МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания
по выполнению лабораторных работ
по дисциплине
«Цифровые системы автоматизированного проектирования»
для студентов направления подготовки
15.03.04 Автоматизация технологических процессов и производств
направленность (профиль):
Информационно-управляющие системы

Невинномысск 2024 г.

Содержание

Введение

Современная АСУТП (автоматизированная система управления технологическим процессом) представляет собой многоуровневую человеко-машинную систему управления. Создание АСУ сложными технологическими процессами осуществляется с использованием автоматических информационных систем сбора данных и вычислительных комплексов, которые постоянно совершенствуются по мере эволюции технических средств и программного обеспечения.

Непрерывную во времени картину развития АСУТП можно разделить на три этапа, обусловленные появлением качественно новых научных идей и технических средств. В ходе истории меняется характер объектов и методов управления, средств автоматизации и других компонентов, составляющих содержание современной системы управления.

Первый этапотражает внедрение систем автоматического регулирования (САР). Объектами управления на этом этапе являются отдельные параметры, установки, агрегаты; решение задач стабилизации, программного управления, слежения переходит от человека к САР. У человека появляются функции расчета задания и параметры настройки регуляторов.

Второй этап- автоматизация технологических процессов. Объектом управления становится рассредоточенная в пространстве система; с помощью систем автоматического управления (САУ) реализуются все более сложные законы управления, решаются задачи оптимального и адаптивного управления, проводится идентификация объекта и состояний системы. Характерной особенностью этого этапа является внедрение систем телемеханики в управление технологическими процессами. Человек все больше отдаляется от объекта управления, между объектом и диспетчером выстраивается целый ряд измерительных систем, исполнительных механизмов, средств телемеханики, мнемосхем и других средств отображения информации (СОИ).

Третий этап- автоматизированные системы управления технологическими процессами - характеризуется внедрением в управление технологическими процессами вычислительной техники. Вначале - применение микропроцессоров, использование на отдельных фазах управления вычислительных систем; затем активное развитие человеко-машинных систем управления, инженерной психологии, методов и моделей исследования операций и, наконец, диспетчерское управление на основе использования автоматических информационных систем сбора данных и современных вычислительных комплексов.

От этапа к этапу менялись и функции человека (оператора/диспетчера), призванного обеспечить регламентное функционирование технологического процесса. Расширяется круг задач, решаемых на уровне управления; ограниченный прямой необходимостью управления технологическим процессом набор задач пополняется качественно новыми задачами, ранее имеющими вспомогательный характер или относящиеся к другому уровню управления.

Диспетчер в многоуровневой автоматизированной системе управления технологическими процессами получает информацию с монитора ЭВМ или с электронной системы отображения информации и воздействует на объекты, находящиеся от него на значительном расстоянии с помощью телекоммуникационных систем, контроллеров, интеллектуальных исполнительных механизмов.

Основой, необходимым условием эффективной реализации диспетчерского управления, имеющего ярко выраженный динамический характер, становится работа с информацией, т. е. процессы сбора, передачи, обработки, отображения, представления информации. От диспетчера уже требуется не только профессиональное знание технологического процесса, основ управления им, но и опыт работы в информационных системах, умение принимать решение (в диалоге с ЭВМ) в нештатных и аварийных ситуациях и многое другое. Диспетчер становится главным действующим лицом в управлении технологическим процессом.

Говоря о диспетчерском управлении, нельзя не затронуть проблему технологического риска. Технологические процессы в энергетике, нефтегазовой и ряде других отраслей промышленности являются потенциально опасными и при возникновении аварий приводят к человеческим жертвам,

а также к значительному материальному и экологическому ущербу. Статистика говорит, что за тридцать лет число учтенных аварий удваивается примерно каждые десять лет. В основе любой аварии за исключением стихийных бедствий лежит ошибка человека.

В результате анализа большинства аварий и происшествий на всех видах транспорта, в промышленности и энергетике были получены интересные данные. В 60 - х годах ошибка человека была первоначальной причиной аварий лишь в 20% случаев, тогда как к концу 80-х доля "человеческого фактора" стала приближаться к 80 %.

Одна из причин этой тенденции - старый традиционный подход к построению сложных систем управления, т. е. ориентация на применение новейших технических и технологических достижений и недооценка необходимости построения эффективного человеко - машинного интерфейса, ориентированного на человека (диспетчера).

Таким образом, требование повышения надежности систем диспетчерского управления является одной из предпосылок появления нового подхода при разработке таких систем: ориентация на оператора/диспетчера и его задачи.

Концепция SCADA (Supervisory Control And Data Acquisition - диспетчерское управление и сбор данных) предопределена всем ходом развития систем управления и результатами научно-технического прогресса. Применение SCADA-технологий позволяет достичь высокого уровня автоматизации в решении задач разработки систем управления, сбора, обработки, передачи, хранения и отображения информации.

Дружественность человеко-машинного интерфейса (HMI/MMI), предоставляемого SCADA - системами, полнота и наглядность представляемой на экране информации, доступность "рычагов" управления, удобство пользования подсказками и справочной системой и т. д. - повышает эффективность взаимодействия диспетчера с системой и сводит к нулю его критические ошибки при управлении.

Следует отметить, что концепция SCADA, основу которой составляет автоматизированная разработка систем управления, позволяет решить еще ряд задач, долгое время считавшихся неразрешимыми: сократить сроки разработки проектов по автоматизации и прямые финансовые затраты на их разработку.

В настоящее время SCADA является основным и наиболее перспективным методом автоматизированного управления сложными динамическими системами (процессами).

Управление технологическими процессами на основе систем SCADA стало осуществляться в передовых западных странах в 80-е годы. Область применения охватывает сложные объекты электро- и водоснабжения, химические, нефтехимические и нефтеперерабатывающие производства, железнодорожный транспорт, транспорт нефти и газа и др.

В России диспетчерское управление технологическими процессами опиралось, главным образом, на опыт оперативно-диспетчерского персонала. Поэтому переход к управлению на основе SCADA-систем стал осуществляться несколько позднее. К трудностям освоения в России новой информационной технологии, какой являются SCADA-системы, относится как отсутствие эксплуатационного опыта, так и недостаток информации о различных SCADA-системах. В мире насчитывается не один десяток компаний, активно занимающихся разработкой и внедрением SCADA-систем. Каждая SCADA-система - это "know-how" компании и поэтому данные о той или иной системе не столь обширны.

Большое значение при внедрении современных систем диспетчерского управления имеет решение следующих задач:

- выбора SCADA-системы (исходя из требований и особенностей технологического процесса);
- кадрового сопровождения.

Выбор SCADA-системы представляет собой достаточно трудную задачу, аналогичную принятию решений в условиях многокритериальности, усложненную невозможностью количественной оценки ряда критериев из-за недостатка информации.

Подготовка специалистов по разработке и эксплуатации систем управления на базе программного обеспечения SCADA осуществляется на специализированных курсах различных

фирм, курсах повышения квалификации. В настоящее время в учебные планы ряда технических университетов начали вводиться дисциплины, связанные с изучением SCADA-систем. Однако специальная литература по SCADA-системам отсутствует; имеются лишь отдельные статьи и рекламные проспекты.

В результате выполнения практических данятий осваиваются следующие компетенции:

ОК-3 готовностью к саморазвитию, самореализации, использованию творческого потенциала
Знать методы саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
Уметь применять методы саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
Владеть методами саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
ПК-5 способностью разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования
Знать методы разработки функциональной, логической и технической организации автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
Уметь разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
Владеть способностью разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
ППК-1 - способностью разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством.
Знать методы совершенствования систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством.
Уметь разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством
Владеть способностью разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством

Лабораторная работа №1
«Создание программы на языке FBD»

Запустите Concept. Создайте новый проект File → New project. Выполните конфигурирование контроллера. Закройте окно PLC Configuration. Создайте новую секцию File → New section…, выберите язык FBD и введите имя секции (до 32 символов), которое должно быть уникальным для всего проекта и должно удовлетворять соглашениям стандарта IEC 61131-3 по имени, в противном случае появится сообщение об ошибках. Согласно стандарту, только буквы позволяются как первый символ имени секции. В результате описанных выше действий появится поле для размещения блоков из 23 строк и 30 столбцов.

В редакторе FBD на фоне плоскости окна видна некоторая логическая сетка. В процессе конфигурирования каждый FFB размещается в ячейках этой сетки. Если FFB помещаются вне ячейки сетки или в случае перекрытия их с другими FFB, то появляется сообщение об ошибке и FFB не будет размещен в этой ячейке. Внешние параметры, задаваемые в явном виде на входах/выходах FFB, могут пересекаться другим объектом, но не должны нарушать границы ячеек сетки.

Чтобы вставить FFB в секцию, достаточно набрать команду меню Objects → FFB selection... Диалоговое окно FFBs from Library будет открыто. С помощью командной кнопки Library... в этом диалоговом окне осуществляется выбор библиотеки, из которой будут выбираться FFB. Для отображения уже созданных DFB, чтобы выбрать один из них, используется командная кнопка DFB. Теперь поместите выбранный FFB в секцию.

Если линия связи, обеспечивающая соединение с другим FFB, зафиксирована, то это соединение будет контролироваться редактором FBD. Если же подобное соединение запрещено, то будет выдано сообщение о его запрещении и линия связи не будет сгенерирована. В процессе формирования связей между FFB допускаются перекрытия и перекрещивания с другими связями и FFB.

Базируясь на логике программы, входу/выходу FFB с помощью редактора переменных можно назначить: переменную, константу, литерал, прямой адрес. Окно, изображенное на рисунке 1 выводится по двойному щелчку левой кнопки мыши на входе FFB.

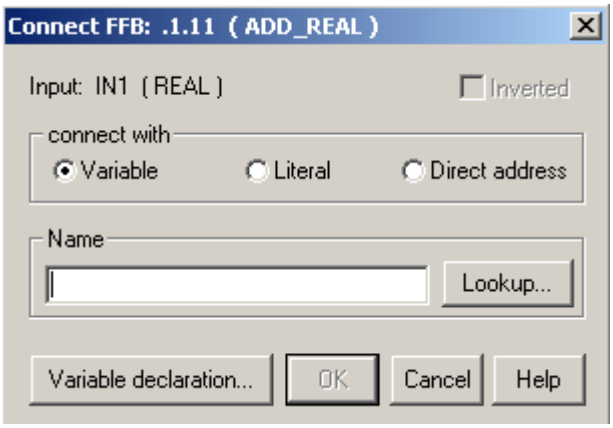


Рисунок 1 – Редактор переменных

Нелокализованная переменная, назначаемая входу/выходу, может использоваться как маркер, то есть для создания контуров или для передачи значений между различными секциями.

С помощью локализованной переменной, связанной с конкретным адресом, входу/выходу может быть назначен сигнал ввода/вывода аппаратного обеспечения.

Константа может быть помещена в другие секции.

Чтобы просмотреть список всех объявленных переменных или входов/выходов и сделать выбор из этого списка, используйте командную кнопку Lookup.

Если переменная не была объявлена, используйте команду Project → Variable declarations... или командную кнопку  , или кнопку F8 – функциональной части клавиатуры, для того, чтобы вызывать диалоговое окно Variable Editor для объявления нелокализованных переменных, констант, входных или выходных параметров. В этом окне можно объявить имя, тип данных, начальное значение и/или комментарий для нелокализованных переменных. Для констант объявляется имя, тип данных, значение и, в случае необходимости, комментарий.

Текстовые объекты не могут перекрываться с FFB, но могут перекрываться с линиями связи. Текстовые объекты не занимают память в ПЛК, потому что они, как правило, не загружаются в ПЛК.

Сохраните FBD-секцию с помощью команды меню File → Save project.

Пример 1. Создание программы на языке FBD.

Составим программу на языке FBD, реализующую систему управления перемещением горизонтального крана. Имеется три дискретных входа системы управления («кнопки без памяти»):

ON_LE – начать перемещение влево;

ON_RI – начать перемещение вправо;

STOP – остановить движение.

Имеется два выхода для исполнительных механизмов:

MOT_LE – перемещение крана влево;

MOT_RI – перемещение крана вправо.

Двигаясь в одном направлении, кран не воспринимает команду изменения направления перемещения. Чтобы изменить направление перемещения, кран необходимо остановить.

Одновременная подача единичных изменений на выходы не допускается.

На рисунке 3 представлено решение этой задачи на языке FBD (файл KRANFBD1.PRJ). На рисунке 4 представлено окно редактора переменных согласно условиям задачи.

Описание блока RS из группы Bistable библиотеки IEC. Блок RS работает по принципу RS-триггера. Пример применения блока RS смотрите в части I методических указаний к лабораторным работам.

Общий вид блока представлен на рисунке 2.

