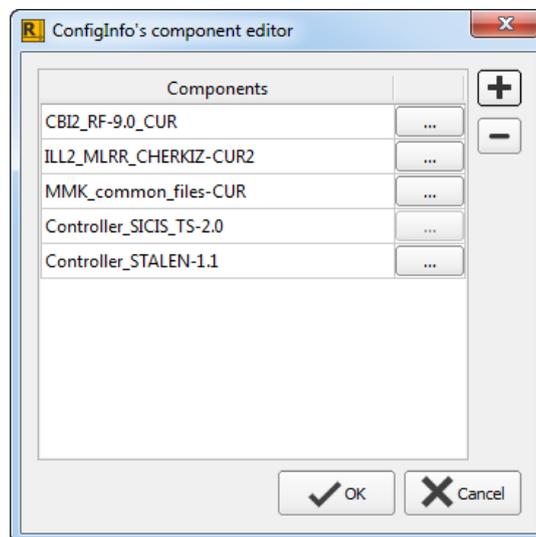


Рисунок 173 Редактор компонентов

Добавление компонента осуществляется при выборе файла **ConfigInfo.CI** из соответствующей директории (Generic directory/Adaptation directory).

Удалить компонент, являющийся типом контроллера невозможно (необходимо сначала удалить из проекта тип контроллера).

При нажатии кнопки **Ok** в реакторе компонентов файл данных **ConfigInfo** проекта будет автоматически перезаписан (до сохранения проекта). **Cancel** - отменить корректировки.

6.3 Типы контроллеров

Для добавления в проект экземпляров контроллеров необходимо сформировать перечень типов, по которым будет доступен ввод данных. При описании типов контроллеров используется информация о возможных индивидуализациях и соответствующих им контактах (данные из Telegram Description).

Редактор типов контроллеров позволяет откорректировать:

- список типов контроллеров;
- параметры типа:
 - общие параметры: соответствие контактов IPU типам;
 - для импорта данных: prefix, import number;
 - для автоматической генерации IPU объектов из формы контроллера: маски типа и контактов.

Редактор типов контроллеров открывается по кнопке  (**Controllers types**) в тулбаре проекта.

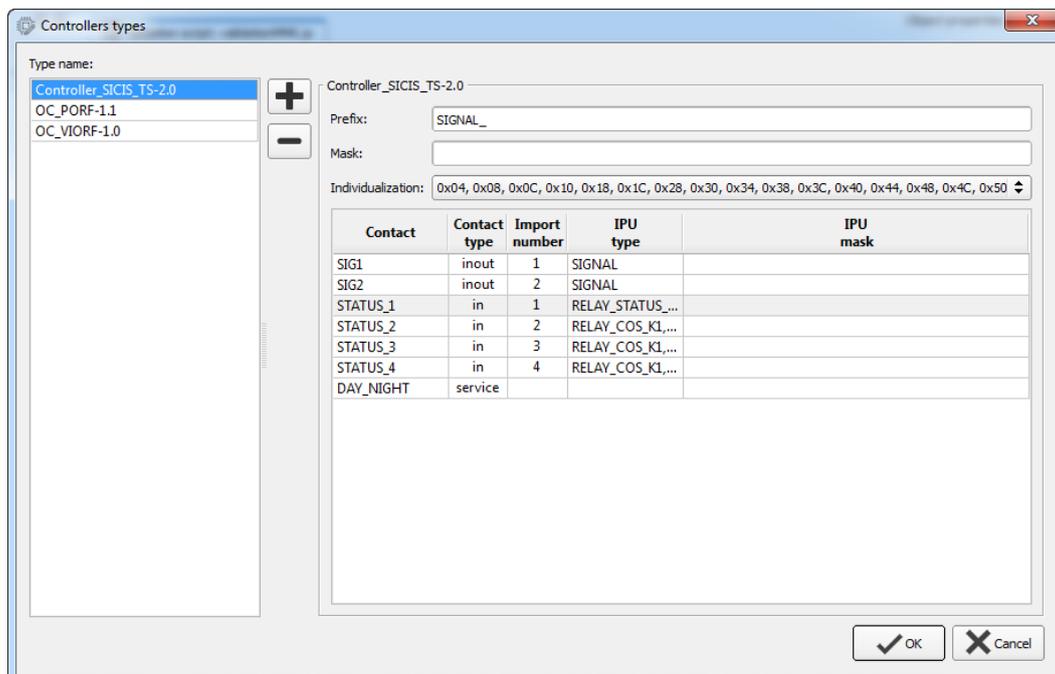


Рисунок 184 Редактор типов контроллеров

6.3.1 Список типов контроллеров проекта

Базовый список типов контроллеров проекта определяется шаблоном проекта.

Редактирование списка типов контроллеров доступно в форме **Controllers types** (открывается при нажатии на кнопку ) в списке **Type name:**

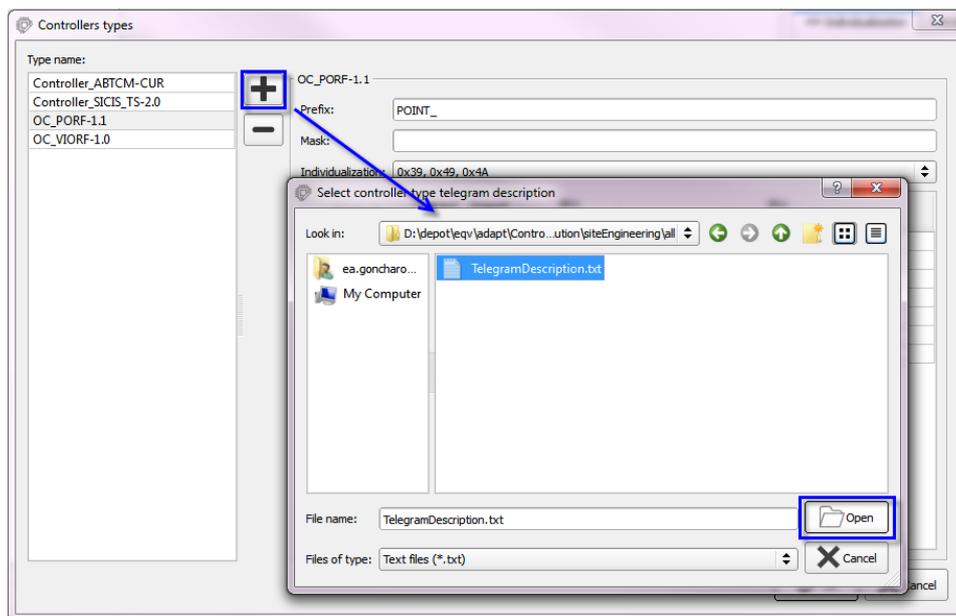


Рисунок 195 Добавление типа контроллера

-  - добавить тип контроллера: из рабочей директории (generic или adaptation directory) выбрать TelegramDescription нужного типа контроллера. Добавленный тип контроллера отобразится в списке типов, для него автоматически, в соответствии с TelegramDescription, определится перечень индивидуализаций и контактов, а также тип контактов (in, out, inout, service).

Если формат необходимого файла **TelegramDescription** отсутствует, то для отображения его в списке доступных для выбора файлов в выпадающем списке **Files of type:** выбрать значение **All files (*)**.

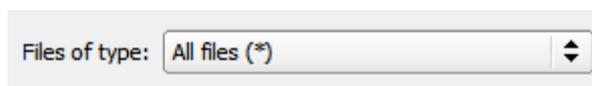


Рисунок 206 Выбор типа отображаемого файла

-  - удалить из списка выбранный тип контроллера.

- **Ok** – сохранить корректировки; откорректированный список отобразится в дереве проекта.
- **Cancel** - отменить корректировки.

Тип контроллера должен содержаться в списке компонентов приложения (**ConfigInfo**).

Также для используемых в проекте типов контроллеров необходимо заполнить параметры импорта, соответствие контактов IPU типам и настройки масок.

6.3.2 Префикс типа контроллера

Префикс (**Prefix**) типа контроллера используется при импорте экземпляров контроллеров из `object_list`. На основании префикса определяется, к какому типу относится импортируемый контроллер.

Для редактирования префикса в редакторе типов выбрать необходимый тип, далее указать или удалить префикс типа.



Рисунок 217 Редактирование префикса типа контроллера

Префикс типа контроллера должен соответствовать префиксу экземпляра контроллера, импортируемого из `object_list`.

6.3.3 Общие параметры типа: соответствие контактов IPU типам

Для базовых типов контроллеров соответствие контактов и IPU типов определяется шаблоном проекта.

Настройка соответствий контактов контроллера IPU типам является общим параметром для создания и импорта экземпляров контроллеров, а также для генерации IPU объектов из формы контроллера.

Если для контактов контроллера не указаны соответствия IPU типам, то контакты экземпляра контроллера невозможно связать с IPU объектами при его создании и импорте из `object_list`.

	Field name	IPU object	IPU type
1	ORDER_R1	R_2CSN	RELAY_ORDER
2	ORDER_R2	R_2CSNM	RELAY_ORDER
3	ORDER_R3		
4	ORDER_R4		
5	Dummy		
6	STATUS_1		
7	STATUS_2		
8	STATUS_3		
9	STATUS_4		

Для контакта в настройках типа контроллера не указано соответствие IPU типу, привязать IPU объект невозможно.

Рисунок 228 Форма контроллера. Связывание контактов с IPU объектами

Редактирование соответствий контактов контроллеров IPU типам доступно в форме **Controllers types** (открывается при нажатии на кнопку ).

В форме редактирования типов **Controllers types**:

Controller_SICIS_TS-2.0

Prefix:

Mask:

Individualization:

Contact	Contact type	Import number	IPU type	
SIG1	inout	1	SIGNAL	<input type="checkbox"/>
SIG2	inout	2	SIGNAL	<input type="checkbox"/>
SIG3	inout	3	SIGNAL	<input type="checkbox"/>
SIG4	inout	4	SIGNAL	<input type="checkbox"/>
STATUS_1	in	1	RELAY_COS_K1, RELAY_STATUS_COS	<input type="checkbox"/>
STATUS_2	in	2	RELAY_COS_K1, RELAY_STATUS_COS	<input type="checkbox"/>
STATUS_3	in	3	RELAY_COS_K1, RELAY_STATUS_COS	<input type="checkbox"/>
STATUS_4	in	4	RELAY_COS_K1, RELAY_STATUS_COS	<input checked="" type="checkbox"/>
DAY_NIGHT	service			<input type="checkbox"/>

FEU
 OP_MODE
 DAY_NIGHT
 LOG_STATUS
 STATION_OBJECT
 NVSH
 POINTS_HEATING
 PHOTO_CELL
 NVSH_STATUS
 SIGNAL
 SIGNAL_APP
 INDICATOR
 POINT

Рисунок 2923 Редактирование соответствий контактов контроллеров IPU типам

- для типа контроллера выбрать группу индивидуализаций, для которой необходимо настроить соответствия;
- в колонке IPU type в строке с нужным контактом в выпадающем списке флагами отметить те IPU типы, объекты которых могут быть привязаны к контактам. Для контакта может быть одно или более соответствий IPU типам;

- **Ok** – сохранить корректировки;
- **Cancel** - отменить корректировки.

6.3.4 Import number для контактов

На основании **Import number** происходит автоматическое связывание контактов с IPU объектами при **импорте данных** из object_list. Если **Import number** не заполнен, импорт информации об IPU объектах невозможен.

Для редактирования **Import number** в редакторе типов контроллеров  выбрать необходимый тип, далее указать или удалить номер для импорта. **Import number** должен быть уникален в пределах группы контакта: **in, out, inout, service**.

Contact	Contact type	Import number	IPU type
SIG1	inout	1	SIGNAL
SIG2	inout	2	SIGNAL
SIG3	inout	3	SIGNAL
SIG4	inout	4	SIGNAL
STATUS_1	in	1	RELAY_COS_K1, RELAY_STATUS_COS
STATUS_2	in	2	RELAY_COS_K1, RELAY_STATUS_COS
STATUS_3	in	3	RELAY_COS_K1, RELAY_STATUS_COS
STATUS_4	in	4	RELAY_COS_K1, RELAY_STATUS_COS
DAY_NIGHT	service		

Рисунок 240 Редактирование Import number для контактов типа контроллера

Для корректной привязки IPU объектов к контактам контроллера необходимо, чтобы **Import number** контакта в **RailCAD** соответствовал номеру контакта в object_list.

Пример 1:

Импорт данных по контроллеру SIGNAL_1CA_2CA_3CA (номер в object_list 6185).

objectlist		RailCAD	
IPU объект	Контакт	Контакт	Import number
S_1CA	6185_1	SIG1	1
S_2CA	6185_2	SIG2	2
S_3CA	6185_3	SIG3	3

Контакты и импортируемые в **RailCAD** IPU объекты будут связаны по найденным соответствиям номеров контактов (как указано в таблице).

Если для контакта указан **Import number**, соответствие контакта **IPU типу** также должно быть заполнено.

6.3.5 Маски типов контроллеров и IPU объектов

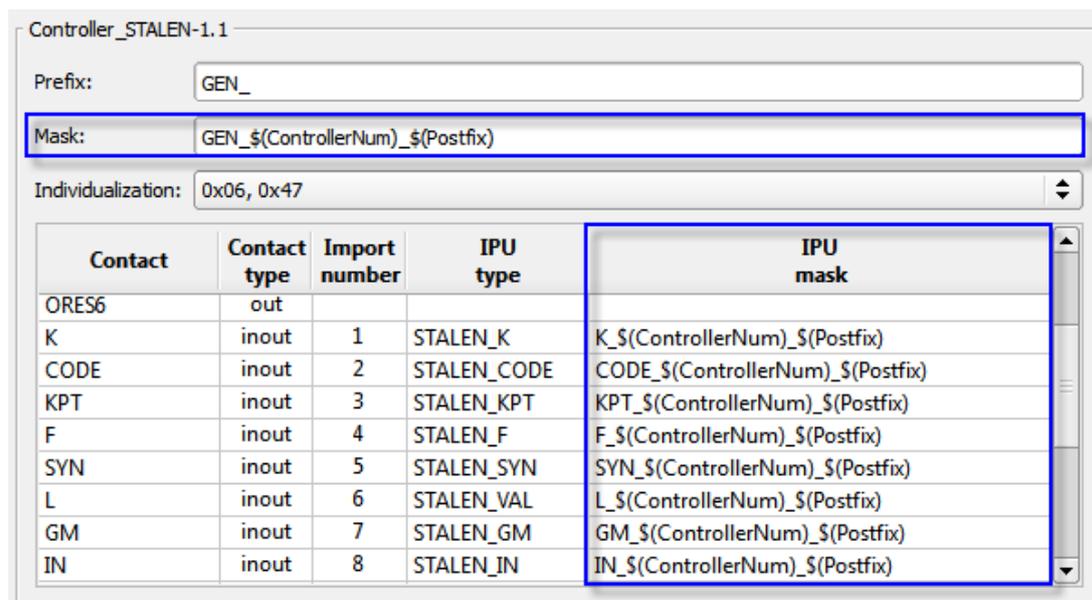
Маски типа контроллера и маски IPU объектов необходимы для автоматического генерирования имен IPU объектов из формы контроллера.

Для базовых типов контроллеров, маски определяются шаблонами проекта.

В редакторе типов контроллеров  задается маска:

- для типа контроллера (**Mask**), имя контроллера должно соответствовать указанной маске;
- для контакта типа контроллера задается маска IPU объекта (**IPU mask**), по которой будет автоматически генерироваться имя IPU объекта из формы контроллера.

Маски IPU объектов указываются для каждого контакта для группы индивидуализаций.



The screenshot shows the configuration for Controller_STALEN-1.1. The Prefix is GEN_. The Mask is GEN_\$(ControllerNum)_\$(Postfix). The Individualization is 0x06, 0x47. A table lists contacts and their corresponding IPU masks.

Contact	Contact type	Import number	IPU type	IPU mask
ORES6	out			
K	inout	1	STALEN_K	K_\$(ControllerNum)_\$(Postfix)
CODE	inout	2	STALEN_CODE	CODE_\$(ControllerNum)_\$(Postfix)
KPT	inout	3	STALEN_KPT	KPT_\$(ControllerNum)_\$(Postfix)
F	inout	4	STALEN_F	F_\$(ControllerNum)_\$(Postfix)
SYN	inout	5	STALEN_SYN	SYN_\$(ControllerNum)_\$(Postfix)
L	inout	6	STALEN_VAL	L_\$(ControllerNum)_\$(Postfix)
GM	inout	7	STALEN_GM	GM_\$(ControllerNum)_\$(Postfix)
IN	inout	8	STALEN_IN	IN_\$(ControllerNum)_\$(Postfix)

Рисунок 251 Редактирование масок типов контроллеров

Автоматическое генерирование IPU объектов в форме контроллера недоступно, если:

- отсутствует маска типа;
- некорректна или отсутствует маска IPU объекта;
- имя контроллера не соответствует маске типа.

Правила формирования масок

При заполнении масок можно выделить общую статичную часть (например, префикс типа контроллера GEN_), а также переменную часть (например, номер контроллера 1_2). Статическая часть в неизменном виде записывается в маску, переменная – обозначается при помощи специальных символов \$ или \$(number). Переменных частей в маске может быть несколько, статические и переменные части могут чередоваться.

Пример 2. Маски для типов контроллеров и IPU объектов

Имя контроллера **GEN_1_2_CH** соответствует маске **GEN_\$(ControllerNum)_\$(Postfix)**, где

- **GEN _** - статическая часть, т.е. все контроллеры данного типа должны иметь такой префикс;
- **\$(ControllerNum)_** - переменная часть, номер контроллера;
- **\$(Postfix)** – переменная часть, постфикс станции.

Аналогично для имени IPU объектов:

- IPU объект **L_1_2_CH** соответствует маске **L_\$(ControllerNum)_\$(Postfix)**.

Пример 3: Генерация имен для контактов контроллера

GEN_1_2_CH (маска: **GEN_\$(ControllerNum)_\$(Postfix)**):

Контакт	IPU объект	Маска IPU объекта
L	L_1_2_CH	L_\$(ControllerNum)_\$(Postfix)
F	F_1_2_CH	F_\$(ControllerNum)_\$(Postfix)

6.4 Параметры импорта из object_list

Для корректного импорта данных из object_list необходимо настроить следующие параметры проекта:

префиксы IPU типов – по префиксам определяется тип IPU объектов;

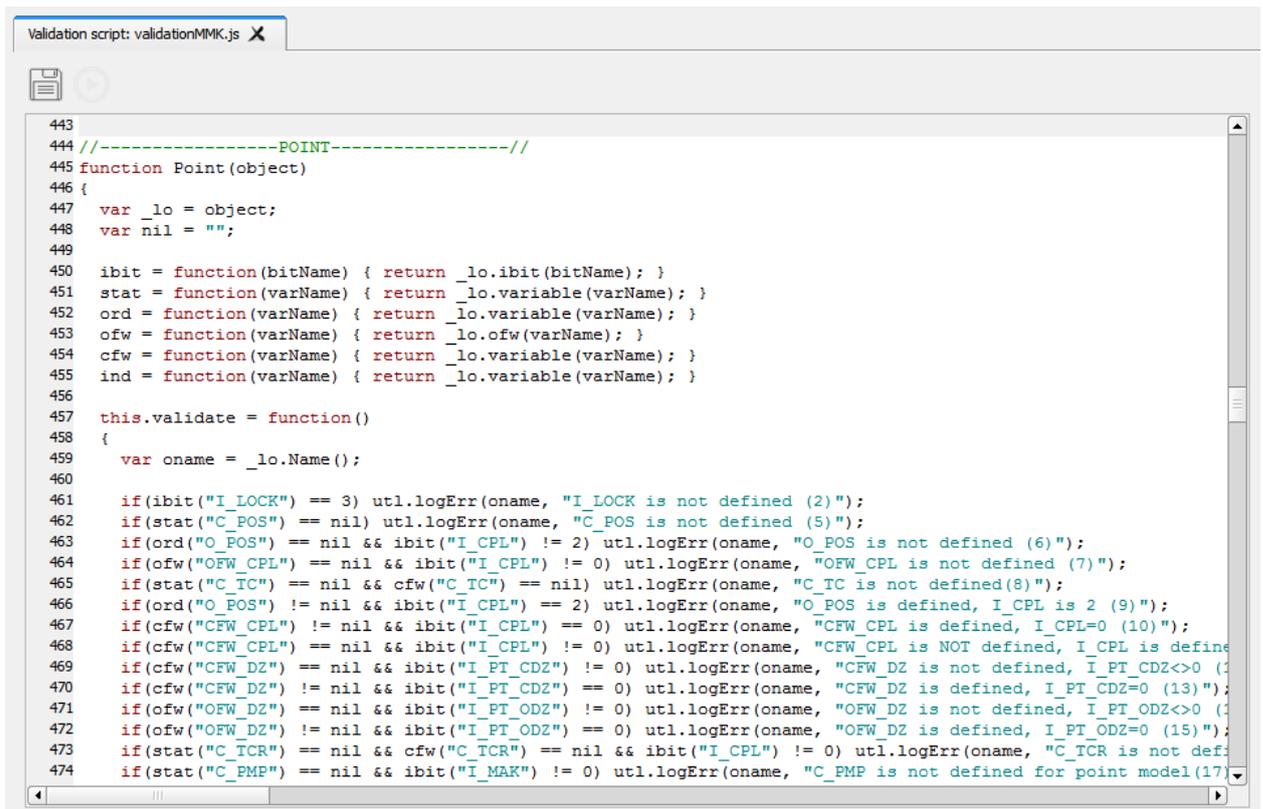
типы контроллеров, их префиксы – определяется тип экземпляров контроллеров;

соответствия контактов контроллеров IPU типам, Import number для контактов – определяется привязка IPU объектов на контакты контроллеров.

В типовом случае все базовые настройки проекта заполняются автоматически при создании проекта на основе выбранного пользователем шаблона, могут быть откорректированы пользователем!

6.5 Редактор скриптов валидации

В приложении предусмотрена возможность редактирования скриптов валидации (кнопка  **Show editor** в главном тулбаре проекта).



```
Validation script: validationMMK.js X
443
444 //-----POINT-----//
445 function Point(object)
446 {
447   var _lo = object;
448   var nil = "";
449
450   ibit = function(bitName) { return _lo.ibit(bitName); };
451   stat = function(varName) { return _lo.variable(varName); };
452   ord = function(varName) { return _lo.variable(varName); };
453   ofw = function(varName) { return _lo.ofw(varName); };
454   cfw = function(varName) { return _lo.variable(varName); };
455   ind = function(varName) { return _lo.variable(varName); };
456
457   this.validate = function()
458   {
459     var oname = _lo.Name();
460
461     if(ibit("I_LOCK") == 3) utl.logErr(oname, "I_LOCK is not defined (2)");
462     if(stat("C_POS") == nil) utl.logErr(oname, "C_POS is not defined (5)");
463     if(ord("O_POS") == nil && ibit("I_CPL") != 2) utl.logErr(oname, "O_POS is not defined (6)");
464     if(ofw("OFW_CPL") == nil && ibit("I_CPL") != 0) utl.logErr(oname, "OFW_CPL is not defined (7)");
465     if(stat("C_TC") == nil && cfw("C_TC") == nil) utl.logErr(oname, "C_TC is not defined(8)");
466     if(ord("O_POS") != nil && ibit("I_CPL") == 2) utl.logErr(oname, "O_POS is defined, I_CPL is 2 (9)");
467     if(cfw("CFW_CPL") != nil && ibit("I_CPL") == 0) utl.logErr(oname, "CFW_CPL is defined, I_CPL=0 (10)");
468     if(cfw("CFW_CPL") == nil && ibit("I_CPL") != 0) utl.logErr(oname, "CFW_CPL is NOT defined, I_CPL is define");
469     if(cfw("CFW_DZ") == nil && ibit("I_PT_CDZ") != 0) utl.logErr(oname, "CFW_DZ is not defined, I_PT_CDZ<0 (11)");
470     if(cfw("CFW_DZ") != nil && ibit("I_PT_CDZ") == 0) utl.logErr(oname, "CFW_DZ is defined, I_PT_CDZ=0 (13)");
471     if(ofw("OFW_DZ") == nil && ibit("I_PT_ODZ") != 0) utl.logErr(oname, "OFW_DZ is not defined, I_PT_ODZ<0 (14)");
472     if(ofw("OFW_DZ") != nil && ibit("I_PT_ODZ") == 0) utl.logErr(oname, "OFW_DZ is defined, I_PT_ODZ=0 (15)");
473     if(stat("C_TCR") == nil && cfw("C_TCR") == nil && ibit("I_CPL") != 0) utl.logErr(oname, "C_TCR is not def");
474     if(stat("C_PMP") == nil && ibit("I_MAK") != 0) utl.logErr(oname, "C_PMP is not defined for point model(17)");
```

Рисунок 262 Редактор скриптов

Кнопка  **Save scripts** для сохранения внесенных корректировок (файл со скриптами валидации, расположенный в директории с файлом проекта будет перезаписан).