Рисунок 2 – Общий вид блока RS

Описание параметров блока:

Параметр	Тип данных	Назначение
S	BOOL	Установка
R1	BOOL	Доминирующий сброс
Q1	BOOL	Выход

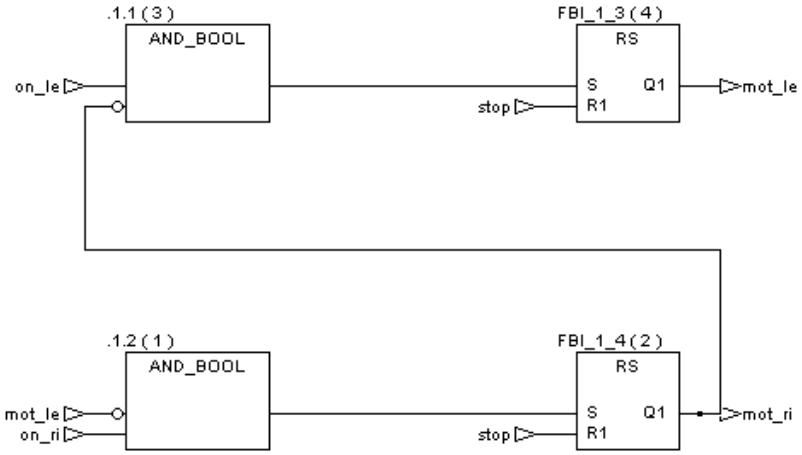


Рисунок 3 – Решение на языке FBD

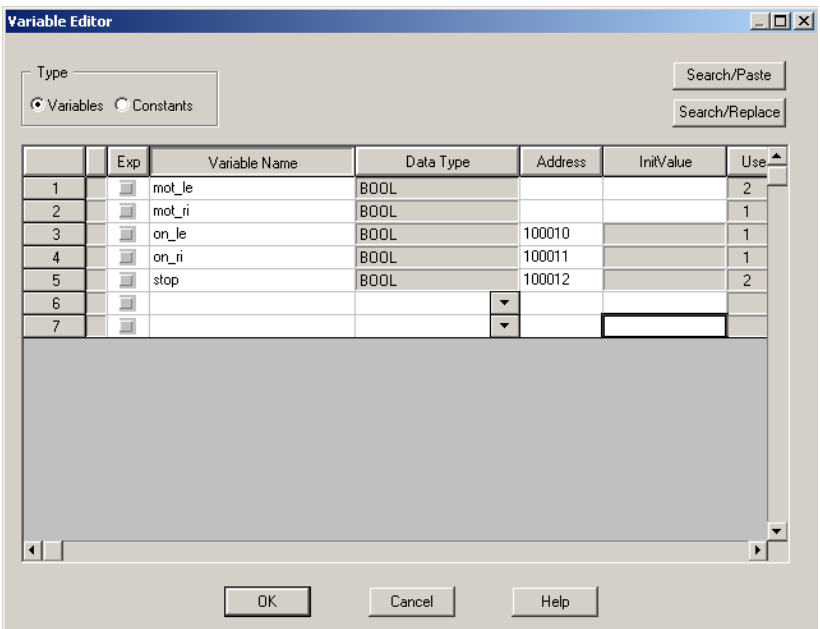


Рисунок 4 – Окно редактора переменных

Задание 1.

Создать программу на языке FBD согласно условиям задачи примера 1, изменив условия задачи следующим образом:

1) Время перемещения в каждом направлении нужно контролировать. Перемещение влево должно прекращаться через 5 с, а перемещение вправо – через 10 с.

2) Направление перемещения изменяется автоматически по истечении указанного времени.

Рекомендации: используйте стандартный блок TON из библиотеки Timer.

Описание блока TON. Таймер. Общий вид блока приведен на рисунке 5.

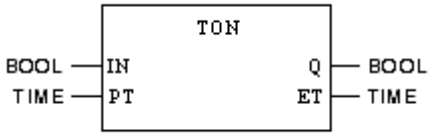


Рисунок 5 – Общий вид блока TON

Описание параметров блока

Параметр	Тип данных	Назначение
IN	BOOL	Начало задержки
PT	TIME	Установка времени задержки
Q	BOOL	Выход
ET	TIME	Внутреннее время

Решение представлено на рисунке 6. На рисунке 7 представлено окно редактора переменных.

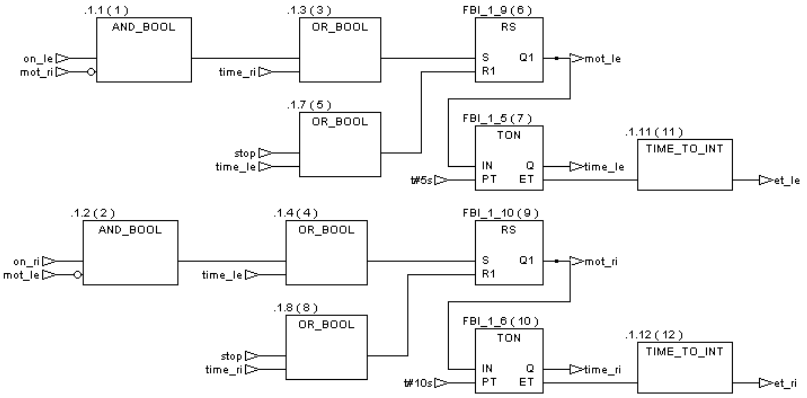


Рисунок 6 – Решение на языке FBD

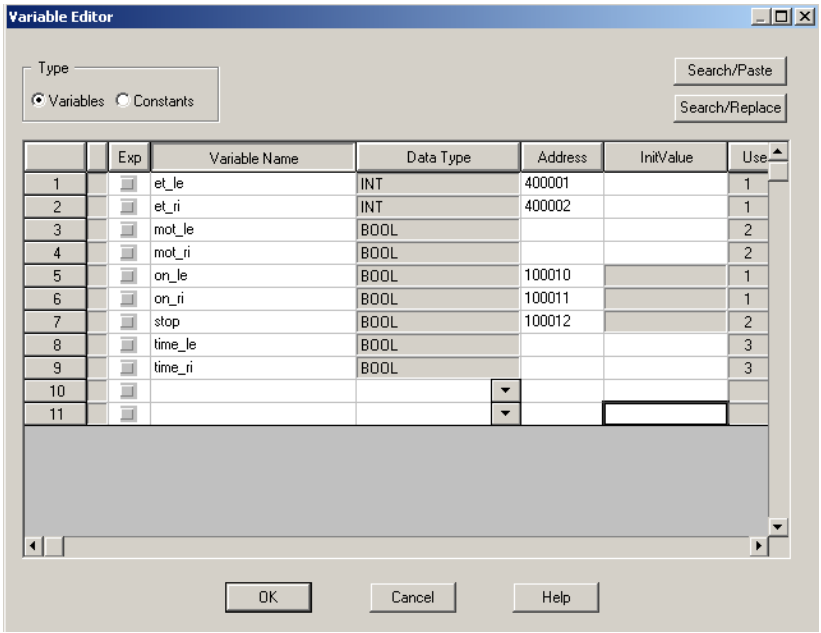


Рисунок 7 – Окно редактора переменных

Контрольные вопросы

1. Охарактеризуйте язык функциональных блок-схем FBD.
2. Что такое EFB, DFB, UDEFB?
3. Каково назначение входов EN и выходов ENO функциональных блоков?
4. Какую структуру имени, присваиваемого автоматически, имеет FFB?
5. Для чего служат связи?
6. Какое значение назначается по умолчанию несвязанным входам FFB?

Лабораторная работа №2
«Создание программы на языке LD»

Запустите Concept. Создайте новый проект File → New project. Выполните конфигурирование контроллера. Закройте окно PLC Configuration. Для создания программы на языке LD необходимо создать новую LD-секцию и ввести имя секции. Появится информационное поле для размещения элементов языка LD на 230 строк и 52 столбца.

Имя секции (до 32 символов) должно быть уникальным для всего проекта и должно удовлетворять соглашениям стандарта по имени, в противном случае появится сообщение об ошибках. Если введенное имя секции уже существует, Вы будете предупреждены, и должны выбрать другое имя.

В LD-редакторе фон окна является логической сеткой, которая показывает так называемую левую шину питания на левой стороне. Эта левая шина питания соответствует фазе (L-шина) ступени.

Как и в ступени, только те объекты LD (контакты, катушки), которые соединены с источником питания, т. е. подключены к левой шине питания, будут обработаны во время программирования секции LD. Правая шина питания, которая соответствует нейтральному проводнику, оптически не отображается. Но внутренне все катушки и выходы FFB подключены к ней, что обеспечивает протекание тока.

Чтобы вставить контакт или катушку в секцию, нужно открыть главное меню Objects и выбрать желаемые контакт или катушку. Контакты и катушки могут также быть выбраны в инструментальной панели.

Поместите контакты или катушки в секцию.

Чтобы вставить FFB в секцию, выберите команду меню Objects → FFB Selection... . Диалоговое окно FFBs from Library будет открыто. Используйте командную кнопку Library... в этом диалоговом окне, чтобы найти библиотеку, из которой можно выбрать FFB. Вы можете также использовать командную кнопку <DFB> для отображения созданных блоков DFB, чтобы выбрать один из них. Теперь поместите выбранный FFB в секцию.

Во время размещения объекты выравниваются в растре сетки. При размещении объектов снаружи фрейма секции или при перекрывании другим объектом появится сообщение об ошибках, и объект не будет размещен. При размещении контактов и катушек они автоматически связываются со смежными, несвязанными контактами и катушками, если контакт или катушка находится на той же самой вертикали. Связь с шиной питания будет установлена, только если контакт помещен поблизости. Если катушка или контакт помещены в уже существующую горизонтальную связь, то она автоматически будет разорвана и контакт или катушка вставлены. Если размещаются фактические параметры, они могут накладываться на другой объект, но не так, чтобы нарушить границу фрейма секции. Если связь служит как соединение с другим объектом, это соединение будет проверено. Если соединение не разрешено, то появится сообщение, и связь не будет сгенерирована.

После того как все блоки FFB будут размещены, закройте диалоговое окно.

Используйте команду Objects → Select, чтобы активизировать режим Mode и перемещать контакты, катушки и блок FFB к выбранной позиции.

Используйте команду Objects → Link, чтобы активизировать режим соединения и установить соединения между контактами, катушками и блоками FFB. Установите соединение между контактами, блоками FFB и левой шиной питания.

При создании связей разрешаются перекрытия другими связями и объектами и пересечения с ними. Если выбран блок FFB, его комментарий будет отображаться в первом столбце строки статуса. Если выбран фактический параметр, его имя (если оно есть), его прямой адрес и его комментарий будут отображаться в первом столбце строки статуса.

Несвязанные контакты, катушки и входы/выходы FFB имеют по умолчанию значение 0.

В дополнение к объектам, рассмотренным выше, текст также может быть помещен в LD-секцию. Размер этого текстового объекта зависит от длины текста. В зависимости от размера текста размер объекта может быть расширен на большее количество модулей сетки в вертикальном, а также в горизонтальном направлении. Текстовым объектам не разрешено накладываться на другие объекты, хотя они могут накладываться на связи.

Текстовые объекты не занимают место в ПЛК, потому что текст, как правило, не загружается в ПЛК.

Теперь используйте команду Objects → Select, чтобы повторно активизировать режим выбора, и дважды щелкните мышкой по контакту или катушке. Откроется диалоговое окно Properties: LD object, в котором контакту или катушке может быть назначен фактический параметр.

В зависимости от логики программы контакту или катушке и входам/выходам FFB могут быть назначены:

- переменная (локализованная или нелокализованная);
- константа;
- литерал;
- прямой адрес.

Сохраните LD-секцию с помощью команды меню File → Save project.

Пример 2. Создания программы на языке LD.

По условиям примера 1 требуется создать программу на языке LD.

На рисунке 8 представлено решение этой задачи на языке LD (файл KRAN_LD1.PRJ). На рисунке 9 представлено окно редактора данных согласно условиям задачи.

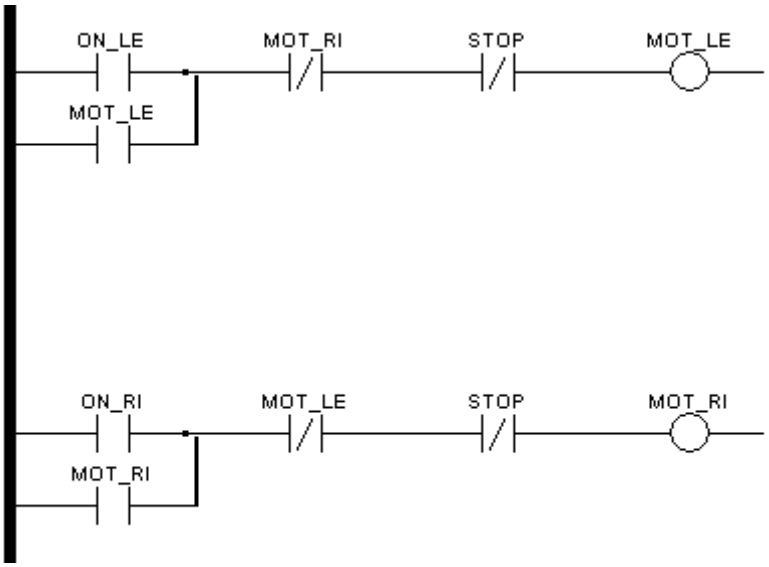


Рисунок 8 – Решение на языке LD

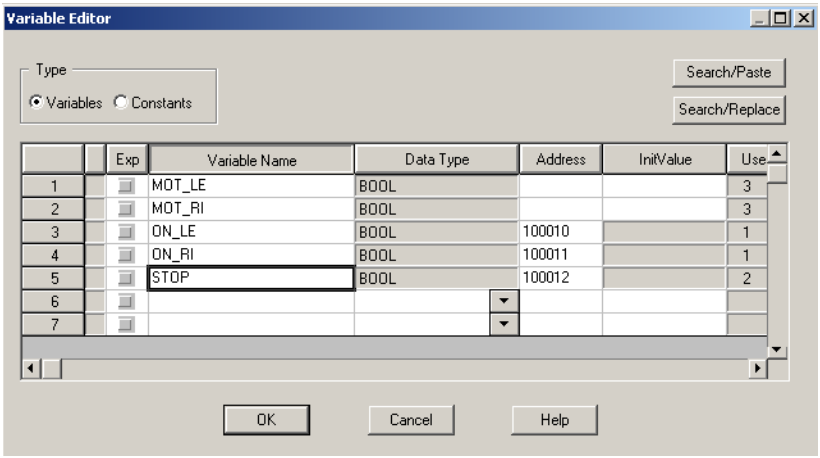


Рисунок 9 – Окно редактора переменных

Задание 2.

Создать программу на языке LD согласно условиям задания 1.

Решение представлено на рисунке 10 (файл KRAN_LD.PRJ). На рисунке 11 представлено окно редактора переменных.

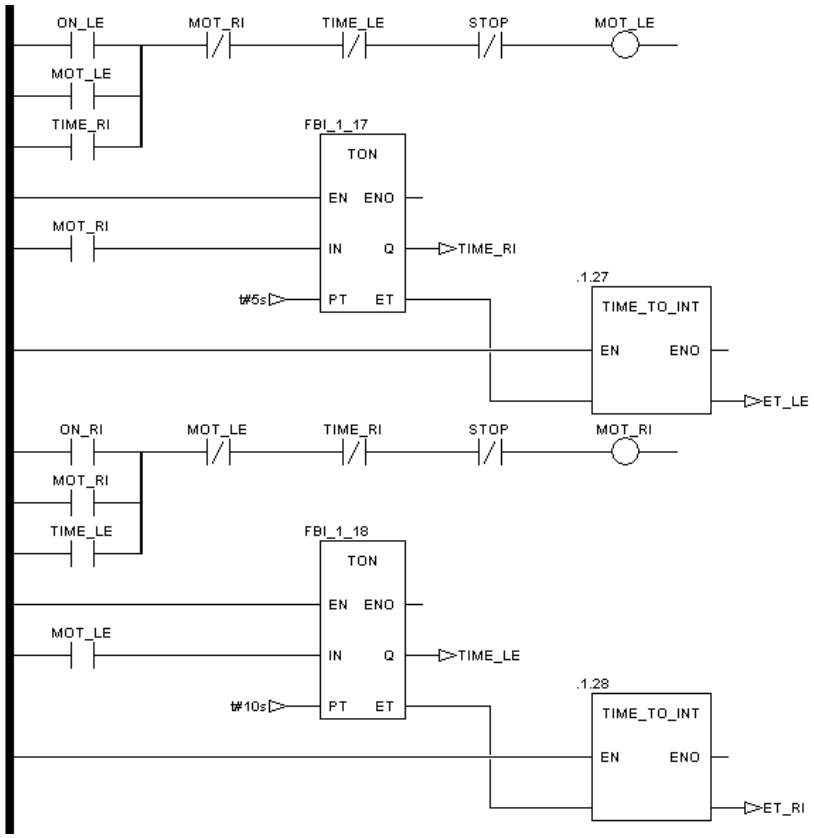


Рисунок 10 – Решение на языке LD

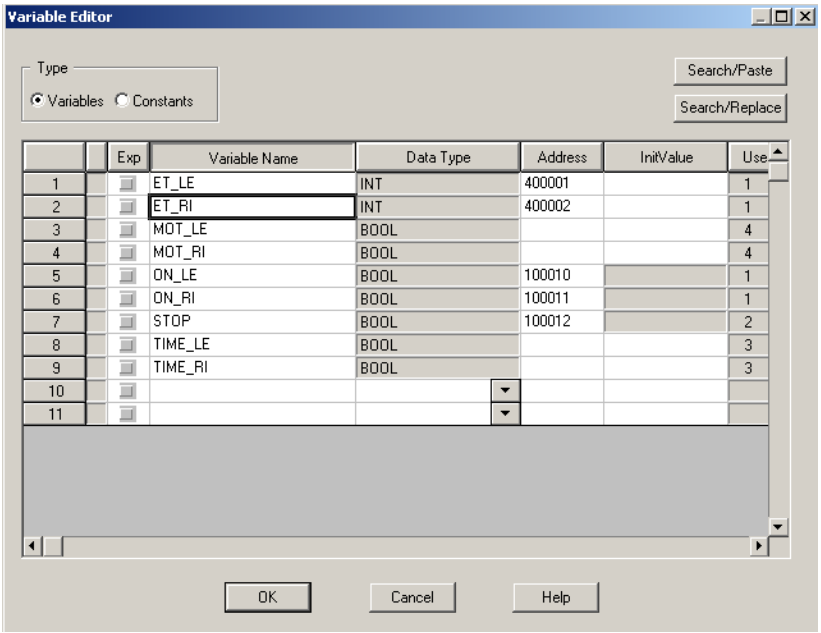


Рисунок 11 – Окно редактора переменных

Контрольные вопросы

1. Охарактеризуйте язык лестничной диаграммы LD.
2. Что такое катушка в языке LD?
3. Что такое контакт в языке LD?
4. Каким типом данных должен быть фактический параметр для контактов и катушек?
5. Для чего служат связи? Какие связи различают в языке LD?
6. Что является необходимым условием выполнения FFB в диаграмме LD?

Лабораторная работа №3
«Создание программы на языке SFC»

Запустите Concept. Создайте новый проект File → New project. Выполните конфигурирование контроллера. Закройте окно PLC Configuration. Создайте новую секцию File → New section…, выберите язык SFC и введите имя секции (до 32 символов), которое должно быть уникальным для всего проекта и должно удовлетворять соглашениям стандарта IEC 61131-3 по имени, в противном случае появится сообщение об ошибках. Согласно стандарту, только буквы допускается использовать как первый символ имени секции.

Фон окна в редакторе SFC – это логическая сетка на 200 строк и 32 столбца. Теоретически объекты SFC могут быть помещены в любую свободную ячейку. Если при этом создается связь с объектом (явно или размещением объекта в соседней ячейке), она будет проверена. При несанкционированном соединении появится сообщение об ошибке.

Вставка объектов. Объекты SFC (шаг, переход и т.д.) могут быть вставлены по отдельности или как группа согласно размеру секции с помощью команд главного меню Objects (последовательность шагов и переходов, структурированная параллельная последовательность и т.д.). То же самое можно сделать с помощью командных кнопок на панели инструментов.

Для выбора Объектов имеется несколько вариантов.

Для выбора одного объекта:

1) перейдите к режиму выбора;

2) позиционируйте указатель мыши на объект, который будет выбран, и щелкните левой кнопки мыши.

Для выбора нескольких объектов используется клавиша SHIFT.

Для вызова реквизитов шага сделайте двойной щелчок на шаге или выберите шаг и вызовите команду меню Objects → Properties, чтобы открыть диалоговое окно Step Properties (рисунок 12). В появившемся окне необходимо назначить действие шагу, если требуется – указать спецификатор, время задержки шага. Добавить новые определения к списку действий как новое действие с помощью командной кнопки New Action.

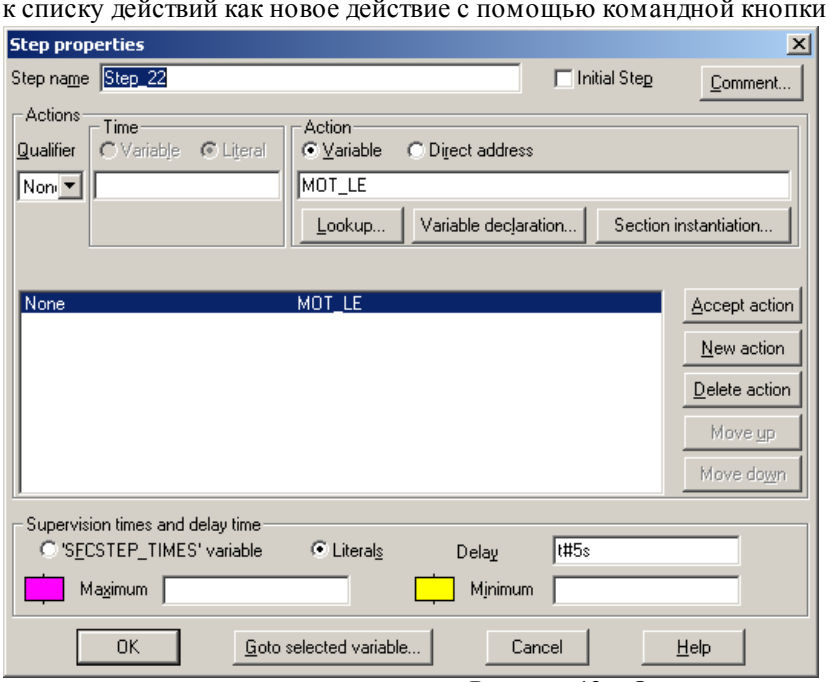


Рисунок 12 – Окно реквизитов шага

Для вызова реквизитов перехода сделайте двойной щелчок на переходе или выберите переход и вызовите команду меню Objects → Properties, чтобы открыть диалоговое окно Transition Properties.

Для вызова реквизитов прыжков сделайте двойной щелчок на прыжке или выберите переход и вызовите команду меню Objects → Properties, чтобы открыть диалоговое окно Jump Properties.

Объявите переменные и их начальные значения в редакторе переменных Project → Variable Editor...

Создайте логику программы.

Сохраните SFC-секцию с помощью команды меню File → Save project.

Пример 3. Программирование на языке SFC.

По условиям задачи примера 1 требуется создать программу на языке SFC.

Возможное решение задачи представлено на рисунке 13 (файл KRANSFC1.PRJ). На рисунке 14 представлено окно редактора переменных

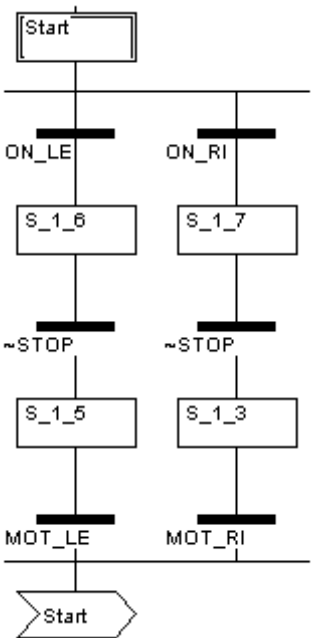


Рисунок 13 – Решение на языке SFC

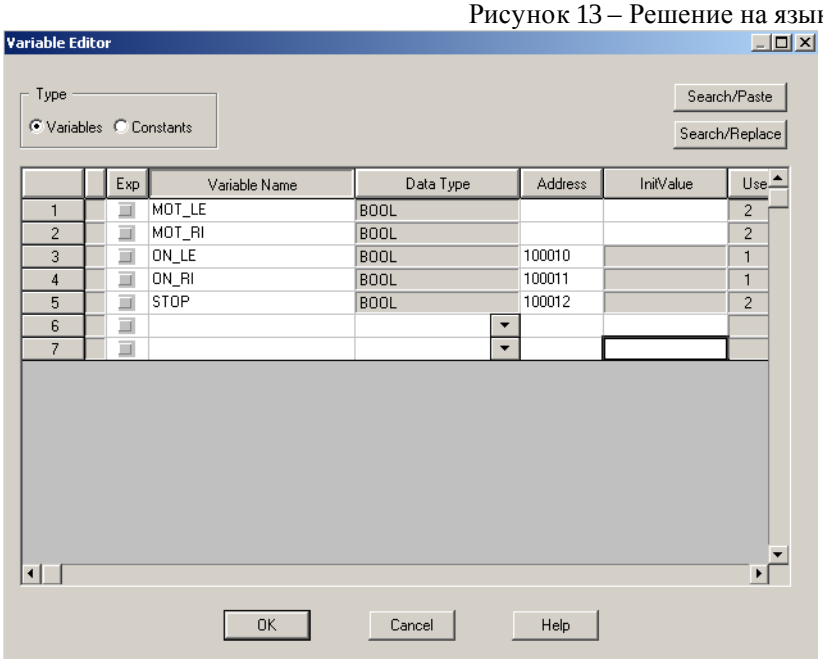


Рисунок 14 – Окно редактора переменных

Задание 3.

Создать программу на языке SFC согласно условиям задания 1.

Возможное решение представлено на рисунке 15 (файл KRAN_SFC.PRJ). На рисунке 16 представлено окно редактора переменных.

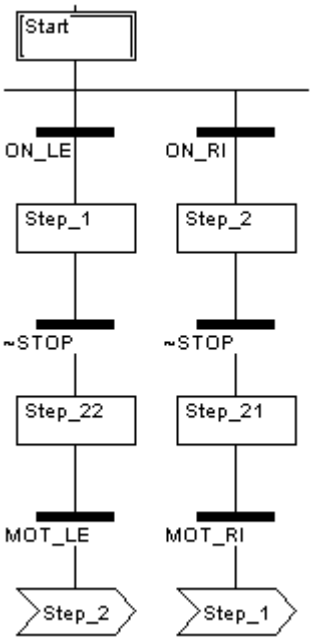


Рисунок 15 – Решение на языке SFC

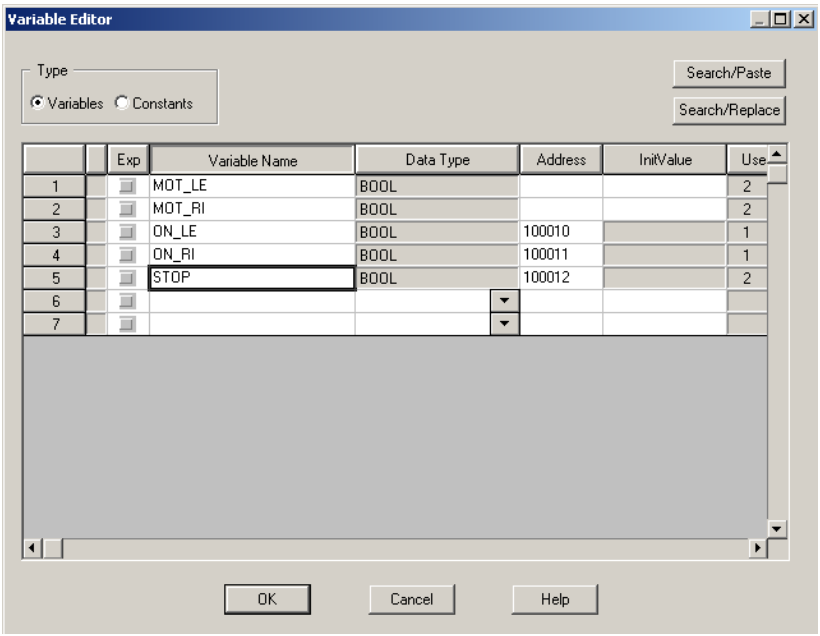


Рисунок 16 – Окно редактора переменных

Контрольные вопросы

1. Охарактеризуйте язык функционального управления SFC.
2. Для чего служат спецификаторы в языке SFC?
3. Каково назначение переходов в языке SFC?
4. Что такое секция перехода и ее назначение в языке SFC?
5. Назовите основные элементы языка SFC.
6. Что такое альтернативное (параллельное) соединение (ответвление)?

Лабораторная работа №4
«Создание программы на языке ST»

Запустите Concept. Создайте новый проект File → New project. Выполните конфигурирование контроллера. Закройте окно PLC Configuration. Для создания секции используйте команду меню File → New Section... и введите имя секции.

Имя секции (до 32 символов) должно быть уникальным для всего проекта. Имя секции должно удовлетворять соглашениям стандарта по имени иначе появится сообщение об ошибках.

Согласно стандарту только буквы допускается использовать как первый символ имени. Чтобы разрешить числа как первый символ, используйте команду меню Options → Preferences → EEC Extensions → IEC Extensions → Allow leading digits in identifiers.