7 Добавление станций

После завершения базовых настроек приложения (в типовом случае будут заданы автоматически при создании проекта) в проект необходимо добавить одну или несколько станций.

Добавление станции доступно на уровне проекта (**Add station**).

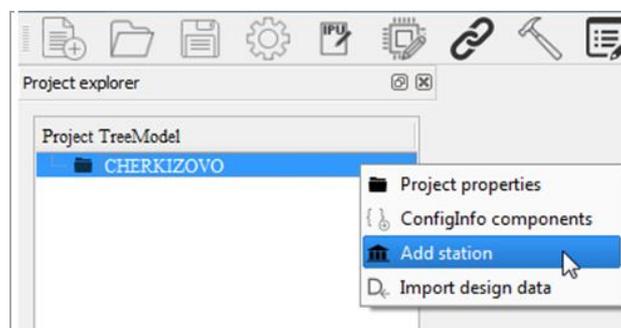


Рисунок 273 Добавление станции в древе проекта

В форме **Add station** необходимо указать имя станции и, при необходимости, ее постфикс. Постфикс необходим для разделения объектов в проекте на несколько станций; при создании и импорте, данные добавляются на станцию с постфиксом соответствующем постфиксу объекта.

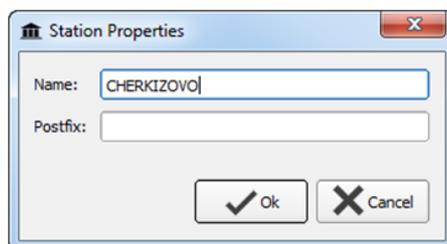


Рисунок 284 Добавление станции

В проект может быть добавлена одна станция без постфикса или неограниченное количество станций с постфиксами. В области панели инструментов отобразится панель инструментов станции (для работы с логическими объектами станции).

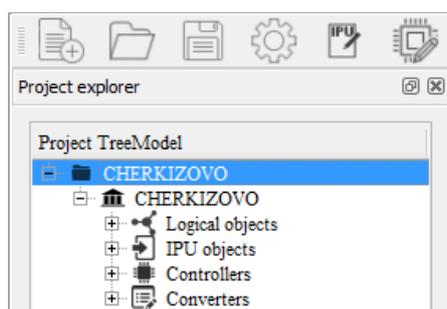


Рисунок 295 Отображение станции в дереве проекта

8 Автоматизированный ввод данных (базовое заполнение IntData, OCdata)

Автоматизированный ввод данных включает в себя следующие возможности:

- импорт данных о логических объектах из файла графа станции Visio;
- импорт / обновление данных от проектировщиков (из objectlist).

Если объекты относятся к типу со свойством COS indicated, то в результате импорта данных автоматически создаются соответствующие LogCOS и IPUCOS объекты, а также автоматически заполняется связь с соответствующими логическими и IPU объектами.

8.1 Импорт данных о логических объектах из Visio

Импорт данных из файла Visio позволяет загрузить данные о логических объектах и связях между ними, выполнить базовое автоматическое заполнение файла данных IntData.

Для импорта данных из Visio необходимо:

- из файла графа станции MS Visio при помощи Visio-Plugin выгрузить данные в формат *.xml;
- полученный файл *.xml импортировать в RailCAD в дереве проекта (**Import data from Visio plug-in**).

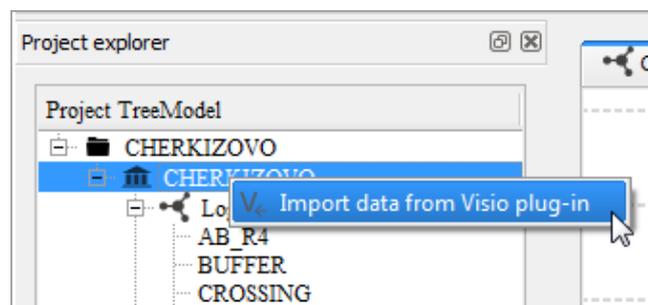


Рисунок 306 Импорт данных из файла Visio

В результате успешного импорта:

- данные о логических объектах и связях между ними отобразятся в графическом редакторе соответствующей станции;
- для импортированных объектов автоматически заполнится информация о связях индикационных переменных с LogCOS объектом, присвоятся COS номера.

8.2 Импорт данных проектировщиков из object_list

Импорт данных проектировщиков (из object_list) — импорт IPU объектов, контроллеров, связей между ними, позволяет выполнить базовое автоматическое заполнение файлов данных IntDATA, OSdata.

Импорт данных из object_list доступен в дереве проекта (**Import design data**).

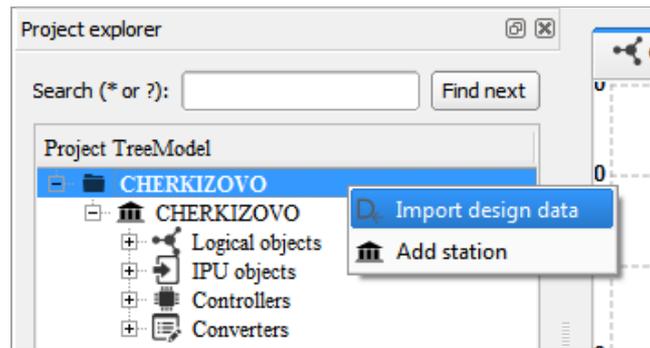


Рисунок 317 Импорт данных из object_list

В форме импорта необходимо выбрать object_list.txt и способ загрузки данных:

- **Import** — импорт данных с предварительной очисткой существующих данных в проекте;
- **Update** — обновление данных, без удаления существующих в проекте данных.

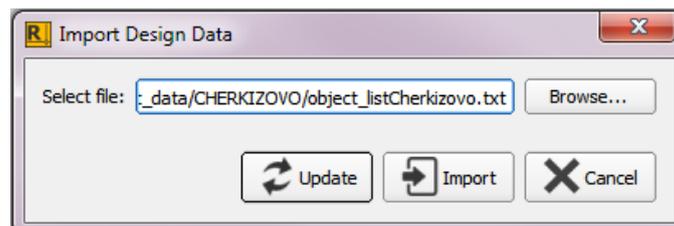


Рисунок 328 Выбор режима импорта данных из object_list

В результате успешного импорта:

- в проекте отобразятся данные о контроллерах, IPU объектах, связях между ними;
- для импортированных объектов автоматически заполнится информация о связях IPU объектов с IPUCOS объектами, присвоятся COS номера.

При завершении импорта данных с ошибками необходимо откорректировать импортируемые данные или настройки импорта.

При импорте данных из object_list в RailCAD автоматически выявляются ошибки при составлении исходного файла для загрузки:

- кириллические символы в названиях IPU объектов и контроллеров;

- несуществующие индивидуализации контроллеров;
- неуникальные адреса и номера контроллеров;
- неуникальные IPU объекты;
- IPU объекты, ссылающиеся на несуществующие контроллеры и т.д.

Если в процессе импорта найдены **некритические ошибки** (кириллические символы, строчные символы) – **импорт может быть продолжен** с автоматическим исправлением ошибок.

Если в процессе импорта найдены **критические ошибки** – **импорт / обновление данных невозможны**, то после устранения ошибок, необходимо выполнить импорт / обновление данных повторно.

К критическим ошибкам импорта можно отнести:

- невозможно определить тип IPU объекта или контроллера – необходимо откорректировать настройки импорта, проверить корректность префиксов;
- некорректные индивидуализации, IPU объекты связаны с несуществующими ОК и т.д – необходимо откорректировать импортируемый файл;
- и т.д.

8.3 Автоматическое связывание IPU объектов с переменными логических объектов

При помощи данной функциональности можно автоматически связать переменные логических объектов с IPU объектами.

Логический объект	Переменная	IPU объекта
PT1	O_POS	PT1N
PT2	O_POS	PT2N

Привязывание переменных происходит по определенным правилам, которые задаются при помощи присвоения маски для имен переменных и соответствующих им IPU объектов.

Автоматическое связывания IPU объектов с переменными логических объектов выполняется в редакторе ( **Logic object's vars connections to IPU**) при нажатии кнопки  **Connect**.

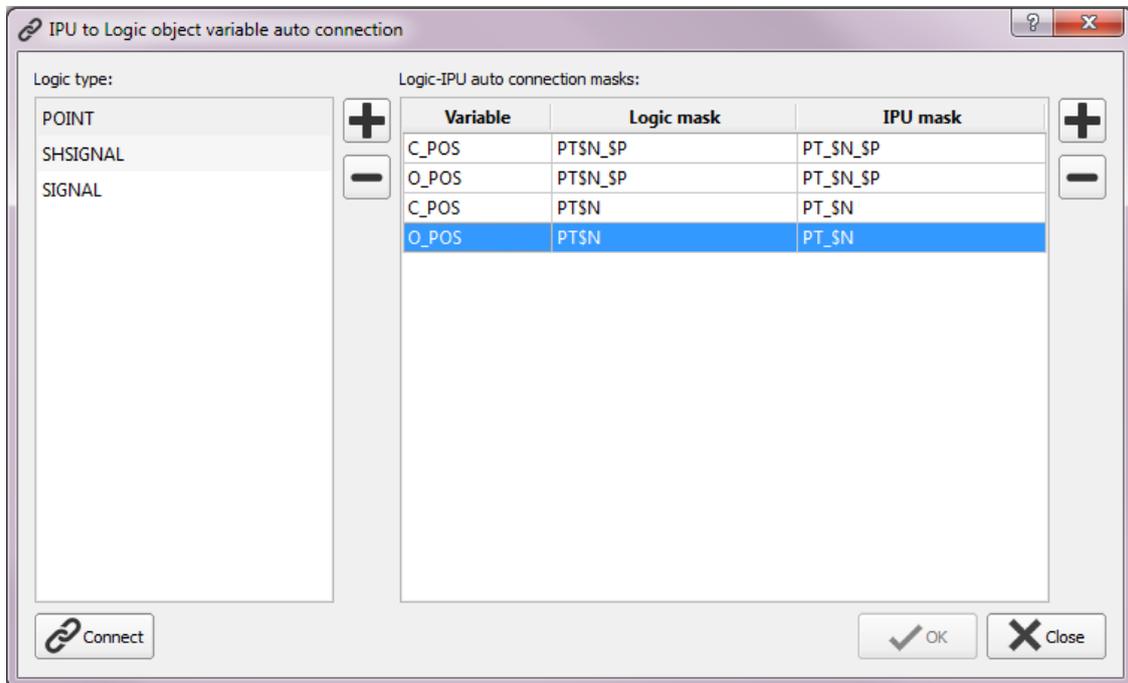


Рисунок 3933 Редактор связей IPU объектов с переменными логических объектов

Правила автоматического связывания IPU объектов с переменными логических объектов:

- **добавить/ удалить тип логического объекта**, для которого настроены правила; при удалении типа логического объекта все правила для этого типа также будут удалены;
- **добавить/ откорректировать/ удалить маску переменной или IPU объекта.**
- **Ok** – сохранить корректировки;
- **Cancel** – отменить корректировки.