При создании программы используйте команду VAR...END_VAR для объявления функциональных блоков и блоков DFB, с которыми вы хотите работать.

Объявите переменные и их начальные значения в редакторе переменных.

Пример логики программы:

```
SUM : = 0 ;
FOR I := 1 TO 3 DO
FOR J := 1 TO 2 DO
IF FLAG=1 THEN EXIT;
END_IF ;
SUM := SUM + J ;
 END_FOR;
SUM := SUM +1;
END_FOR
```

Команда условия выбора меню доступна, только если имеется открытая текстовая секция (IL-редактор, ST-редактор или редактор типов данных).

Используйте команду меню Edit → Expand Statement, чтобы завершить утверждение. Для ее вызова курсор мыши должен быть помещен на начальное утверждение (например, VAR, IF, CASE и т. д.).

Пример ST-программы:

Шаг 1 Введите IF.

Шаг 2 Выполните команду меню *Expand statement*.

Реакция Утверждение будет завершено с:

```
IF THEN
ELSEIF  THEN
ELSE
END_IF;
```

Используйте команду меню Edit → Insert Text File..., чтобы открыть стандартное диалоговое окно Windows Open для загрузки файла ASCII в текстовую секцию.

Если нужно также разрешить символы строчных букв, используйте диалоговое окно Options → Preferences → IEC Extensions... → IEC Extensions с опцией Allow case insensitive keywords.

Сохраните ST-секцию с помощью команды меню File → Save project.

Пример 4. Программирование на языке ST.

По условиям примера 1 требуется создать программу на языке ST.

Возможное решение задачи представлено ниже (файл KRAN_ST1.PRJ):

```
mot_le:=(on_le OR mot_le) AND NOT mot_ri AND NOT stop;
mot_ri:=(on_ri OR mot_ri) AND NOT mot_le AND NOT stop;
```

На рисунке 17 представлено окно редактора переменных

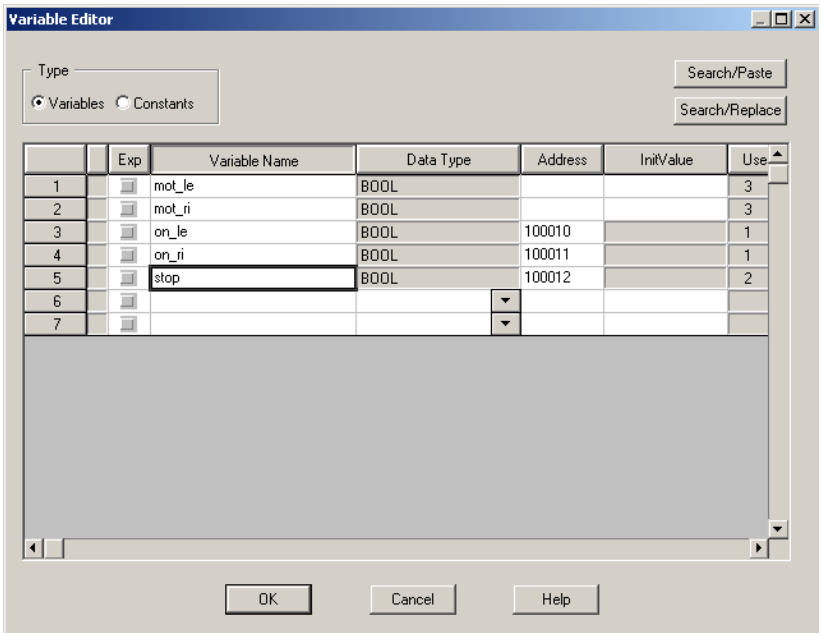


Рисунок 17 – Окно редактора переменных

Задание 4.

Создать программу на языке ST согласно условиям задания 1.

Возможное решение представлено ниже (файл KRAN_ST.PRJ):

```
VAR
FBI_1_10:TON;
FBI_1_15:RS;
FBI_1_9:TON;
FBI_1_3:RS;
END_VAR
FBI_1_3(S:=on_le AND NOT mot_ri OR time_ri, R1:=stop OR time_le);
mot_le:=FBI_1_3.Q1;
FBI_1_9(IN:=FBI_1_3.Q1,PT:=t#5s);
time_le:=FBI_1_9.Q;
et_le:=TIME_TO_INT(IN:=FBI_1_9.ET);
FBI_1_15(S:=on_ri AND NOT mot_le OR time_le, R1:=stop OR time_ri);
mot_ri:=FBI_1_15.Q1;
FBI_1_10(IN:=FBI_1_15.Q1,PT:=t#10s);
time_ri:=FBI_1_10.Q;
et_ri:=TIME_TO_INT(IN:=FBI_1_10.ET);
```

На рисунке 18 представлено окно редактора переменных

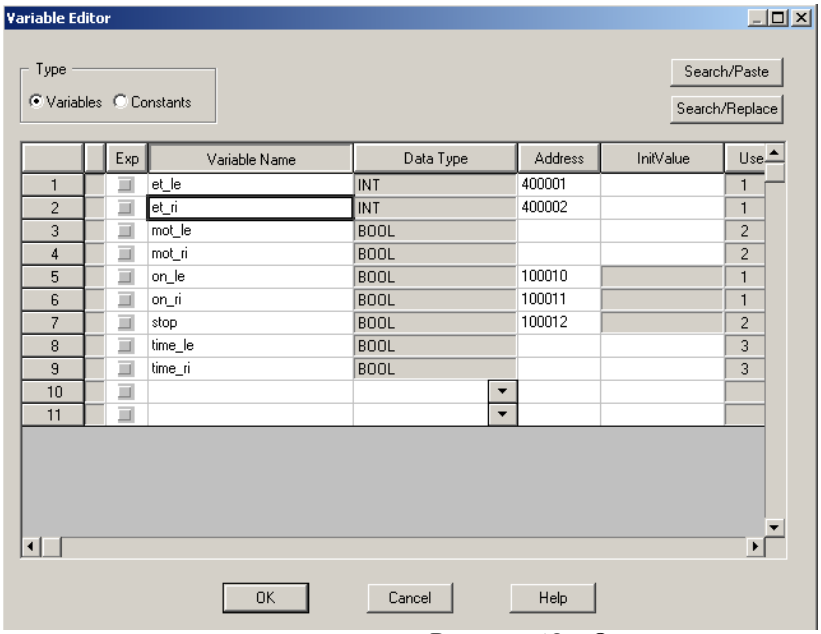


Рисунок 18 – Окно редактора переменных

Контрольные вопросы

1. Охарактеризуйте язык структурированного текста ST.
2. С помощью какой команды объявляются блоки FB/DFB в языке ST?
3. Что такое оператор в языке ST?
4. Что такое операнд в языке ST?
5. Что такое утверждение в языке ST?
6. Какое утверждение используется в языке ST для завершения утверждения повторения (FOR, WHILE, REPEAT) прежде, чем конечное условие будет выполнено?

Лабораторная работа №5
«Создание программы на языке IL»

Запустите Concept. Создайте новый проект File → New project. Выполните конфигурирование контроллера. Закройте окно PLC Configuration. Создайте новую секцию File → New section…, выберите язык IL и введите имя секции (до 32 символов), которое должно быть уникальным для всего проекта и должно удовлетворять соглашениям стандарта IEC 61131-3 по имени, в противном случае появится сообщение об ошибках. Согласно стандарту, только буквы допускается использовать как первый символ имени секции.

Используйте команду VAR…END_VAR для объявления функциональных блоков и блоков DFB, с которыми предстоит работать.

Объявите переменные и их начальные значения в редакторе переменных. Project → Variable Editor...

Создайте логику программы.

Сохраните IL-секцию с помощью команды меню File → Save project.

Пример 5. Программирование на языке IL.

По условиям задачи примера 1 требуется создать программу на языке IL.

Возможное решение задачи представлено ниже (файл KRAN_IL1.PRJ):

```
LD on_le
OR mot_le
ANDN mot_ri
AND stop
ST mot_le
LD on_ri
OR mot_ri
ANDN mot_le
AND stop
ST mot_ri
LD on_le
ANDN mot_ri
S mot_le
LD stop
R mot_le
LD on_ri
ANDN mot_le
S mot_ri
LD stop
R mot_ri
```

На рисунке 19 представлено окно редактора переменных

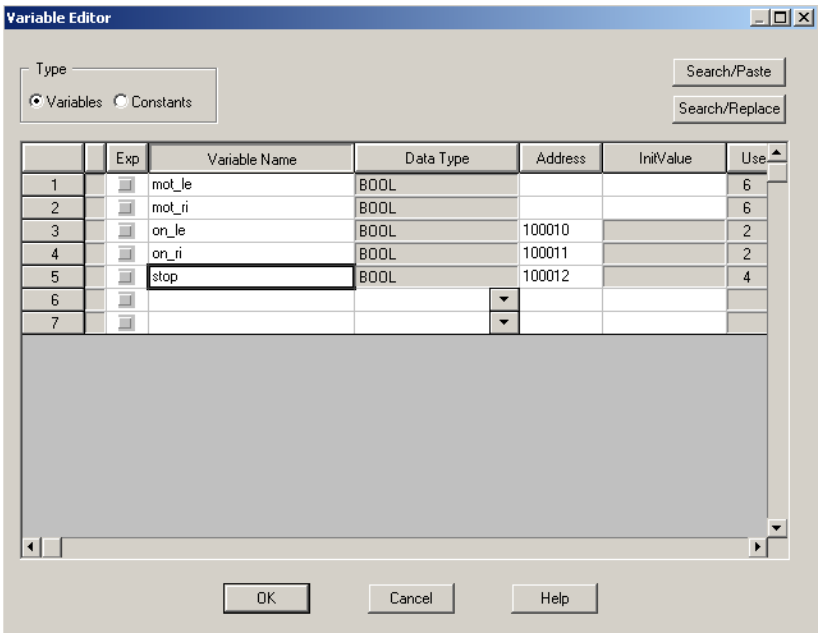


Рисунок 19 – Окно редактора переменных

Задание 5.

Создать программу на языке IL согласно условиям задания 1.

Возможное решение представлено ниже (файл KRAN_IL.PRJ):

```
VAR
FBI_1_10:TON;
FBI_1_15:RS;
FBI_1_9:TON;
FBI_1_3:RS;
END_VAR
LD on_le
ANDN mot_ri
OR time_ri
ST FBI_1_3.S
LD stop
OR time_le
ST FBI_1_3.R1
CAL FBI_1_3
LD FBI_1_3.Q1
ST mot_le
LD FBI_1_3.Q1
ST FBI_1_9.IN
LD t#5s
ST FBI_1_9.PT
CAL FBI_1_9
LD FBI_1_9.q
ST time_le
LD FBI_1_9.ET
TIME_TO_INT
ST et_le
LD on_ri
ANDN mot_le
OR time_le
ST FBI_1_15.S
LD stop
OR time_ri
ST FBI_1_15.R1
CAL FBI_1_15
LD FBI_1_15.Q1
ST mot_ri
LD FBI_1_15.Q1
ST FBI_1_10.IN
LD t#10s
ST FBI_1_10.PT
CAL FBI_1_10
LD FBI_1_10.Q
ST time_ri
LD FBI_1_10.ET
TIME_TO_INT
ST et_ri
```

На рисунке 20 представлено окно редактора переменных

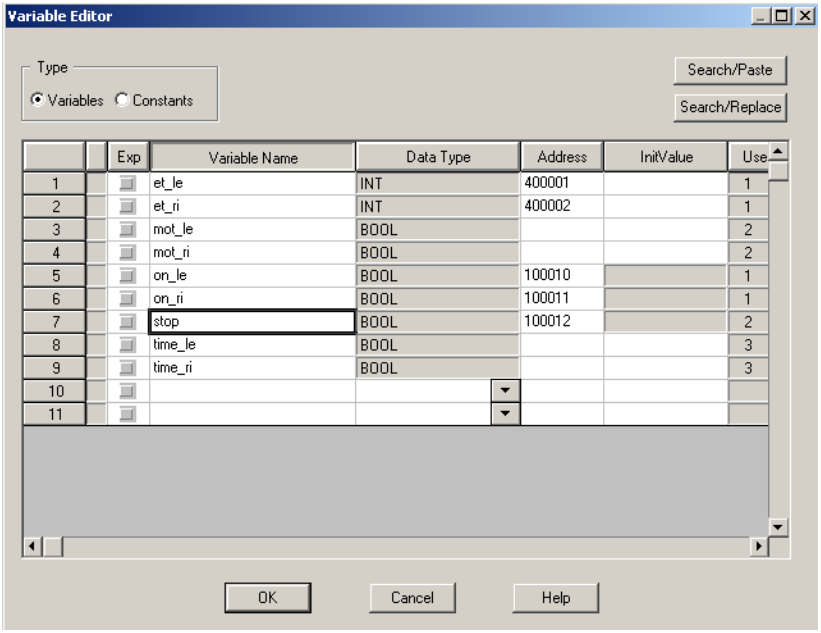


Рисунок 20 – Окно редактора переменных

Контрольные вопросы

1. Охарактеризуйте язык списка инструкций IL.
2. В чем назначение модификаторов в языке IL?
3. Назовите три способа для вызова функциональных блоков в языке IL.
4. Для чего служат модификаторы в языке IL?
5. Что такое оператор в языке IL?
6. С помощью какой команды производят прямое объявление адресов в языке IL?

Литература

Перечень основной литературы:

1 Цифровые системы автоматизированного проектирования. SCADA-системы : учебное пособие / И. А. Елизаров, А. А. Третьяков, А. Н. Пчелинцев [и др.]. — Тамбов : Тамбовский государственный технический университет, ЭБС АСВ, 2015. — 160 с. — ISBN 978-5-8265-1469-6. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/63849.html

2 Компьютерные технологии при проектировании и эксплуатации технологического оборудования : учебное пособие / Г. В. Алексеев, И. И. Бриденко, В. А. Головацкий, Е. И. Верболоз. — Саратов : Вузовское образование, 2017. — 171 с. — ISBN 978-5-4487-0004-0. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/65620.html

Перечень дополнительной литературы:

1 Алексеев, Г. В. Возможности интерактивного проектирования технологического оборудования : учебное пособие / Г. В. Алексеев. — 2-е изд. — Саратов : Вузовское образование, 2021. — 263 с. — ISBN 978-5-4487-0377-5. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/79618.html

2 Бойков, В. И. Цифровые системы автоматизированного проектирования / В. И. Бойков, Г. И. Болтунов, О. К. Мансурова. — СПб. : Университет ИТМО, 2010. — 161 с. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/68653.html

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
НЕВИННОМЫССКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ
(ФИЛИАЛ) СКФУ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению лабораторных работ по дисциплине
«Цифровые системы автоматизированного проектирования»
Методические указания к выполнению лабораторных работ.
Часть 2

Направление подготовки 15.04.04
«Автоматизация технологических процессов и производств»
Направленность (профиль) «Информационно-управляющие системы»
Форма обучения - очно-заочная
Год начала обучения 2022
Реализуется в 4 семестре

Невинномысск 2022

УДК 004.5(073)

Автор-составитель:
канд. техн. наук, доцент Э.Е. Тихонов

«Цифровые системы автоматизированного проектирования»: методические указания по выполнению лабораторных работ для студентов направления 15.04.04 Автоматизация технологических процессов и производств, Магистерская программа: Информационно-управляющие системы / автор – составитель: Э.Е. Тихонов - Невинномысск: НТИ (филиал) СКФУ, 2022. – 166 с.

Методические указания к выполнению лабораторных работ и по освоению дисциплины «Цифровые системы автоматизированного проектирования» предназначены для магистров очно-заочной формы бучения направления 15.04.04 Автоматизация технологических процессов и производств, Магистерская программа: Информационно-управляющие системы

Методические указания составлены на основании федерального государственного образовательного стандарта высшего образования по направлению подготовки 15.04.04 Автоматизация технологических процессов и производств.

Содержание

Введение

Современная АСУТП (автоматизированная система управления технологическим процессом) представляет собой многоуровневую человеко-машинную систему управления. Создание АСУ сложными технологическими процессами осуществляется с использованием автоматических информационных систем сбора данных и вычислительных комплексов, которые постоянно совершенствуются по мере эволюции технических средств и программного обеспечения.

Непрерывную во времени картину развития АСУТП можно разделить на три этапа, обусловленные появлением качественно новых научных идей и технических средств. В ходе истории меняется характер объектов и методов управления, средств автоматизации и других компонентов, составляющих содержание современной системы управления.

Первый этап отражает внедрение систем автоматического регулирования (САР). Объектами управления на этом этапе являются отдельные параметры, установки, агрегаты; решение задач стабилизации, программного управления, слежения переходит от человека к САР. У человека появляются функции расчета задания и параметры настройки регуляторов.

Второй этап- автоматизация технологических процессов. Объектом управления становится рассредоточенная в пространстве система; с помощью систем автоматического управления (САУ) реализуются все более сложные законы управления, решаются задачи оптимального и адаптивного управления, проводится идентификация объекта и состояний системы. Характерной особенностью этого этапа является внедрение систем телемеханики в управление технологическими процессами. Человек все больше отдаляется от объекта управления, между объектом и диспетчером выстраивается целый ряд измерительных систем, исполнительных механизмов, средств телемеханики, мнемосхем и других средств отображения информации (СОИ).

Третий этап- автоматизированные системы управления технологическими процессами - характеризуется внедрением в управление технологическими процессами вычислительной техники. Вначале - применение микропроцессоров, использование на отдельных фазах управления вычислительных систем; затем активное развитие человеко-машинных систем управления, инженерной психологии, методов и моделей исследования операций и, наконец, диспетчерское управление на основе использования автоматических информационных систем сбора данных и современных вычислительных комплексов.

От этапа к этапу менялись и функции человека (оператора/диспетчера), призванного обеспечить регламентное функционирование технологического процесса. Расширяется круг задач, решаемых на уровне управления; ограниченный прямой необходимостью управления технологическим процессом набор задач пополняется качественно новыми задачами, ранее имеющими вспомогательный характер или относящиеся к другому уровню управления.

Диспетчер в многоуровневой автоматизированной системе управления технологическими процессами получает информацию с монитора ЭВМ или с электронной системы отображения информации и воздействует на объекты, находящиеся от него на значительном расстоянии с помощью телекоммуникационных систем, контроллеров, интеллектуальных исполнительных механизмов.

Основой, необходимым условием эффективной реализации диспетчерского управления, имеющего ярко выраженный динамический характер, становится работа с информацией, т. е. процессы сбора, передачи, обработки, отображения, представления информации. От диспетчера уже требуется не только профессиональное знание технологического процесса, основ управления им, но и опыт работы в информационных системах, умение принимать решение (в диалоге с ЭВМ) в нештатных и аварийных ситуациях и многое другое. Диспетчер становится главным действующим лицом в управлении технологическим процессом.

Говоря о диспетчерском управлении, нельзя не затронуть проблему технологического риска. Технологические процессы в энергетике, нефтегазовой и ряде других отраслей промышленности являются потенциально опасными и при возникновении аварий приводят к человеческим жертвам,

а также к значительному материальному и экологическому ущербу. Статистика говорит, что за тридцать лет число учтенных аварий удваивается примерно каждые десять лет. В основе любой аварии за исключением стихийных бедствий лежит ошибка человека.

В результате анализа большинства аварий и происшествий на всех видах транспорта, в промышленности и энергетике были получены интересные данные. В 60 - х годах ошибка человека была первоначальной причиной аварий лишь в 20% случаев, тогда как к концу 80-х доля "человеческого фактора" стала приближаться к 80 %.

Одна из причин этой тенденции - старый традиционный подход к построению сложных систем управления, т. е. ориентация на применение новейших технических и технологических достижений и недооценка необходимости построения эффективного человеко - машинного интерфейса, ориентированного на человека (диспетчера).

Таким образом, требование повышения надежности систем диспетчерского управления является одной из предпосылок появления нового подхода при разработке таких систем: ориентация на оператора/диспетчера и его задачи.

Концепция SCADA (Supervisory Control And Data Acquisition - диспетчерское управление и сбор данных) предопределена всем ходом развития систем управления и результатами научно-технического прогресса. Применение SCADA-технологий позволяет достичь высокого уровня автоматизации в решении задач разработки систем управления, сбора, обработки, передачи, хранения и отображения информации.

Дружественность человеко-машинного интерфейса (HMI/MMI), предоставляемого SCADA - системами, полнота и наглядность представляемой на экране информации, доступность "рычагов" управления, удобство пользования подсказками и справочной системой и т. д. - повышает эффективность взаимодействия диспетчера с системой и сводит к нулю его критические ошибки при управлении.

Следует отметить, что концепция SCADA, основу которой составляет автоматизированная разработка систем управления, позволяет решить еще ряд задач, долгое время считавшихся неразрешимыми: сократить сроки разработки проектов по автоматизации и прямые финансовые затраты на их разработку.

В настоящее время SCADA является основным и наиболее перспективным методом автоматизированного управления сложными динамическими системами (процессами).

Управление технологическими процессами на основе систем SCADA стало осуществляться в передовых западных странах в 80-е годы. Область применения охватывает сложные объекты электро- и водоснабжения, химические, нефтехимические и нефтеперерабатывающие производства, железнодорожный транспорт, транспорт нефти и газа и др.

В России диспетчерское управление технологическими процессами опиралось, главным образом, на опыт оперативно-диспетчерского персонала. Поэтому переход к управлению на основе SCADA-систем стал осуществляться несколько позднее. К трудностям освоения в России новой информационной технологии, какой являются SCADA-системы, относится как отсутствие эксплуатационного опыта, так и недостаток информации о различных SCADA-системах. В мире насчитывается не один десяток компаний, активно занимающихся разработкой и внедрением SCADA-систем. Каждая SCADA-система - это "know-how" компании и поэтому данные о той или иной системе не столь обширны.

Большое значение при внедрении современных систем диспетчерского управления имеет решение следующих задач:

- выбора SCADA-системы (исходя из требований и особенностей технологического процесса);
- кадрового сопровождения.

Выбор SCADA-системы представляет собой достаточно трудную задачу, аналогичную принятию решений в условиях многокритериальности, усложненную невозможностью количественной оценки ряда критериев из-за недостатка информации.

Подготовка специалистов по разработке и эксплуатации систем управления на базе программного обеспечения SCADA осуществляется на специализированных курсах различных

фирм, курсах повышения квалификации. В настоящее время в учебные планы ряда технических университетов начали вводиться дисциплины, связанные с изучением SCADA-систем. Однако специальная литература по SCADA-системам отсутствует; имеются лишь отдельные статьи и рекламные проспекты.

В результате выполнения лабораторных работ осваиваются следующие компетенции:

ОК-3 готовностью к саморазвитию, самореализации, использованию творческого потенциала
Знать методы саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
Уметь применять методы саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
Владеть методами саморазвития, самореализации, использования творческого потенциала в задачах применения интегрированных систем проектирования и управления
ПК-5 способностью разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования
Знать методы разработки функциональной, логической и технической организации автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
Уметь разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
Владеть способностью разрабатывать функциональную, логическую и техническую организацию автоматизированных и автоматических производств, их элементов, технического, алгоритмического и программного обеспечения на базе современных методов, средств и технологий проектирования в задачах применения интегрированных систем проектирования и управления
ППК-1 - способностью разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством.
Знать методы совершенствования систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством.
Уметь разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством
Владеть способностью разрабатывать практические мероприятия по совершенствованию систем и средств автоматизации и управления изготовлением продукции, ее жизненным циклом и качеством

Лабораторная работа №1 Создание простейшего проекта в программе TRACE MODE

Цель работы – освоить методику создания системы мониторинга, содержащую один узел АРМ с использованием механизма автопостроения каналов TRACE MODE методом «от шаблона экрана».

Создание узла АРМ

Разработка любого проекта автоматизации всегда начинается с запуска Интегрированной среды разработки (ИСР). Для ее запуска необходимо выполнить команду TRACE MODE IDE 6 (base) из группы установки инструментальной системы в меню Программы WINDOWS или двойным щелчком ЛК мыши по иконке  рабочего стола Windows рисунок 11.

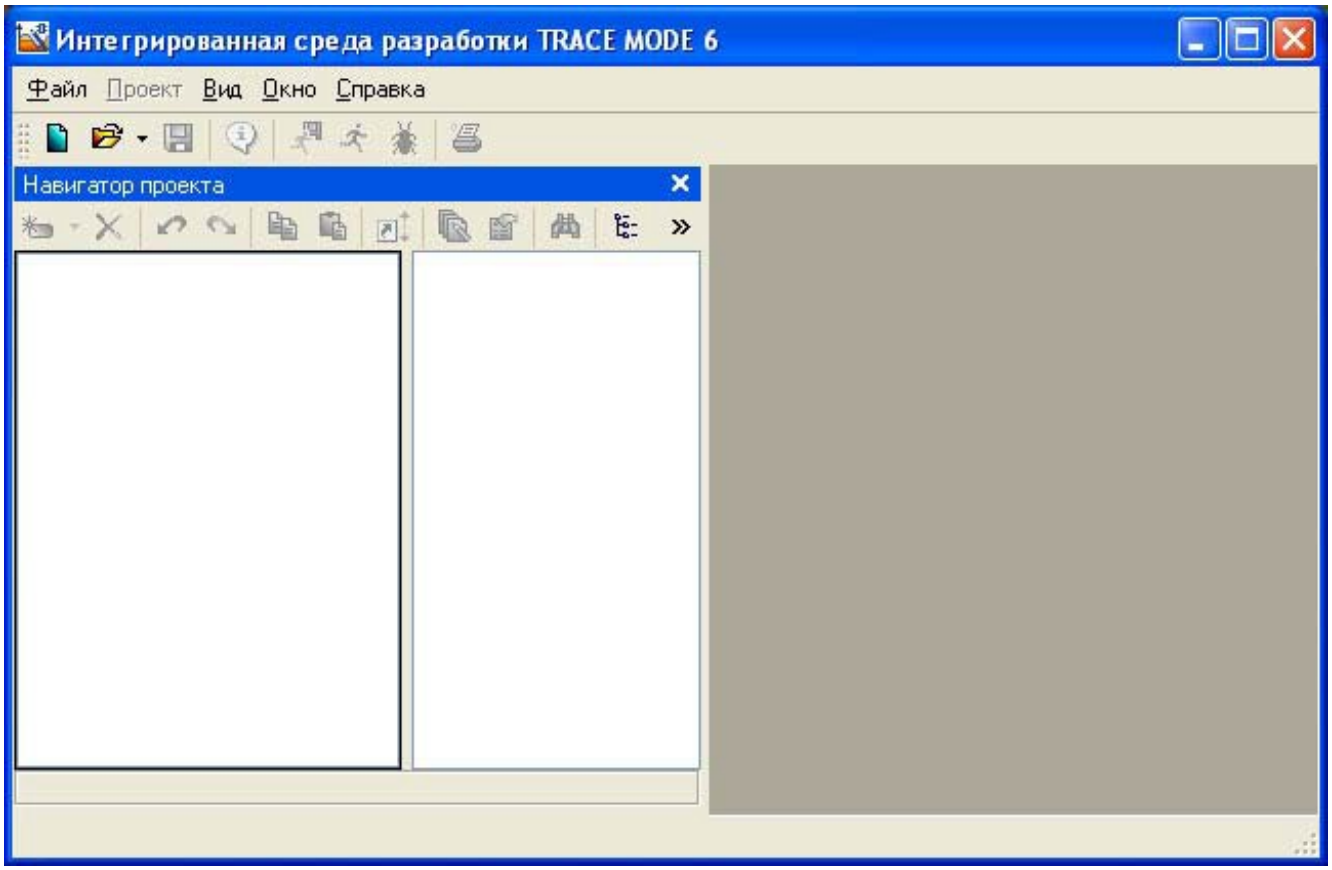


Рисунок 11– Интегрированная среда разработки

После запуска ИСР в меню **Файл** выбрать команду **Настройки ИС…**В появившемся окне выбрать **Уровень сложности** и настроить как показано на рисунке 12, а затем выбрать **Отладка** и настроить как на рисунке 13.