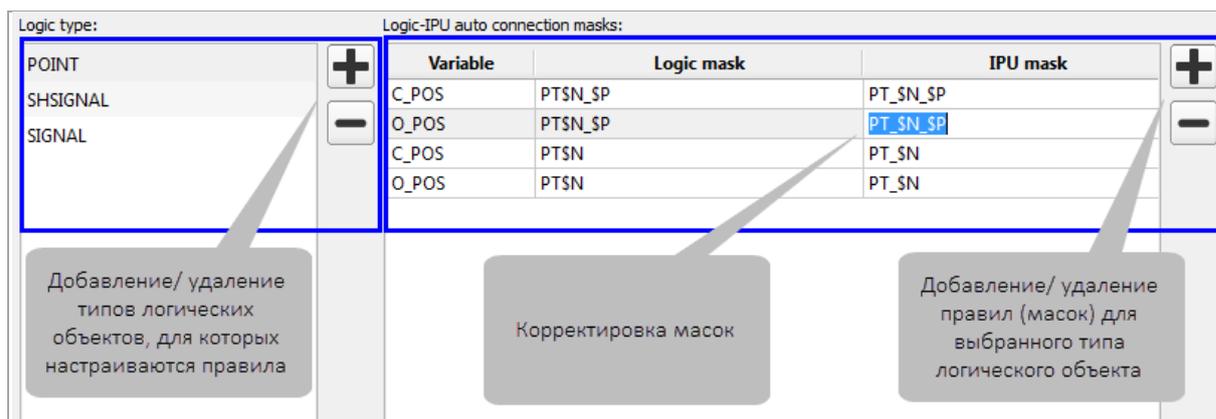


Рисунок 340 Редактор связей IPU объектов с переменными логических объектов

Правила формирования масок для имен переменных и IPU объектов

При заполнении масок можно выделить общую статичную часть (например, префикс типа логического объекта PT), а также переменную часть (например, номер логического объекта 1). Статическая часть в неизменном виде записывается в маску, переменная – обозначается при помощи специального символа \$. Переменных частей в маске может быть несколько, статические и переменные части могут чередоваться.

Пример 4: Маски для переменных логических объектов и IPU объектов

Логический объект типа POINT: **PT1** (маска: PT\$N)

Переменная	IPU объект	Маска IPU объекта
C_POS	PT_1	PT_\$N
O_POS	PT_1	PT_\$N

9 Работа с логическими объектами

В приложении предусмотрены следующие возможности по работе с логическими объектами:

- импорт из файла Visio;
- создание, удаление объектов и связей между ними в области графического редактора;
- корректировка параметров выбранного логического объекта (связывание с другими логическими и IPU объектами).

9.1 Графический редактор графа станции

Графический редактор станции предназначен для формирования графа станции: можно добавлять и удалять логические объекты и связи между ними.

Графический редактор отображается в центральной области приложения по двойному клику на уровне **Logical objects** необходимой станции в дереве проекта. Если в проекте несколько станций, то графический редактор для каждой из них отображается в отдельной закладке.

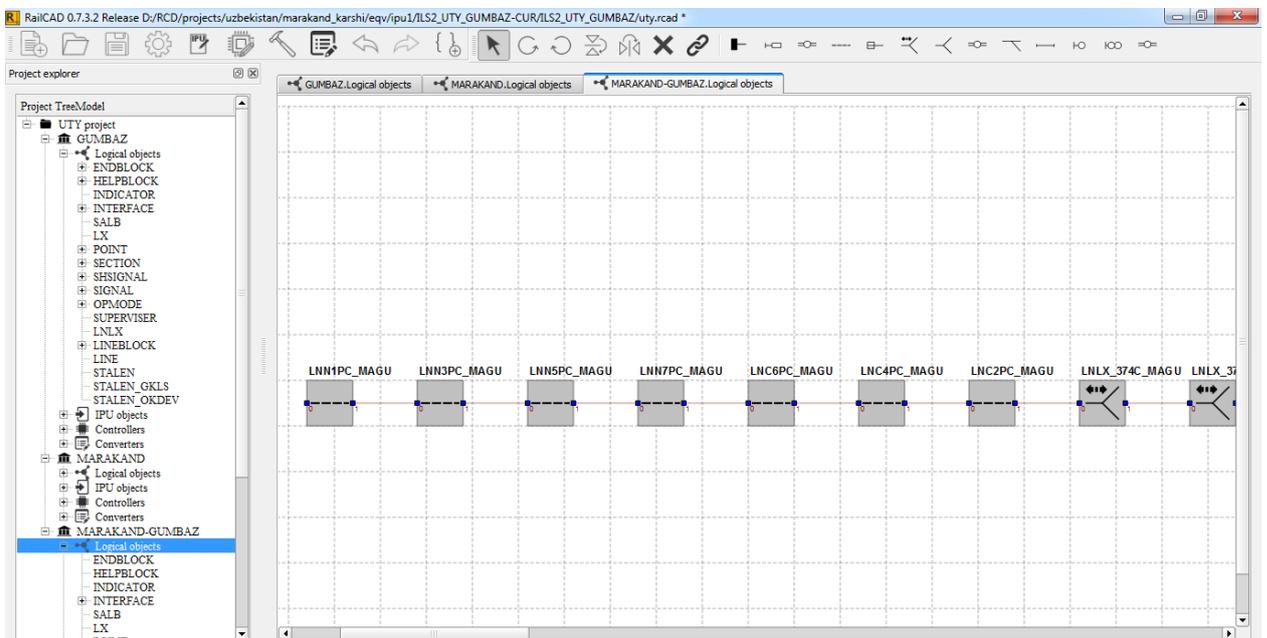


Рисунок 351 Графический редактор графа станции

Порядок закладок можно изменить путем их перетаскивания на нужную позицию. Графический редактор графа станции можно закрывать при нажатии кнопки **X Close** в заголовке закладки.

9.1.1 Добавление, редактирование, удаление логических объектов

Добавление логического объекта в проект выполняется следующим образом:

- на панели инструментов графического редактора станции выбрать тип логического объекта;



Рисунок 362 Выбор типа логического объекта

- левой клавишей мыши щелкнуть в области редактора. Новый объект отобразится в области графического редактора и будет выбран для редактирования;

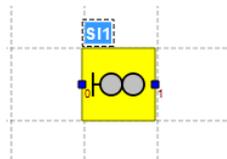


Рисунок 373 Добавление логического объекта

- изменить шаблонное имя логического объекта;
- новый объект отобразится в дереве проекта в соответствующем типе;
- для отмены добавления логического объекта на панели инструментов проекта нажать кнопку **Undo**.

После сохранения проекта, новый логический будет сохранен в файле IntData с базовыми настройками битов индивидуализации, описанными в файле sternol.

Для редактирования имени объекта два раза кликнуть мышкой на названии логического объекта, затем изменить имя.

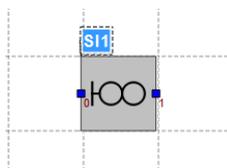


Рисунок 384 Редактирование имени логического объекта

Удаление логических объектов доступно:

- из дерева проекта при нажатии Delete;

- из графического редактора станции (выбрать объект или группу объектов):
- нажать кнопку  Remove на панели инструментов редактора;
- выбрать действие  Remove в контекстном меню объекта (открывается при клике ПКМ на объекте или группе объектов);
- при нажатии кнопки Delete.

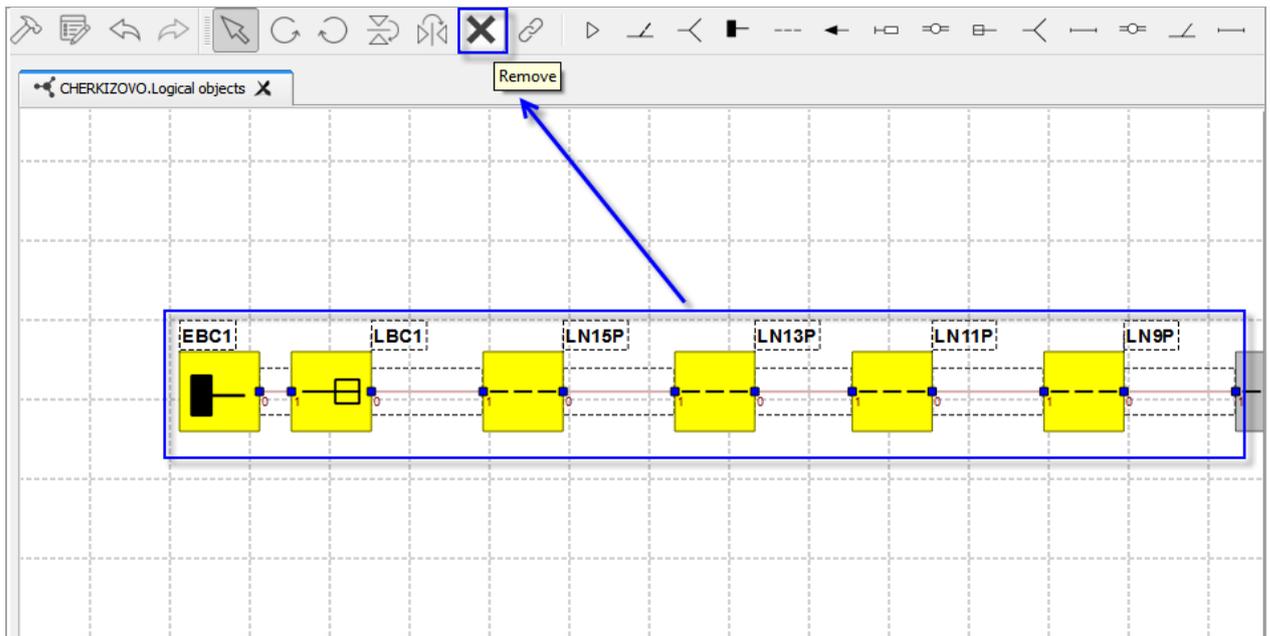


Рисунок 395 Удаление группы логических объектов

При удалении логических объектов связи между объектами разрываются, также удаляются связи логических объектов с IPU объектами и Freewired переменными других объектов.

9.1.2 Графическое редактирование объектов (вращение, отражение)

Для графического редактирования объектов необходимо выделить один или группу объектов, затем выполнить необходимое действие из контекстного меню объекта, при нажатии кнопки на панели управления редактором или при помощи сочетания горячих клавиш:

- вращать: влево  (R), вправо  (Shift + R);
- отразить сверху вниз  (F);
- отразить слева направо  (Shift + F).