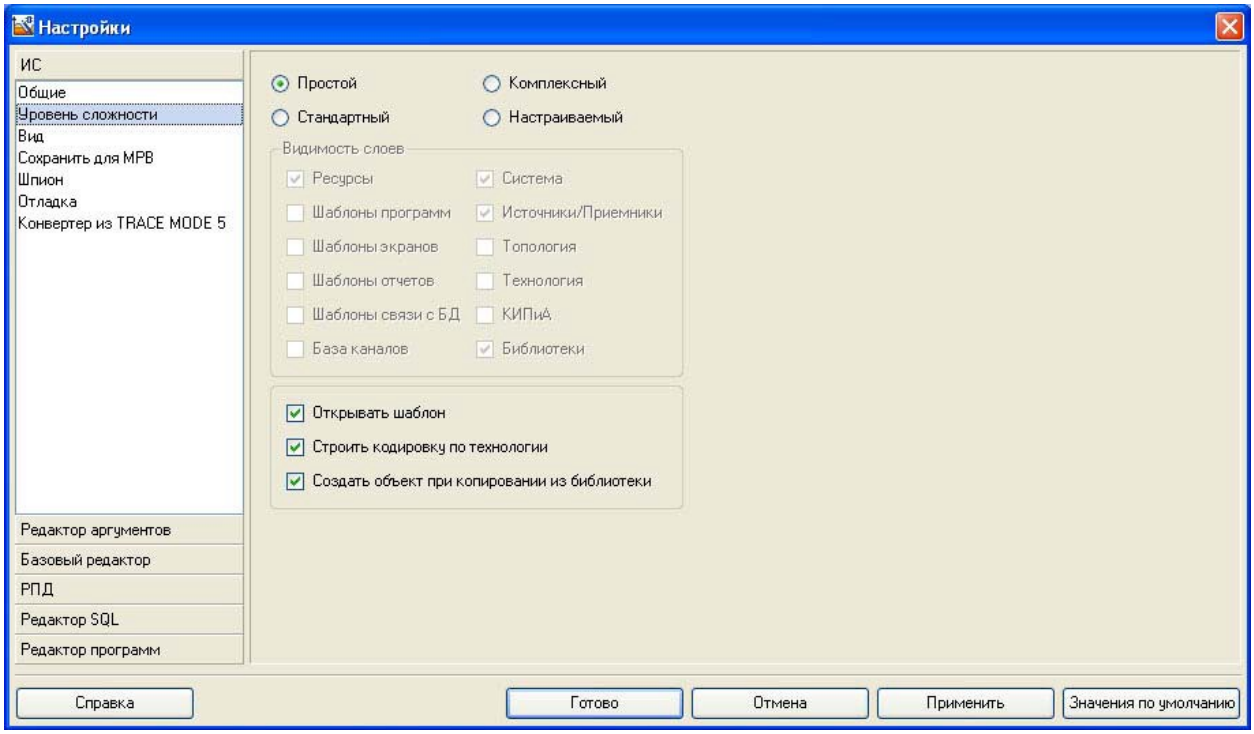


Рисунок 12 – Настройки ИСР

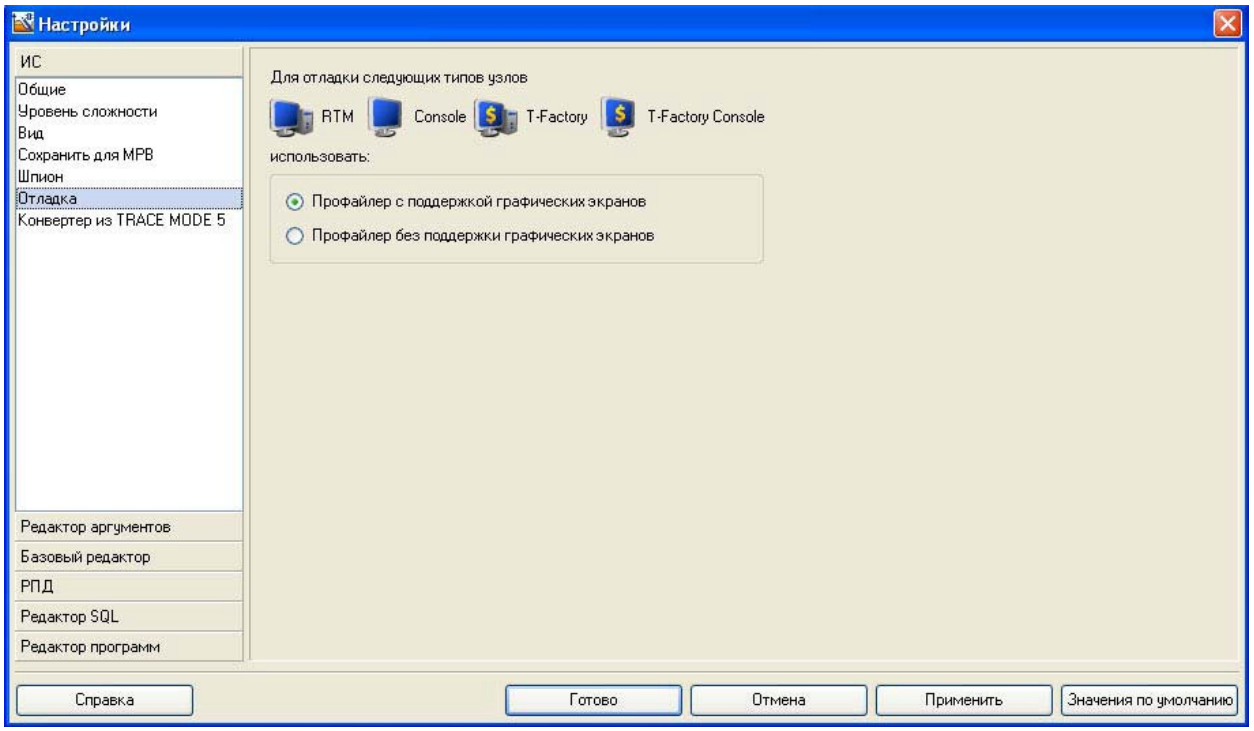


Рисунок 13 – Настройки ИСР

После проведенных настроек ИСР нажать кнопку Готово.

С помощью иконки  инструментальной панели создать новый проект, при этом в открывшемся на экране диалоге рисунок 14.

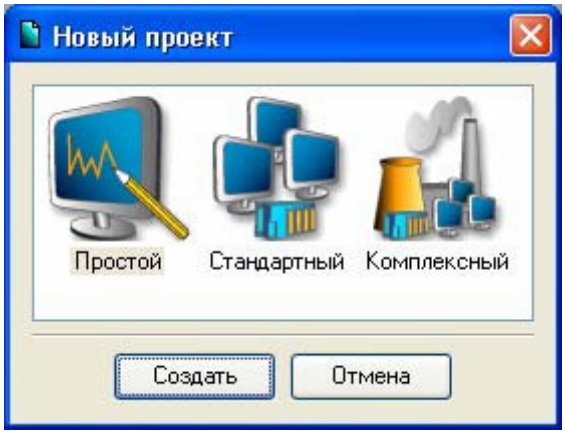


Рисунок 14 – Создание нового проекта

Выберем **Простой** стиль разработки. После нажатия левой клавиши мыши (ЛК) на экранной кнопке **Создать**, в левом окне Навигатора проекта появится дерево проекта с созданным узлом АРМ RTM_1 рисунок 15. Откроем узел RTM_1 двойным щелчком ЛК, в правом окне Навигатора проекта отобразится содержимое узла – пустая группа Каналы и один канал класса Вызов Экран#1, предназначенный для отображения на узле АРМ графического экрана;

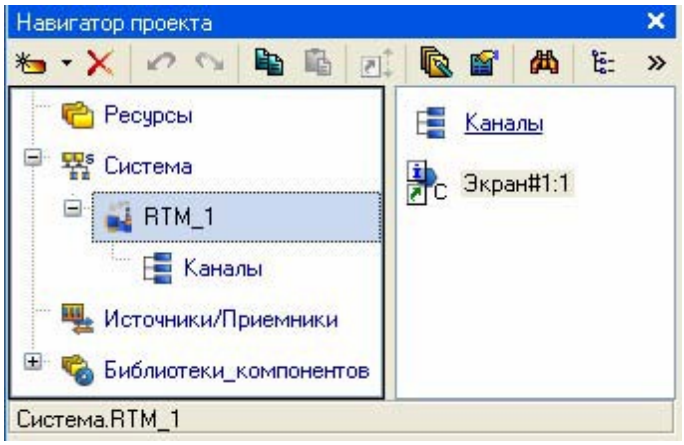


Рисунок 15 – Навигатор проекта

Создание графического экрана

Двойным щелчком ЛК на компоненте Экран#1 открыть окно графического редактора.

Создание статического текста

Разместим в левом верхнем углу экрана статический текст - надпись «Значение параметра». На панели инструментов графического редактора выделить иконку графического элемента (ГЭ) 

, на поле редактора установить прямоугольник ГЭ, для чего рисунок 16: зафиксировать ЛК «точку привязки»; развернуть прямоугольник движением курсора и зафиксировать выбранный ГЭ;

Рисунок 16 – Размещение статического текста

Для перехода в режим редактирования элемента выделить на панели инструментов иконку . Двойным щелчком ЛК по размещенному ГЭ открыть окно его свойств. В правом поле строки **Текст** набрать «Значение параметра» рисунок 17.

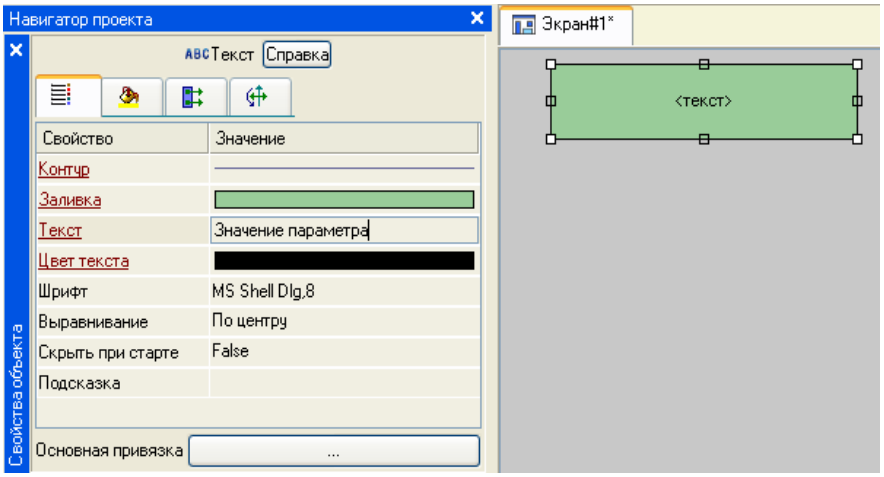


Рисунок 17 – Свойства статического текста Закрыть окно

свойств, ГЭ будет иметь вид рисунок 18:

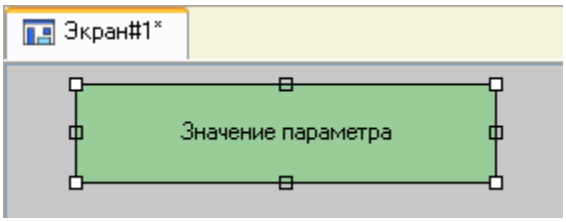


Рисунок 18 – Вид графического экрана

Если введенный Вами текст не уместился в прямоугольнике ГЭ, выделите его и растяните до нужного размера с помощью мыши.

Создание динамического текста, создание аргумента экрана в процессе настройки динамического текста

Подготовим на экране вывод динамического текста для отображения численного значения какого-либо источника сигнала – внешнего или внутреннего путем указания динамизации атрибута ГЭ. Определим назначение аргумента шаблона экрана. Создать и разместить новый ГЭ справа от ГЭ с надписью «Значение параметра». Двойным щелком ЛК на строке **Текст** вызвать меню **Вид индикации** рисунок 19.

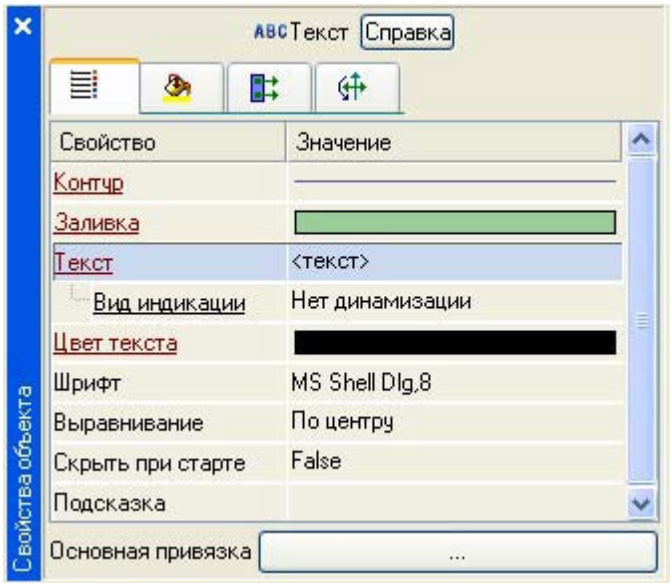


Рисунок 11 – Настройка динамизации9

В правом поле строки нажать ЛК и вызвать список доступных типов, выбрать тип **Значение** рисунок 20.

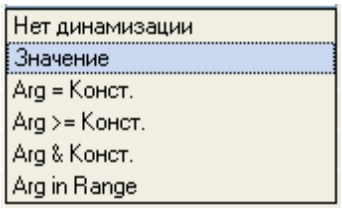


Рисунок 20 – Тип динамизации

В открывшемся меню настройки параметров динамизации рисунок 21 выбрать свойство **Привязка**.

Рисунок 21 – Настройка параметров динамизации

В открывшемся окне **Свойство привязки**, нажав кнопку  на его панели инструментов, создать аргумент экрана рисунок 22.

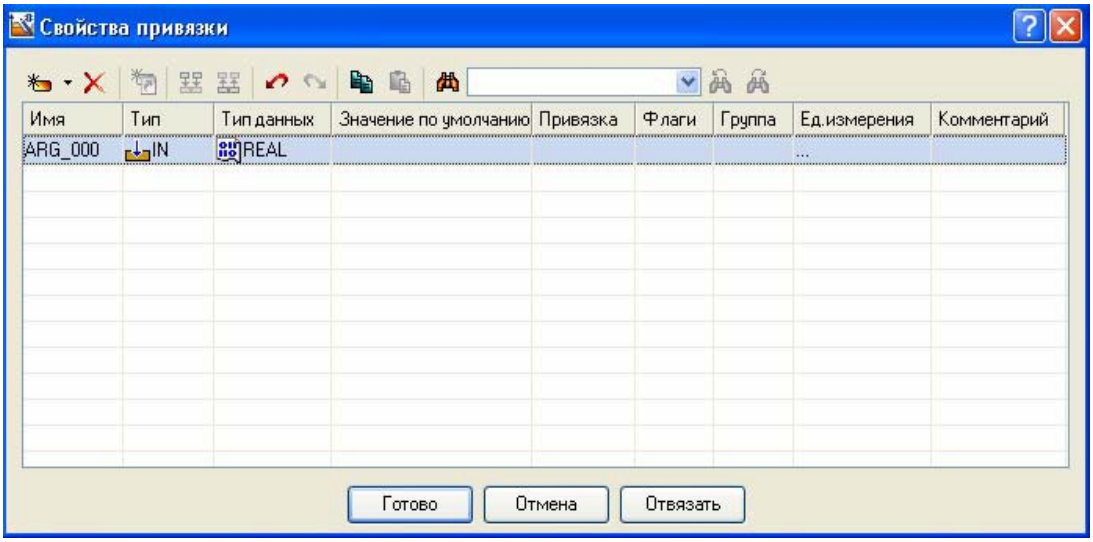


Рисунок 22 – Создание аргумента экрана

Двойным щелчком ЛК выделить имя аргумента и изменить его, введя с клавиатуры «Параметр» (завершить ввод нажатием клавиши Enter). Подтвердить связь с этим аргументом нажатием кнопки **Готово**. Закрыть окно свойств ГЭ, графический экран будет иметь следующий вид рисунок 23.

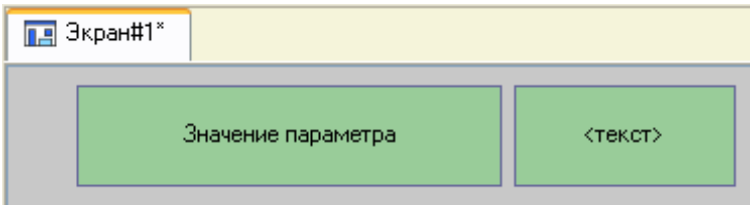


Рисунок 23 – Вид графического экрана

Создание стрелочного прибора, привязка к тому же аргументу

Применим для отображения параметра новый тип ГЭ – Стрелочный прибор рисунок 23. Выделить двойным щелчком ЛК на инструментальной панели редактора графики иконку  и выбрать в появившемся меню иконку стрелочного прибора  . Установить ГЭ Стрелочный прибор, выбрав его размер таким, чтобы все элементы графики и текста на нем было разборчивы и симметричны. Перейти в режим редактирования и открыть окно свойств ГЭ Стрелочный прибор. Щелчком ЛК на кнопке **Основная привязка** открыть окно табличного редактора аргументов. ЛК выбрать аргумент шаблона экрана Параметр. Подтвердить выбор ЛК на кнопке **Готово.** Двойным щелчком ЛК открыть свойство **Заголовок** и в строке **Текст** ввести слово «Параметр». Закрыть окно свойств.

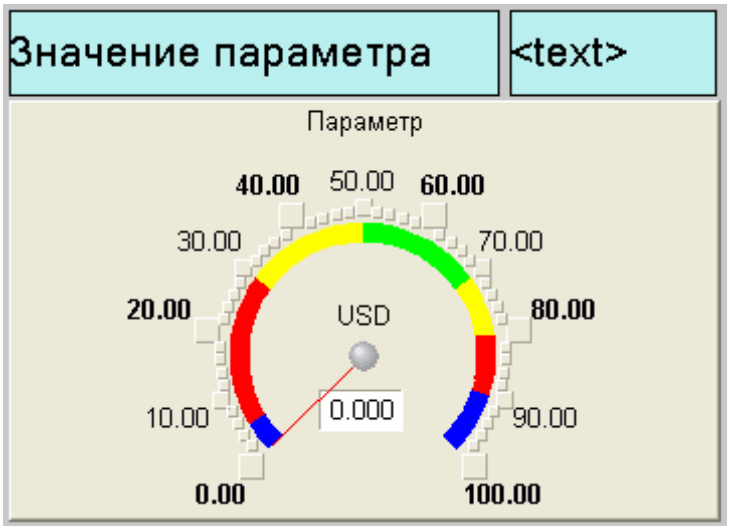


Рисунок 23 – Стрелочный прибор

Автопостроение канала

Для создания канала в узле проекта по аргументу шаблона экрана воспользуемся процедурой автопостроения. В слое **Система** открыть узел RTM_1. С помощью ПК вызвать контекстное меню и ЛК открыть свойства компонента Экран#1 рисунок 24.

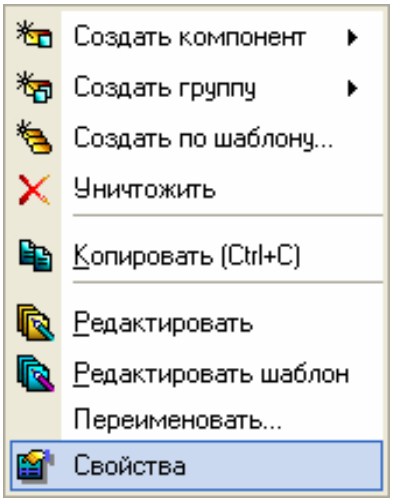


Рисунок 24 – Выбор свойств экрана

Выбрать вкладку **Аргументы**, выделить ЛК аргумент Параметр и с помощью иконки  создать канал класса **Float** типа **Input** с именем **Параметр**.

Создание генератора синуса и привязка его к каналу

Введем в состав проекта источник сигнала – внутренний генератор синусоиды, свяжем его с созданным каналом и опробуем выполненные средства отображения. Открыть слой **Источники/Приемники** и через ПК создать в нем группу **Генераторы** рисунок 25.

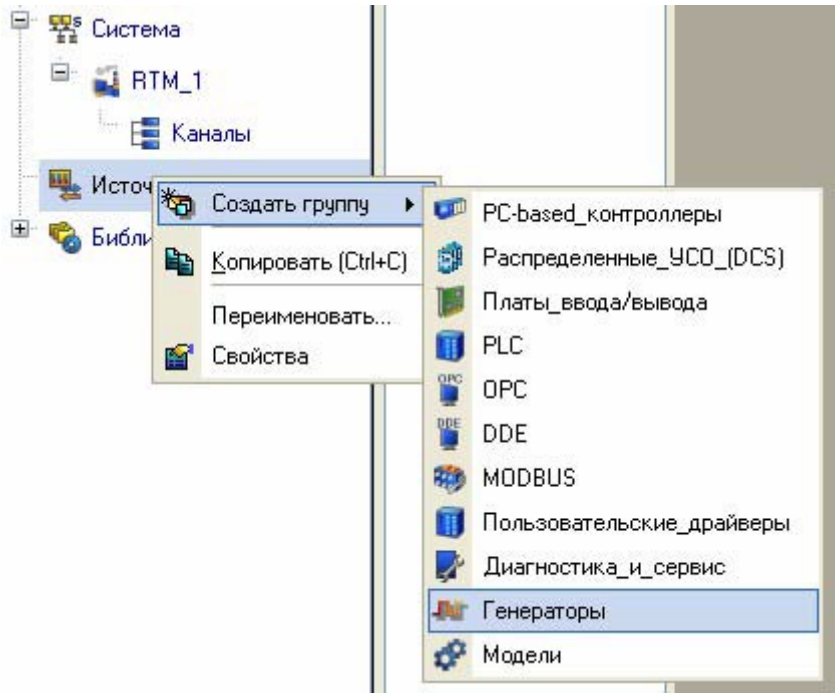


Рисунок 25 – Добавление группы Генераторы

Двойным щелчком ЛК открыть группу **Генераторы** и через ПК создать в ней компонент **Синусоида** рисунок 26.

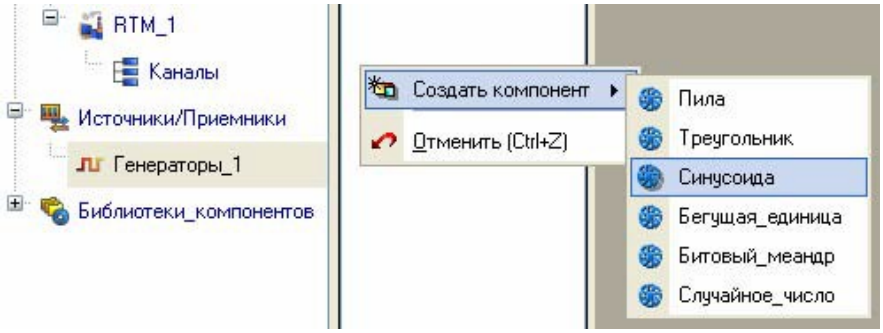


Рисунок 26 – Создание компонента Синусоида Захватить с помощью ЛК созданный источник и, не отпуская ЛК, перетащить курсор на узел RTM_1 в слое **Система**, а затем, в открывшемся окне компонентов RTM_1, на канал **Параметр** рисунок 27. Отпустить ЛК.

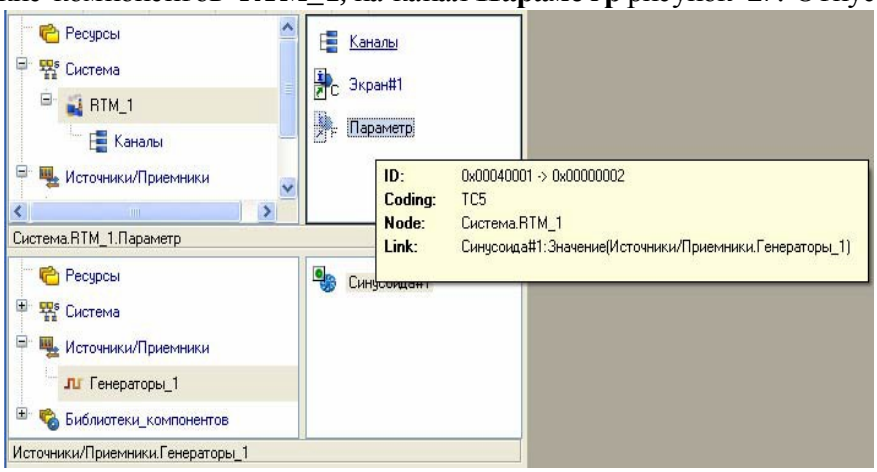


Рисунок 27 – Связь компонента Синусоида с каналом Параметр

Запуск проекта

Закрыть окно графического редактора. Сохранить проект с помощью иконки . На инструментальной панели выбрать команду и скомпилировать проект для запуска в реальном времени. ЛК выделить в слое **Система** узел RTM_1, выбрать иконку на инструментальной панели и запустить режим исполнения. В открывшемся окне ГЭ справа от надписи

«Значение параметра» должно показываться изменение синусоидального сигнала. То же значение должен отображать и стрелочный прибор рисунок 28.

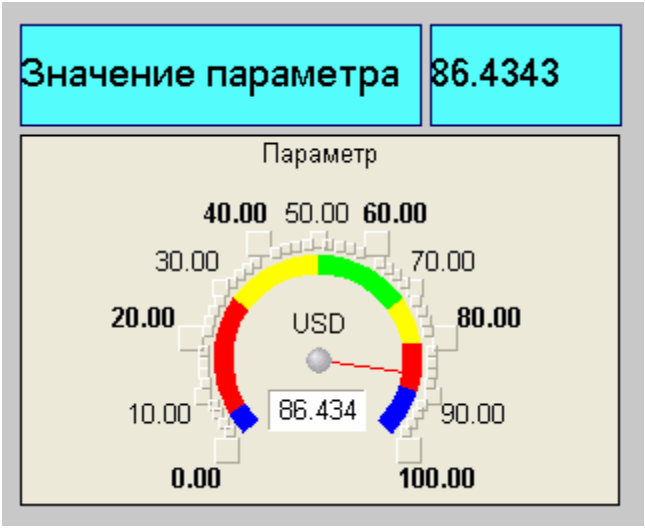


Рисунок 28 – Результат имитационного запуска проекта

Добавление функции управления

Введем в состав графического экрана средство, позволяющее реализовать ввод числовых значений с клавиатуры. Создадим новый аргумент шаблона экрана для их приема.

Вызвать графический экран на редактирование. Выбрать на инструментальной панели графического редактора иконку ГЭ Кнопка - . С помощью мыши разместить его в поле экрана под ГЭ Стрелочный прибор рисунок 29.

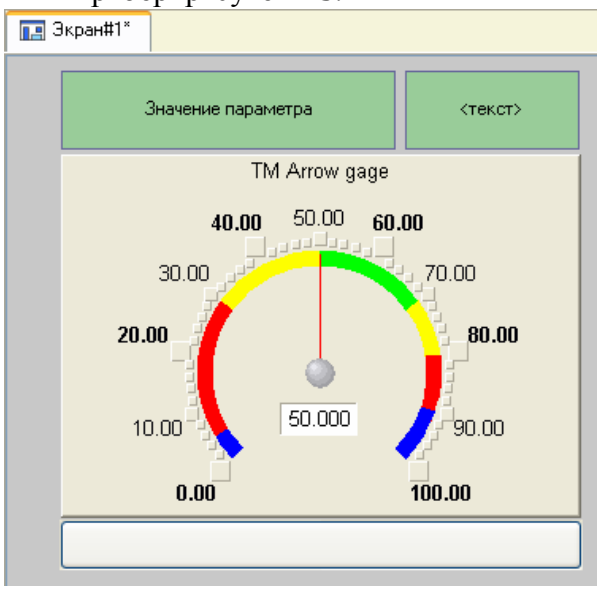


Рисунок 29 – Вид графического экрана

Перейти в режим редактирования , вызвать окно свойств ГЭ рисунок 30.