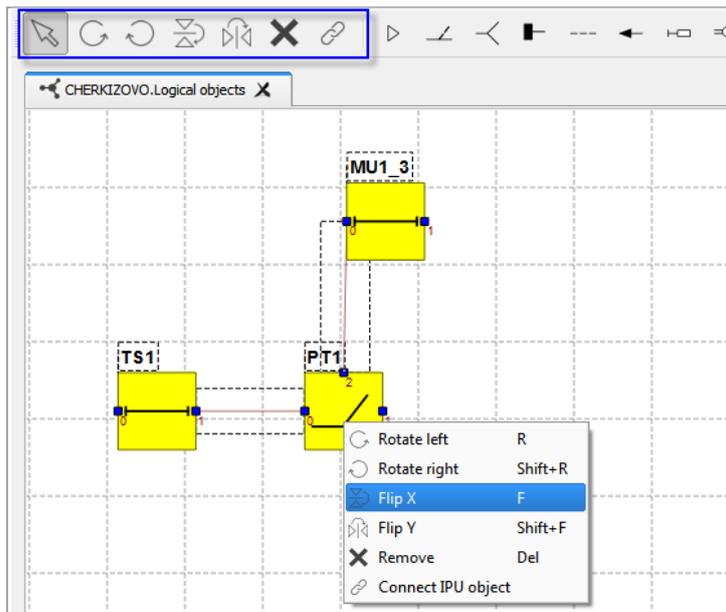


Рисунок 406 Редактирование группы логических объектов

9.1.3 Графическое редактирование связей логических объектов

Для связывания логических объектов между собой необходимо:

- навести курсор мыши на исходную точку соединения, от которой необходимо установить связь (соединительная точка увеличится);

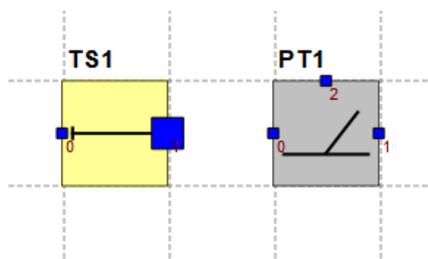


Рисунок 417 Выбор исходной точки соединения объектов

- «захватить» исходную точку соединения левой клавиши мыши (не отпускать), курсор вести к конечной соединительной точке (конечная точка должна подсветиться желтым цветом);

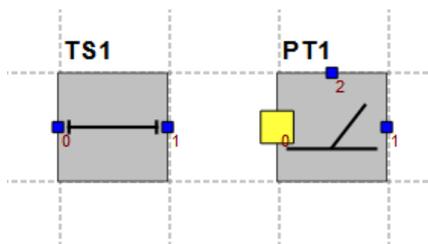


Рисунок 428 Выбор конечной точки соединения объектов

- отжать левую клавишу мыши;
- логические объекты соединены между собой, между объектами отображается соединительная линия.

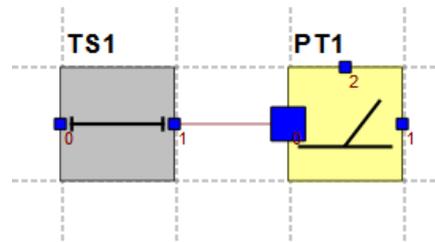


Рисунок 49 Соединение логических объектов

Для отмены связывания логических объектов необходимо нажать кнопку  **Undo**.

Для удаления добавленной связи необходимо выделить соединительную линию и нажать клавишу **Delete** или иконку  **Remove** на панели инструментов редактора.

9.1.4 Связывание IPU объекта с группой логических объектов

Для связывания IPU объекта с группой логических объектов необходимо в графическом редакторе выбрать один или несколько объектов, затем нажать кнопку  **Connect IPU object**.

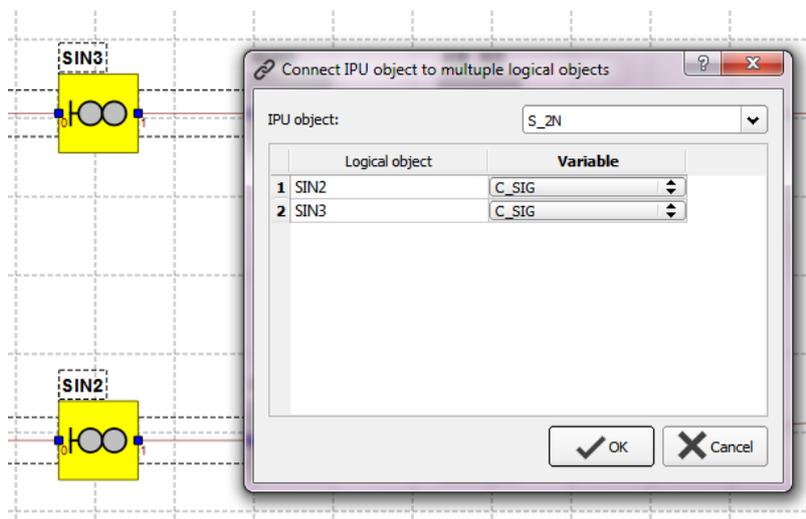


Рисунок 430 Связывание IPU объекта с группой логических объектов

В отобразившейся форме выбрать из списка IPU объект, затем для каждого логического объекта выбрать переменную, с которой будет связан IPU объект. **Ok** — для сохранения изменений, **Cancel** — для отмены связывания.

9.2 Корректировка параметров логических объектов

Корректировка свойств логического объекта доступна в форме **Object properties**, открывается при выборе логического объекта.

В форме логического объекта параметры разделены на группы, каждая группа отображается на отдельной вкладке:

- Общие свойства: имя объекта;
- Individualization – настройка битов индивидуализации;
- Freewired - связывание переменных freewired.
- Status, order - связывание status, order переменных с IPU объектами;
- присвоение COS номера;
- Indication - связывание индикационных переменных с LogCOS объектами.

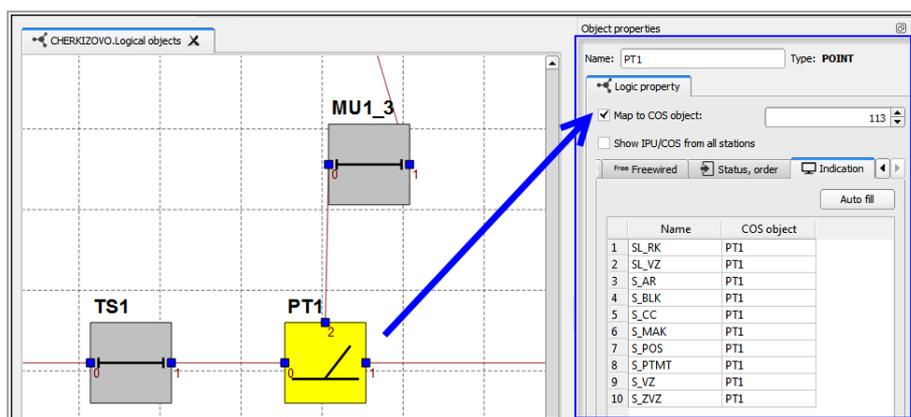


Рисунок 441 Автоматическое присвоение COS номера

Перечень переменных всех вышеуказанных групп, а также значения и описание индивидуализаций автоматически подтягиваются из файла **sternol**.

В форме логического объекта при выборе значений из выпадающих списков (вкладки **Status**, **order** и **Indication**) предусмотрена удобная фильтрация данных:

- список IPU/ COS объектов будет автоматически отфильтрован, если в поле **IPU object/ COS object** ввести часть имени искомого объекта;

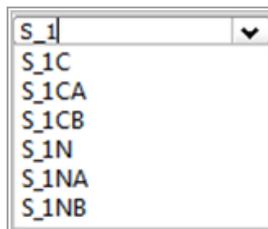


Рисунок 452 Фильтрация выпадающего списка

- по умолчанию в списке IPU/ COS объектов отображаются только объекты текущей станции, для отображения всех объектов проекта (со всех станций) необходимо выставить флаг «**Show IPU/COS from all station**».



Рисунок 463 Отображение объектов всех станций проекта

9.2.1 Переименование логического объекта

Для переименования логического объекта необходимо по двойному клику ЛКМ выбрать объект (в графическом редакторе или в дереве проекта). Далее в форме логического объекта откорректировать значение поля **Name**.

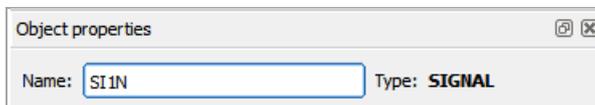


Рисунок 474 Переименование логического объекта

Также, переименование логического объекта доступно из области графического редактора при клике на имя объекта.

9.2.2 Биты индивидуализации

Редактирование битов индивидуализации доступно в форме логического объекта (открывается по двойному клику ЛКМ на логическом объекте) на вкладке **Individualization**.

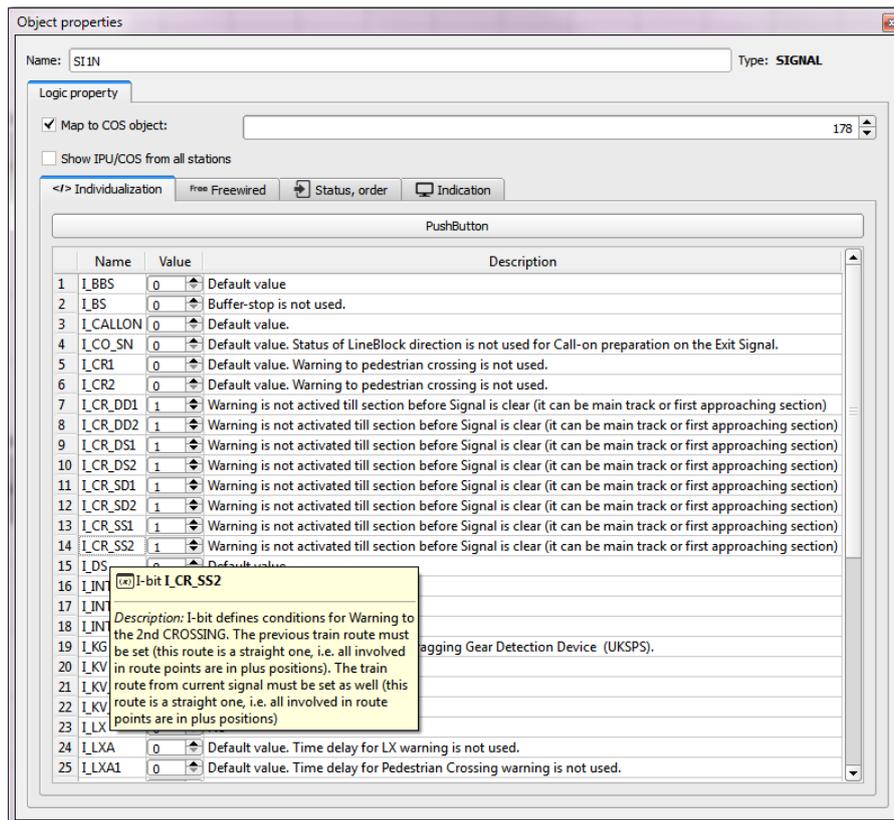


Рисунок 485 Редактирование битов индивидуализации

При наведении курсора на имя переменной во всплывающем окне отображается ее описание. В колонке **Value** можно откорректировать значение бита для выбранной переменной. В колонке **Description** отображается описание установленного значения индивидуализации.

9.2.3 Связывание freewired переменных

Для связывания переменных freewired необходимо открыть форму логического объекта на вкладке **Freewired**, далее по двойному клику в поле **Connected variables** перейти к форме связывания freewired переменных.

В форме **Freewired connections** в списке **Available variables** выбрать необходимые переменные, затем перенести их в список **Connection variables**.

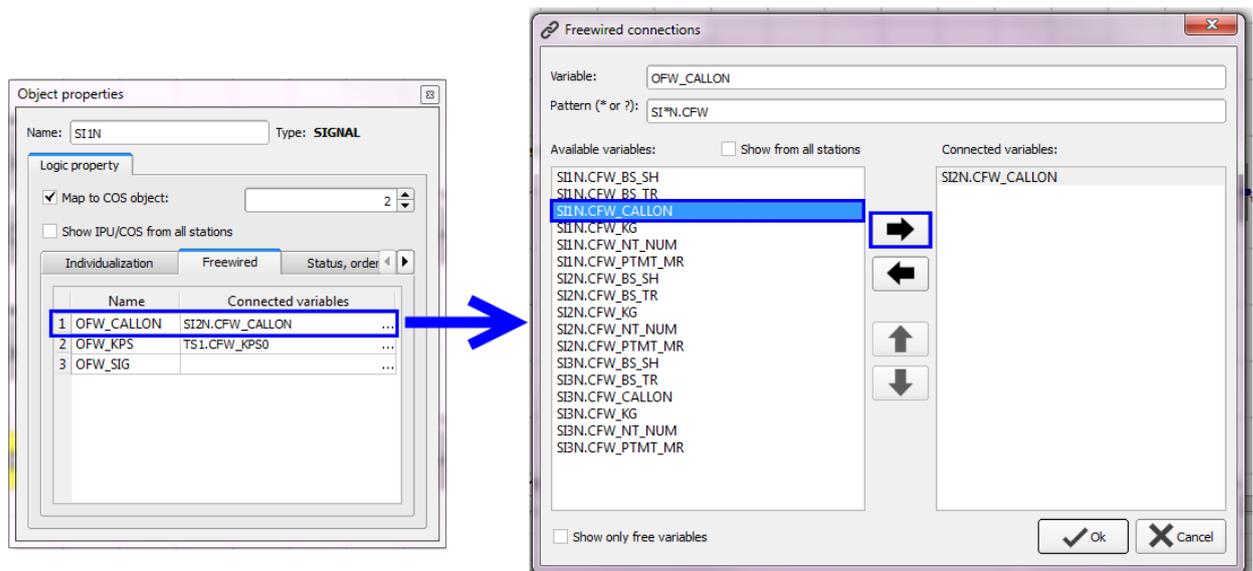


Рисунок 496 Связывание freewired переменных

По умолчанию в списке **Available variables** отображаются свободные переменные логических объектов текущей станции не связанные с другими объектами.

9.2.4 Фильтрация списка Available variables :

- поиск элементов списка с использованием специальных символов (* и ?);
- флаг **Show from all station** — если флаг выставлен, в списке отображаются переменные логических объектов других станций;
- **Show only free variables** — если флаг выставлен, в списке отображаются только свободные переменные (не связанные с другими объектами).

9.2.5 Связывание с IPU объектами (status, order переменные)

Связывание status и order переменных с IPU объектами выполняется в форме логического объекта на вкладке Status, order. Для order переменных доступна привязка к нескольким IPU объектам (заполнение order table).

Для связывания переменной с одним IPU объектом в поле **IPU object** из выпадающего списка выбрать необходимый объект.

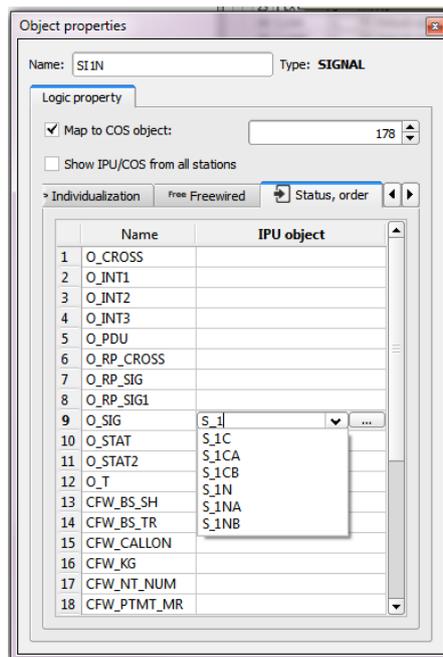


Рисунок 507 Связывание переменных с IPU объектами

9.2.6 Присвоение, удаление COS номеров логическим объектам

Присвоение COS номеров выполняется в форме логического объекта при выставлении флага **Map to COS object** (при добавлении и импорте объектов флаг выставлен по умолчанию).

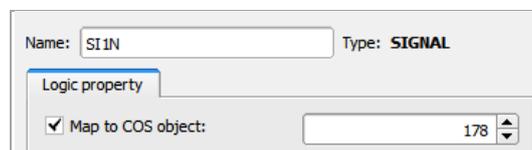


Рисунок 518 Присвоение COS номера логическому объекту

COS номер присваивается автоматически на единицу больше максимального номера в проекте. COS номер может быть изменен пользователем, при этом можно указать любой незанятый номер.

Логический объект является LOGCOS объектом, если для него указан COS номер.

Удаление COS номера (и LOGCOS объекта) происходит при снятии флага **Map to COS object**.

9.2.7 Связывание индикационных переменных с LOGCOS объектами

В приложении предусмотрена возможность связывания индикационных переменных с COS объектами в форме логического объекта на вкладке **Indication**.

Для индикационной переменной из выпадающего списка необходимо выбрать соответствующий COS объект.

При добавлении и импорте новых логических объектов все индикационные переменные логического объекта по умолчанию автоматически связываются с текущим LOGCOS объектом.

9.2.8 Автоматическое связывание переменных с текущим LOGCOS объектом

Автоматическое связывание переменных с текущим COS объектом доступно в форме логического объекта на вкладке **Indication** при нажатии кнопки **Auto fill**, в случае если текущему логическому объекту присвоен COS номер.

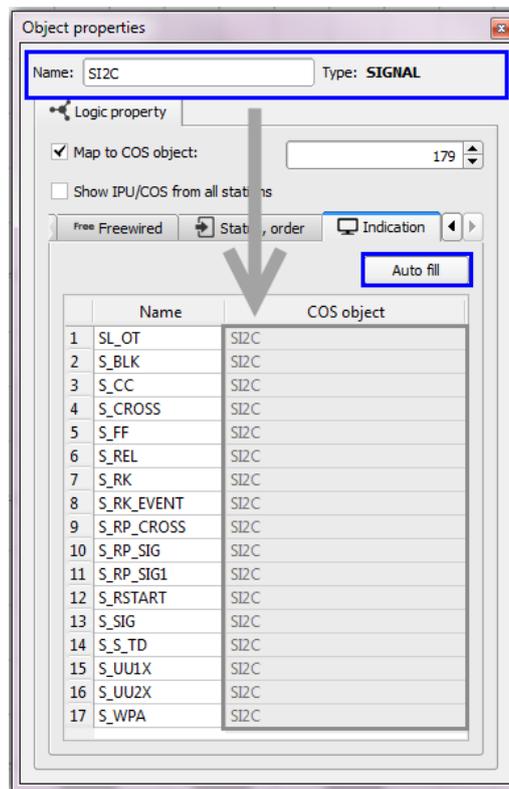


Рисунок 5952 Автоматическое связывание индикационных переменных

9.2.9 Валидация логических объектов

В приложении предусмотрена возможность валидации логических объектов и связей между ними:

проверка имен;

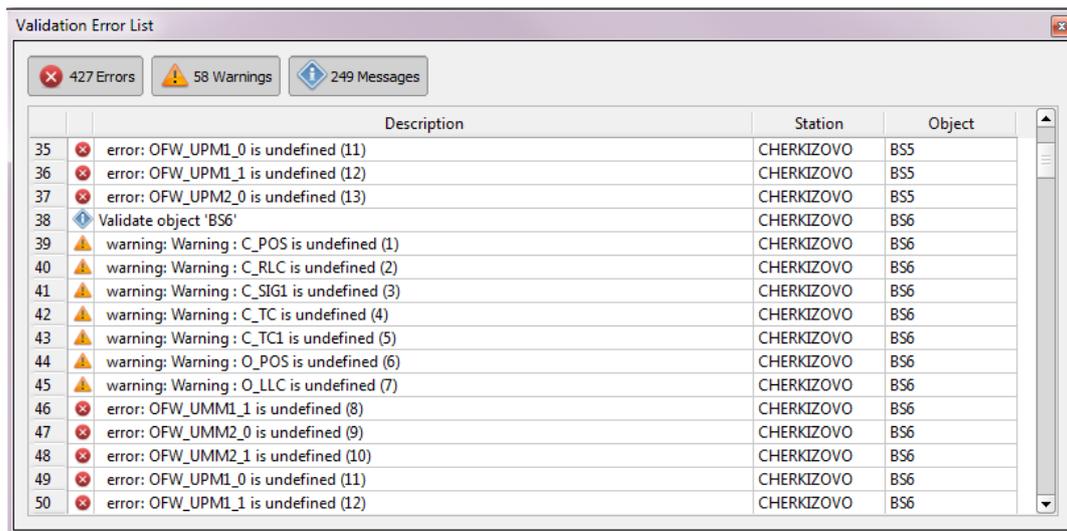
наличие несвязанных логических объектов;

обход виртуальных секций;

проверка корректности установленных значений переменных логических объектов (на основе скриптов валидации).

Для запуска валидации логических объектов необходимо на панели инструментов нажать кнопку  **Start build** или клавишу **F7**.

Результат выполнения валидации будет отображен на закладках:



The screenshot shows a window titled "Validation Error List" with a summary bar at the top indicating 427 Errors, 58 Warnings, and 249 Messages. Below this is a table with columns for line number, icon, description, station, and object.

		Description	Station	Object
35	✖	error: OFW_UPM1_0 is undefined (11)	CHERKIZOVO	BS5
36	✖	error: OFW_UPM1_1 is undefined (12)	CHERKIZOVO	BS5
37	✖	error: OFW_UPM2_0 is undefined (13)	CHERKIZOVO	BS5
38	🔍	Validate object 'BS6'	CHERKIZOVO	BS6
39	⚠	warning: Warning : C_POS is undefined (1)	CHERKIZOVO	BS6
40	⚠	warning: Warning : C_RLC is undefined (2)	CHERKIZOVO	BS6
41	⚠	warning: Warning : C_SIG1 is undefined (3)	CHERKIZOVO	BS6
42	⚠	warning: Warning : C_TC is undefined (4)	CHERKIZOVO	BS6
43	⚠	warning: Warning : C_TC1 is undefined (5)	CHERKIZOVO	BS6
44	⚠	warning: Warning : O_POS is undefined (6)	CHERKIZOVO	BS6
45	⚠	warning: Warning : O_LLC is undefined (7)	CHERKIZOVO	BS6
46	✖	error: OFW_UMM1_1 is undefined (8)	CHERKIZOVO	BS6
47	✖	error: OFW_UMM2_0 is undefined (9)	CHERKIZOVO	BS6
48	✖	error: OFW_UMM2_1 is undefined (10)	CHERKIZOVO	BS6
49	✖	error: OFW_UPM1_0 is undefined (11)	CHERKIZOVO	BS6
50	✖	error: OFW_UPM1_1 is undefined (12)	CHERKIZOVO	BS6

Рисунок 530 Результаты валидации логических объектов

- **Validation Error List** – информации о результатах валидации может быть отфильтрована: ошибки (**Errors**), предупреждения (**Warnings**), сообщения (**Messages**). При двойном клике по объекту валидации (в колонке **Objects**), он выделяется в области графического редактора.
- **Validation Output** – результат валидации в текстовом формате.

10 Работа с IPU объектами

В приложении предусмотрены следующие возможности по работе IPU объектами:

- импорт и обновление из `object_list`;
- создание/ редактирование/ удаление объектов из дерева проекта;
- создание IPU объектов из формы контроллера.

10.1 Добавление, редактирование, удаление IPU объекта

Создание новых IPU объектов доступно в дереве проекта на уровне IPU типа (Add IPU object).

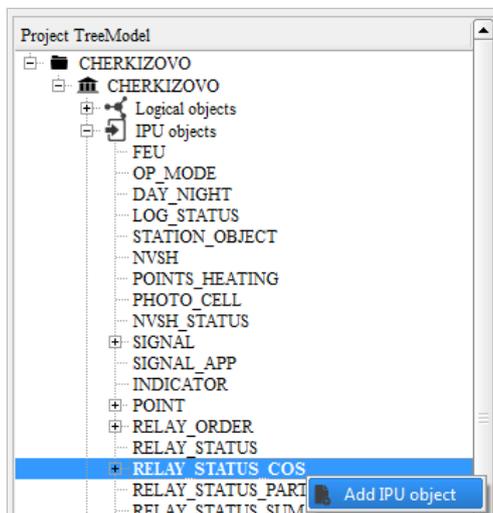


Рисунок 541 Добавление IPU объекта

В форме IPU объекта необходимо заполнить его параметры, а также выполнить связывание с переменными логических объектов.

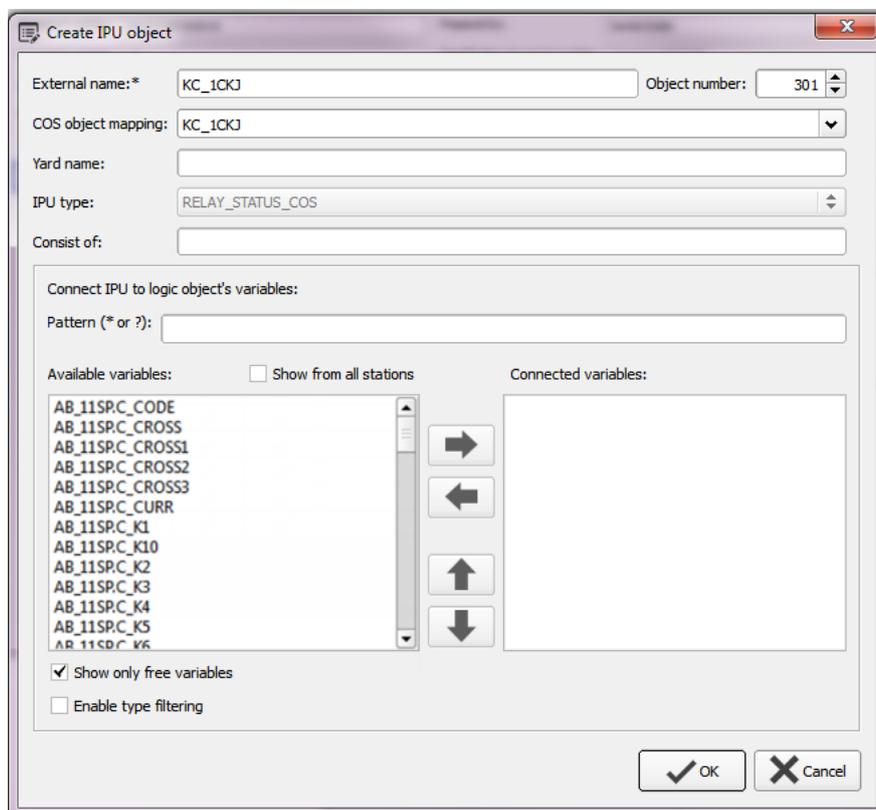


Рисунок 552 Форма IPU объекта

Если для IPU типа, по которому создаем IPU объект, указано свойство COS indicated (например, RELAY_STATUS_COS, RELAY_COS_K1 и т.д.), то при добавлении IPU объекта такого типа автоматически будут созданы соответствующие IPUCOS объекты, присвоены COS номера. Также, для IPU объекта автоматически заполняется связь с текущим IPUCOS объектом.

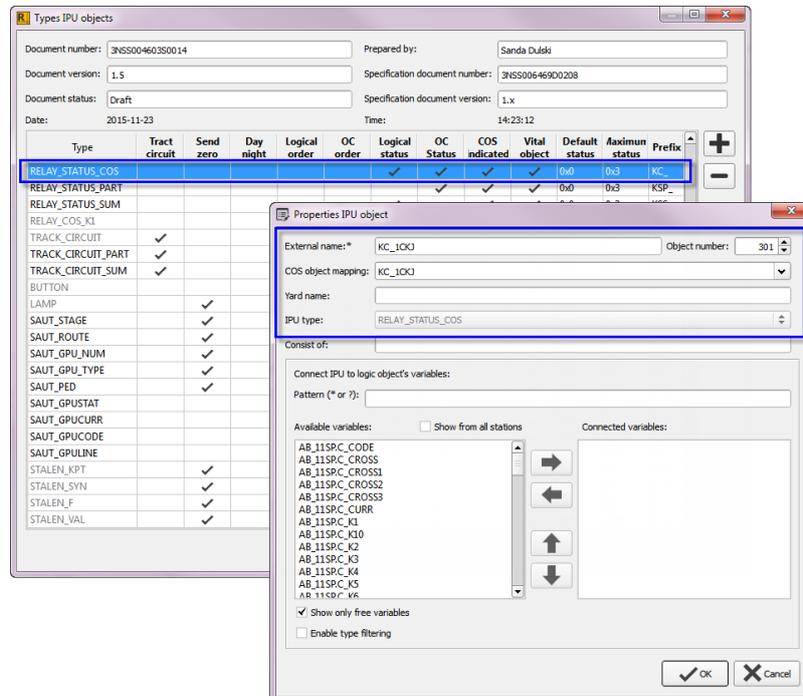


Рисунок 563 Присвоение COS номера IPU объекту

Для связывания IPU объекта с переменными логических объектов необходимо в списке **Available variables** выбрать необходимые переменные, затем перенести их в список **Connection variables**.

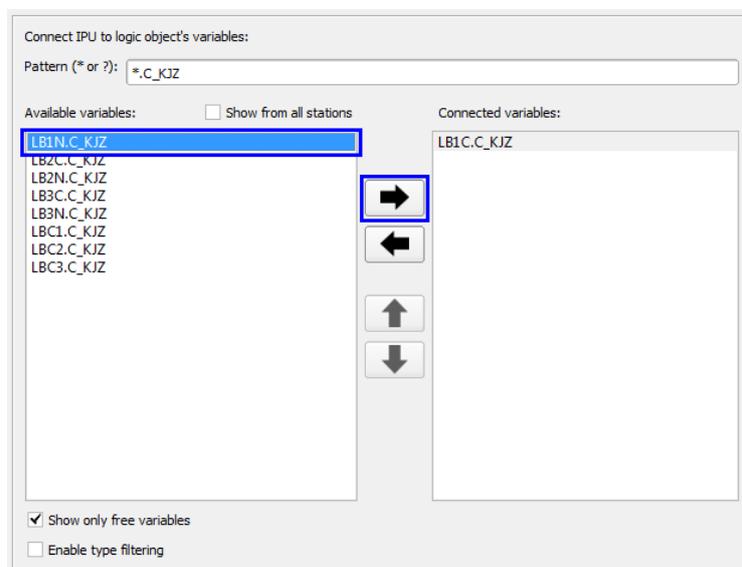


Рисунок 574 Связывание IPU объектов с переменными логических объектов

Если в список Connected variable добавлены «занятые» переменные (связанные с другими IPU объектами), то при нажатии Ok в форме IPU объекта пользователю будет предложено перепривязать переменные на текущий объект.

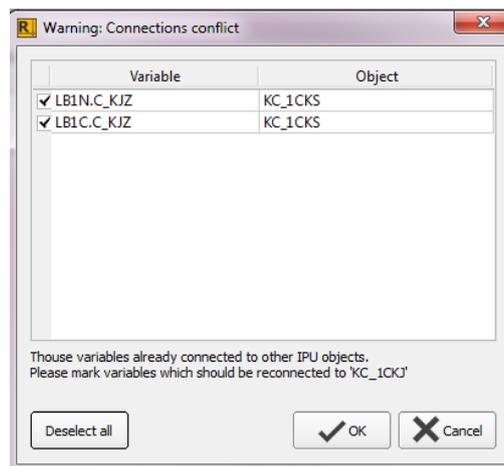


Рисунок 585 Сообщение о конфликтах связи

Фильтрация списка Available variables :

- поиск элементов списка с использованием специальных символов (* и ?) (например, *.C_KJZ - отобразить переменные C_KJZ всех логических объектов);
- флаг Show from all station — если флаг выставлен, в списке отображаются переменные логических объектов других станций;
- Show only free variables — если флаг выставлен, в списке отображаются только свободные переменные (не связанные с другими IPU объектами);
- Enable type filtering - отображение переменных логических объектов, которые соответствуют типу создаваемого/редактируемого IPU объекта (соответствия IPU типов логическим объектам задается в настройках проекта). Например, для IPU объекта POINT в списке переменных будут отображаться только переменные логических объектов типа POINT.

- По умолчанию в списке Available variables отображаются переменные логических объектов текущей станции не связанные с другими IPU объектами.
- Редактирование IPU объекта происходит аналогично его созданию, форма редактирования IPU объекта открывается по двойному клику ЛКМ или по действию Properties контекстного меню экземпляра объекта.



Рисунок 596 Контекстное меню IPU объекта

Удаление IPU объекта: Delete или **✗ Remove IPU object** в контекстном меню экземпляра объекта.

При сохранении проекта IPU, IPUCOS объекты будут сохранены в IntData.

10.2 Отображение связанных логических объектов в графическом редакторе

При выборе IPU объекта в дереве проекта, в области графического редактора подсвечиваются все логические объекты, связанные с текущим IPU объектом.

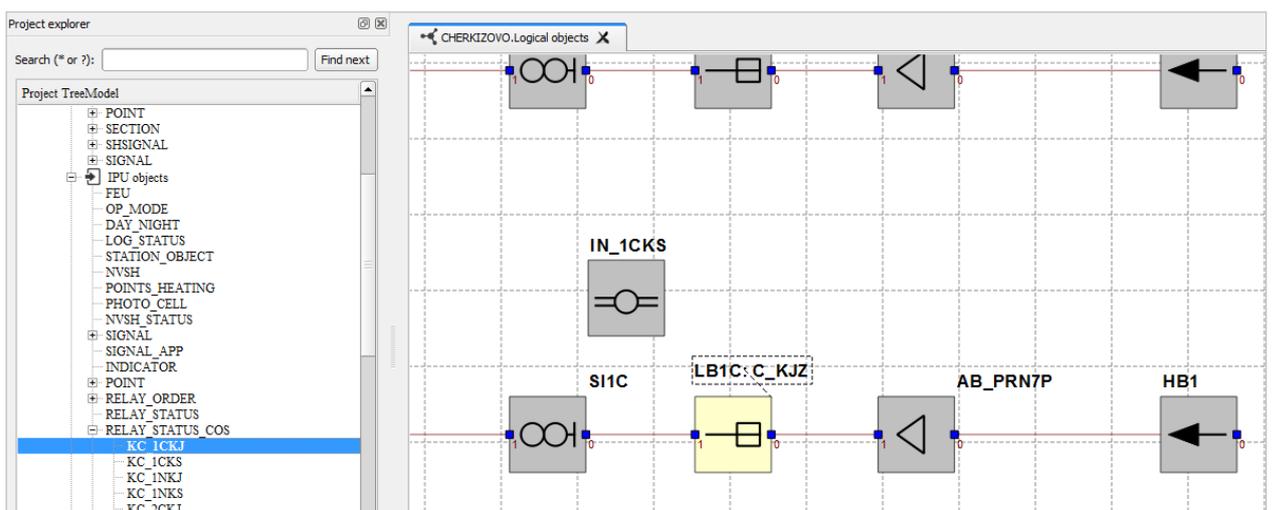


Рисунок 607 Отображение в области редактора связанных с IPU объектом логических объектов

11 Работа с конвертерами

Добавление конвертеров выполняется в дереве проекта в ветке **Converters** на уровне соответствующего типа (**Add converter**).

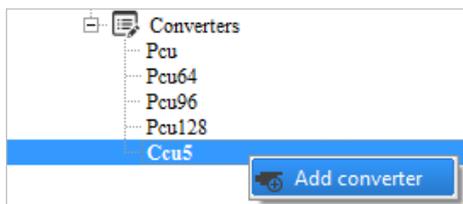


Рисунок 618 Добавление конвертера

В форме конвертера необходимо заполнить информацию о создаваемом объекте (**Ok** - для сохранения данных, **Cancel** - для отмены корректировок).

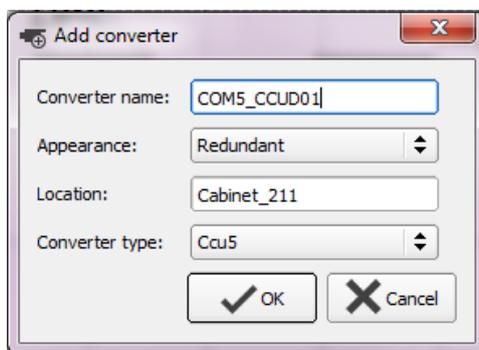


Рисунок 69 Форма конвертера

После сохранения проекта данные о конвертерах будут записаны в OSdata.

Редактирование конвертера происходит аналогично созданию (Properties или двойной клик на экземпляре конвертера).

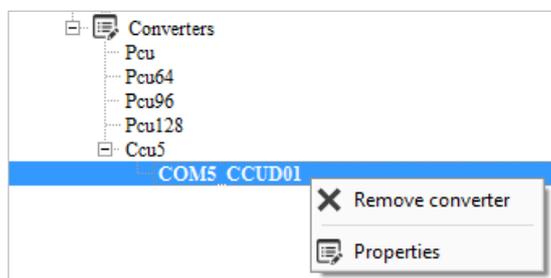


Рисунок 70 Контекстное меню экземпляра конвертера

Удаление конвертера: **Delete** или **✗ Remove converter** в контекстном меню экземпляра конвертера.

12 Работа с контроллерами

В приложении предусмотрены следующие возможности по работе контроллерами:

- импорт и обновление данных из object_list (импорт контроллеров и автоматическое связывание контактов с IPU объектами);
- создание/ редактирование/ удаление экземпляров контроллеров;
- возможность создания IPU объектов из формы контроллера.

Для работы с экземплярами контроллеров, в проект должны быть добавлены соответствующие типы контроллеров (базовые типы контроллеров определяются автоматически шаблоном проекта).

12.1 Добавление, корректировка, удаление контроллеров

Создание новых контроллеров доступно в дереве проекта на уровне типа контроллера (**Add controller**).

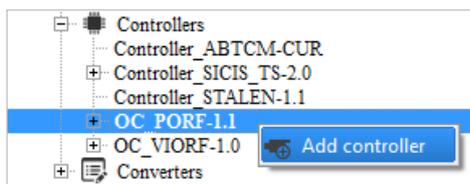
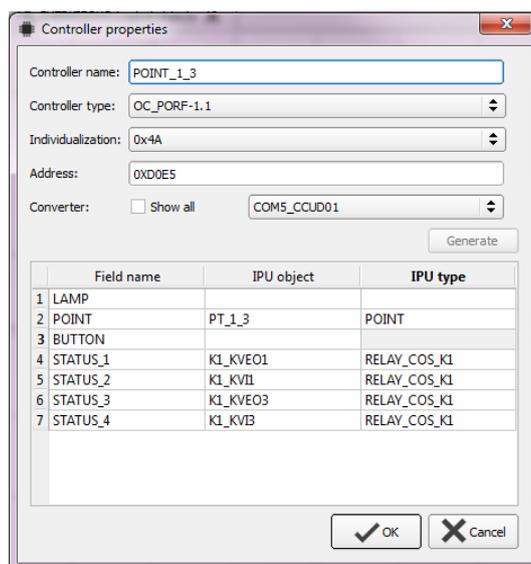


Рисунок 71 Добавление контроллера

В форме контроллера заполнить имя (Controller name) и параметры экземпляра.

A screenshot of a 'Controller properties' dialog box. It contains several input fields: 'Controller name' (POINT_1_3), 'Controller type' (OC_PORF-1.1), 'Individualization' (0x4A), 'Address' (0XDOE5), and 'Converter' (COM5_CCUD01). There is a 'Generate' button. Below the fields is a table with three columns: 'Field name', 'IPU object', and 'IPU type'.

	Field name	IPU object	IPU type
1	LAMP		
2	POINT	PT_1_3	POINT
3	BUTTON		
4	STATUS_1	K1_KVE01	RELAY_COS_K1
5	STATUS_2	K1_KV1B	RELAY_COS_K1
6	STATUS_3	K1_KVE03	RELAY_COS_K1
7	STATUS_4	K1_KV1B	RELAY_COS_K1

At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Рисунок 622 Форма контроллера

Указать конвертер (**Converter**). Если список **Converter** пустой, необходимо поставить флаг **Show all** для отображения в выпадающем списке конвертеров со всех станций или добавить экземпляр конвертера, затем продолжить создание контроллера.

Указать тип контроллера и индивидуализацию. В соответствии с типом контроллера и выбранной индивидуализацией отобразится перечень контактов.

Для каждого контакта из выпадающего списка выбрать необходимый IPU объект (тип определится автоматически).

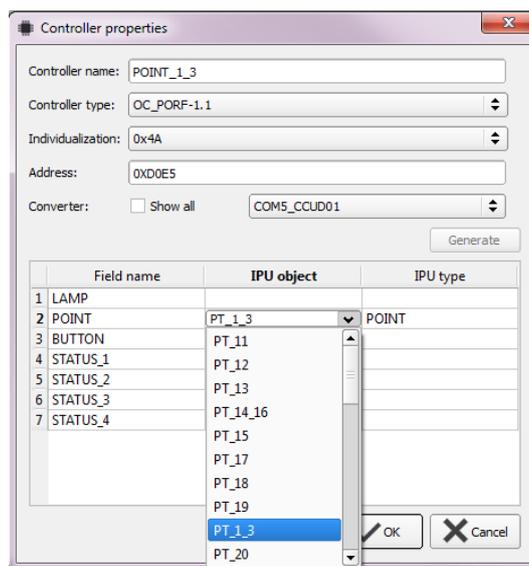


Рисунок 633 Выбор IPU объектов в форме контроллера

Если необходимый IPU объект отсутствует в списке:

указать имя IPU объекта в поле **IPU object** вручную, затем нажать **Ok**; автоматически сгенерировать имена IPU объектов (кнопка **Generate**).

При сохранении экземпляра контроллера (Ok), необходимые IPU объекты будут созданы автоматически.

Если для выбранной индивидуализации не заполнено соответствие контактов IPU типам, связывание контактов контроллера с IPU объектами будет недоступно. Необходимо заполнить соответствия в редакторе типов контроллеров.

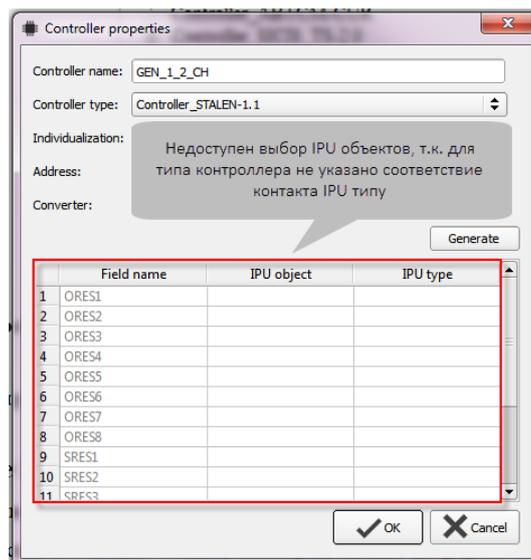


Рисунок 644 Форма контроллера (выбор IPU объектов недоступен)

Нажать **Ok** – для сохранения корректировок, **Cancel** – для отмены корректировок.

Редактирование контроллера происходит аналогично его созданию. Форма редактирования открывается по двойному клику ЛКМ или по действию **Properties** контекстного меню экземпляра.

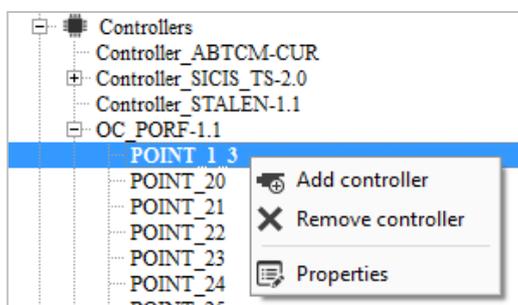


Рисунок 655 Контекстное меню экземпляра контроллера

Удаление контроллера: **X Remove** в контекстном меню экземпляра контроллера.

При сохранении проекта информация о контроллерах будет сохранена в OCdata.

12.2 Создание IPU объектов из формы контроллера

Если при связывании контактов контроллера с IPU объектами, нужный IPU объект отсутствует в списке, его можно создать непосредственно из формы контроллера.

В поле **IPU object** соответствующего контакта ввести имя необходимого IPU объекта (не должен содержаться в выпадающем списке объектов), нажать кнопку **Ок**.

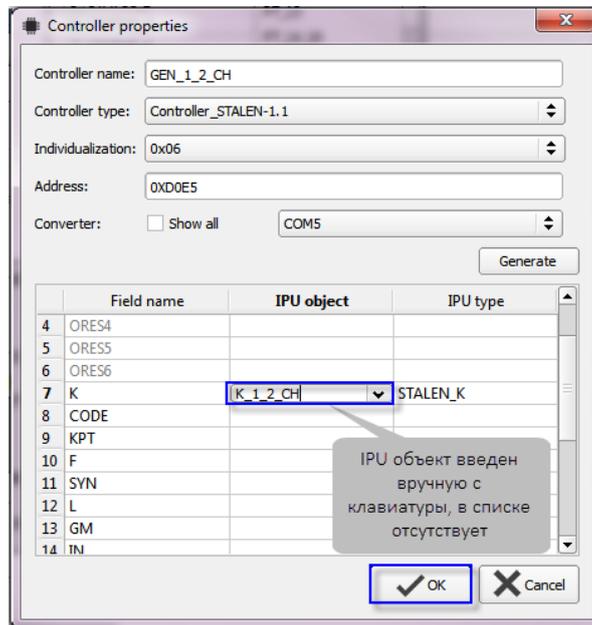


Рисунок 666 Создание IPU объектов из формы контроллера

При сохранении экземпляра контроллера, указанные IPU объекты будут созданы автоматически.

12.3 Автоматическая генерация имен IPU объектов из формы контроллера

При связывании контактов контроллера с IPU объектами, нужные IPU объекты могут быть сгенерированы автоматически на основе указанных масок типов контроллеров.

В форме контроллера указать имя контроллера (должно соответствовать маске типа контроллера), выбрать индивидуализацию и нажать кнопку **Generate**, в соответствии с маской IPU объекта будут сгенерированы имена IPU объектов. При сохранении экземпляра контроллера, сгенерированные IPU объекты будут созданы автоматически.

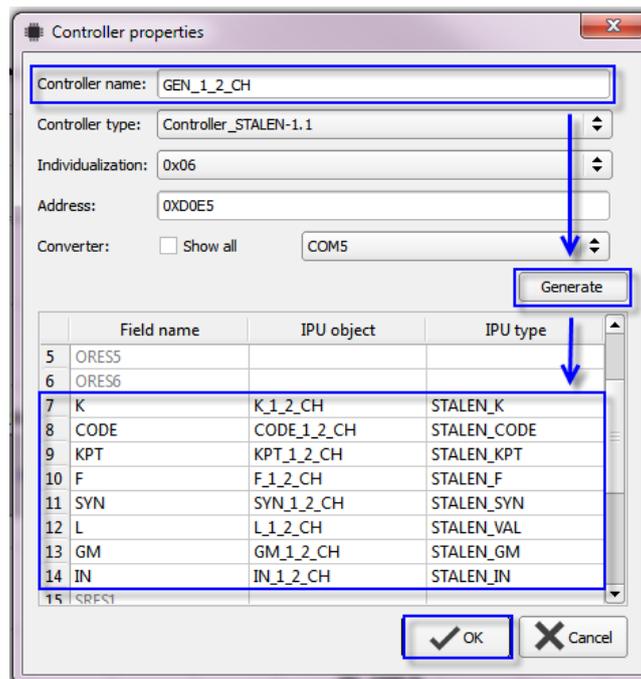


Рисунок 677 Автоматическая генерация IPU объектов

Для создания IPU объектов из формы контроллера необходимо в редакторе типов контроллера:

- указать соответствие контактов контроллера IPU типу;
- указать маски для типа контроллера и IPU объектов. Если маска для типа контроллера не указана, генерирование IPU объектов недоступно.

13 Сохранение проекта

Для сохранения проекта и перезаписи файлов данных необходимо нажать кнопку  **Save** на главной панели инструментов.

В результате сохранения проекта будут перезаписаны следующие файлы данных:

- IntData;
- OCdata;
- ADMGENTAB (если файл не read only и по типам IPU объектов были изменения).

Контрольные суммы файлов данных при сохранении проекта автоматически пересчитываются.