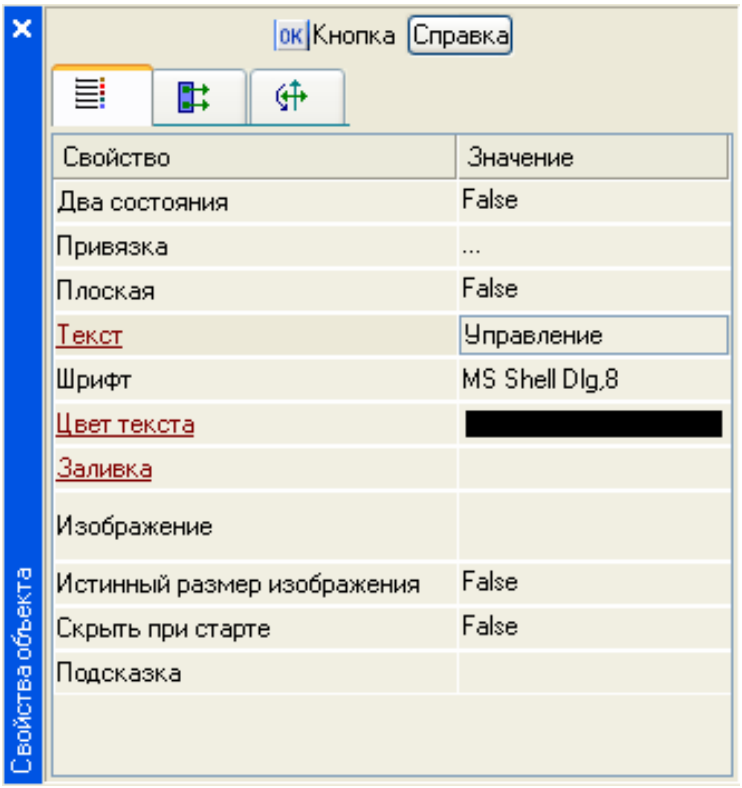


Рисунок 30 – Свойства графического элемента кнопка

В поле **Текст** ввести «Управление». Открыть бланк **События** и ПК раскрыть меню **По нажатию (pressed)**. Выбрать из списка команду **Добавить Send Value**, раскрыть меню настроек выбранной команды рисунок 31.

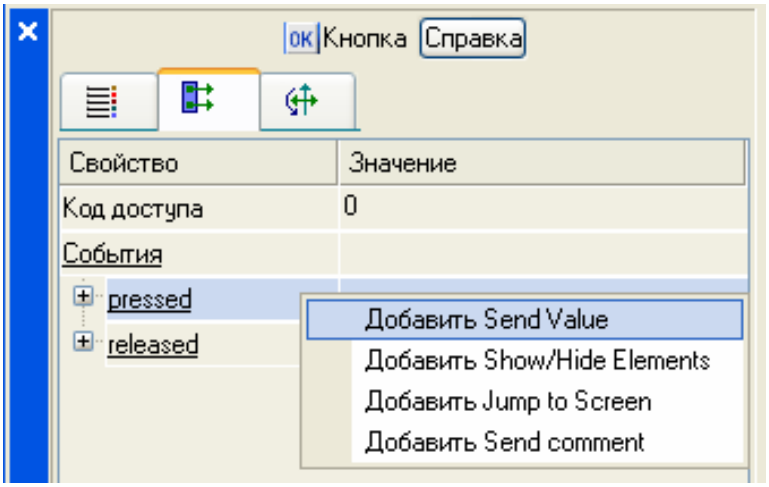


Рисунок 31 – Настройка графического элемента Кнопка

В поле **Тип передачи (Send Type)** выбрать из списка **Ввести и передать (Enter & Send)** рисунок 32.

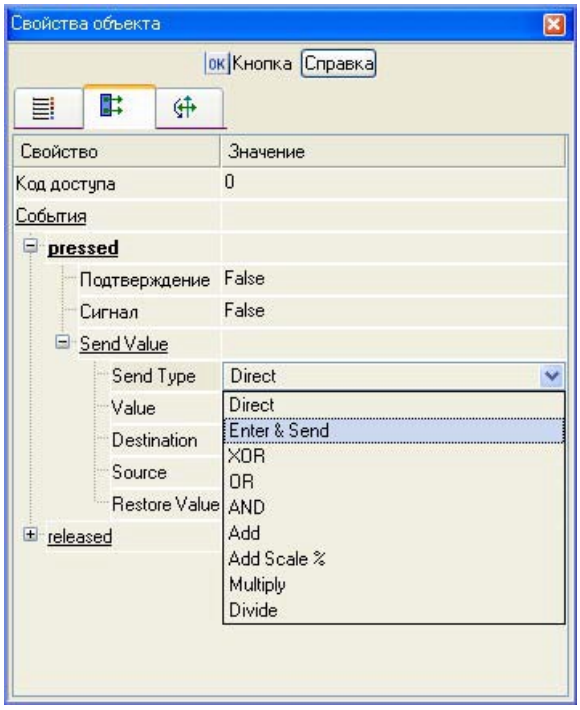


Рисунок 32 – Настройка графического элемента Кнопка

ЛК в поле **Результат** вызвать табличный редактор аргументов. Создать еще один аргумент и задать ему имя **Управление**. Изменить тип аргумента на **IN/OUT**, кнопкой Готово подтвердить привязку атрибута ГЭ к этому аргументу рисунок 33.

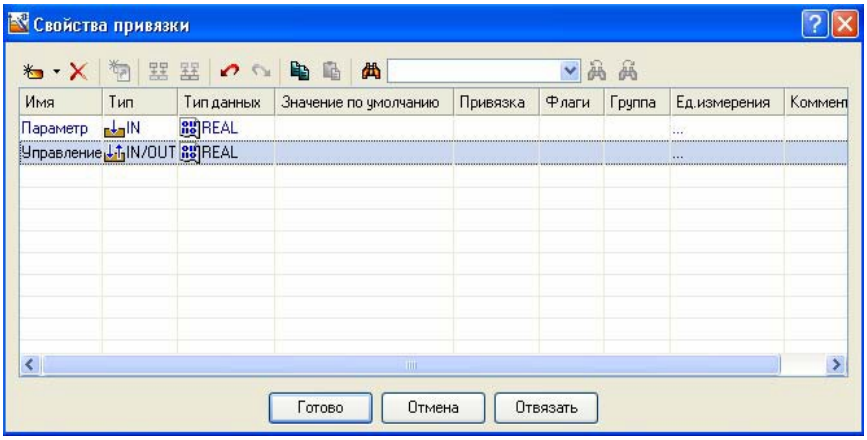


Рисунок 33 – Создание аргумента Управление

Закрыть окно свойств ГЭ с помощью щелчка ЛК по иконке  . Выделить ЛК ГЭ Текст, служащий для отображения значения Параметра рисунок 34

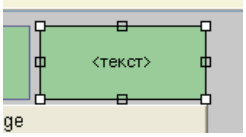


Рисунок 34 – Графический элемент текст

С помощью иконки  на панели инструментов или комбинацией клавиш **Ctrl+C** скопировать выделенный ГЭ Текст в буфер обмена. Далее с помощью иконки  или комбинацией клавиш **Ctrl+V** извлечь копию ГЭ из буфера обмена и поместить ее на графический экран. Переместить, удерживая нажатой ЛК, копию ГЭ **Текст** справа от размещенного на экране ГЭ Кнопка. Двойным

щелчком ЛК на перемещенном ГЭ Текст открыть окно его свойств рисунок 35. Двойным щелчком ЛК на строке Текст вкладки основных свойств  перейти к настройке динамизации

данного атрибута ГЭ. В правом поле строки **Привязка** щелчком ЛК открыть табличный редактор аргументов шаблона экрана. Выделить ЛК в списке аргумент **Управление** и щелчком ЛК по экранной кнопке Готово подтвердить привязку атрибута ГЭ Текст к данному аргументу шаблона экрана. Закрыть окно свойств ГЭ Текст.

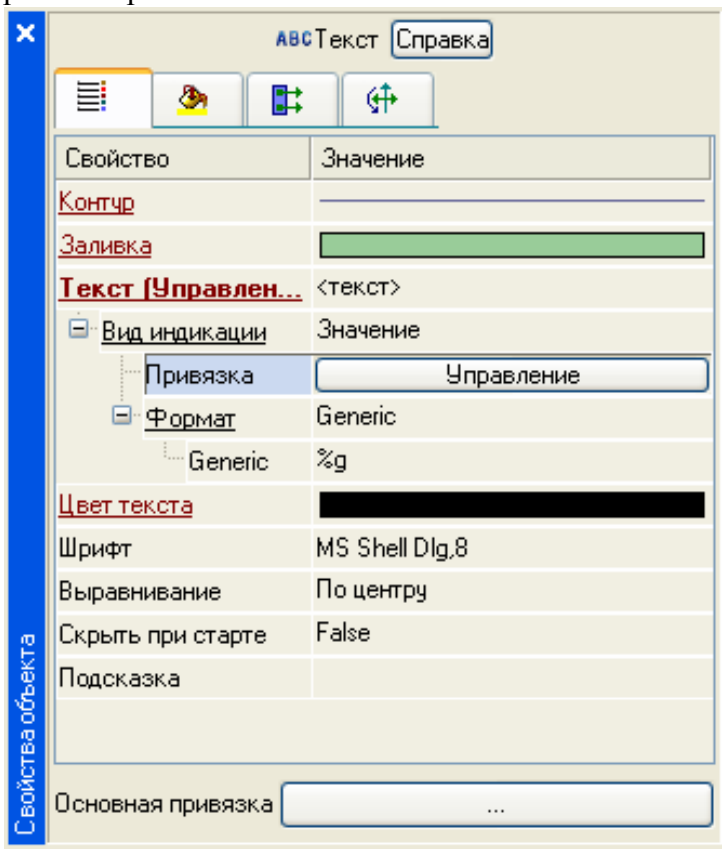


Рисунок 35 – Свойства графического элемента Текст Закрыть окно графического редактора.

Привязка аргумента экрана к каналу

Создадим по аргументу Управление шаблона экрана новый канал, отредактируем его привязку. В слое **Система** открыть узел RTM_1. С помощью ПК вызвать через контекстное меню свойства компонента Экран#1 рисунок 36.

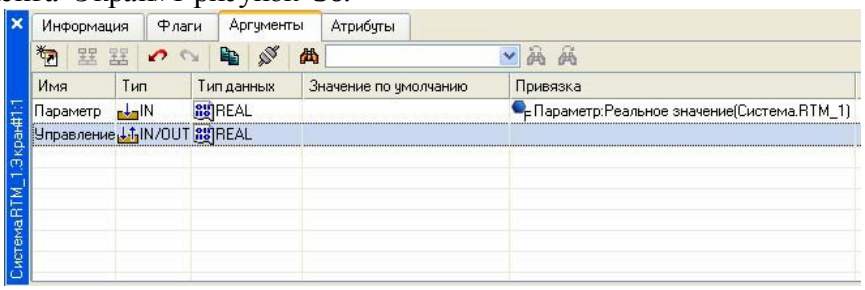


Рисунок 36 – Окно свойств экрана

Выбрать вкладку **Аргументы**, ЛК выделить аргумент **Управление** и с помощью иконки  создать новый канал. В результате, в узле RTM_1 ,будет автопостроен канал с именем **Управление** рисунок 37.

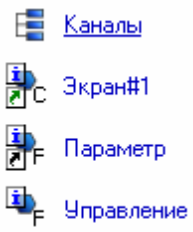


Рисунок 37 – Автопостроенный канал Управление

Двойным щелчком в поле **Привязка** аргумента **Управление** вызвать окно настройки связи, выбрать в нем атрибут **Входное значение** канала **Управление** и кнопкой **Привязка** подтвердить связь аргумента экрана **Управление** с атрибутом **Входное значение** канала **Управление** рисунок 38.

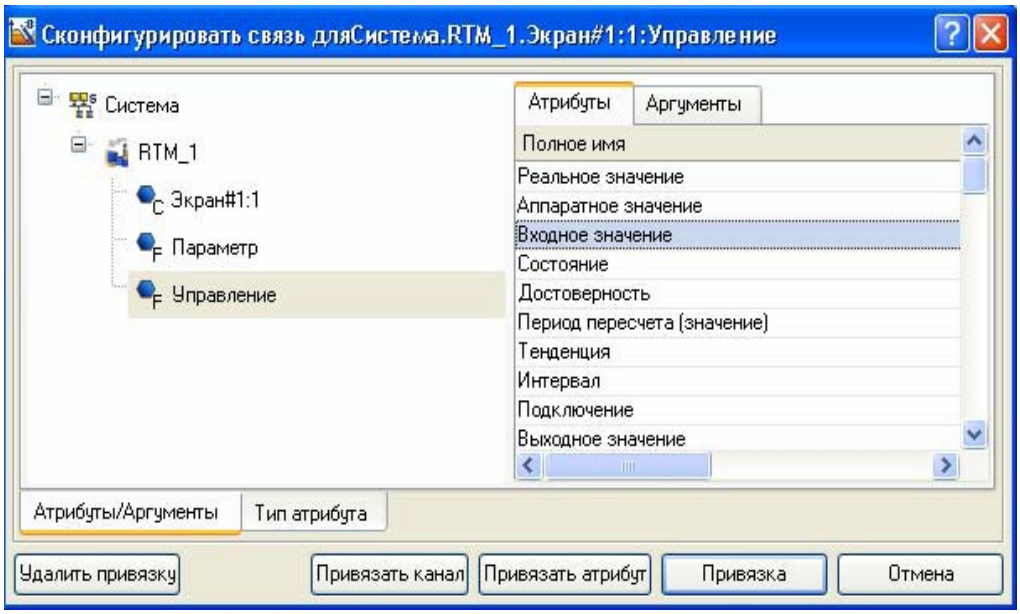


Рисунок 38 – Конфигурация системы Закрыть окно

свойств компонента Экран#1.

Размещение ГЭ Тренд

Дополним созданный экран новым ГЭ для совместного просмотра изменений значений каналов узла во времени и отслеживании предыстории – трендом.

В правой части графического экрана разместим ГЭ Тренд для вывода значений **Параметр** и **Управление**. Основные свойства ГЭ оставим заданными по умолчанию. Перейдем во вкладку и, выделив ЛК строку **Кривые**, с помощью ПК создадим две новые кривые. Настроим их привязки к аргументам, толщину и цвет линий рисунок 39.

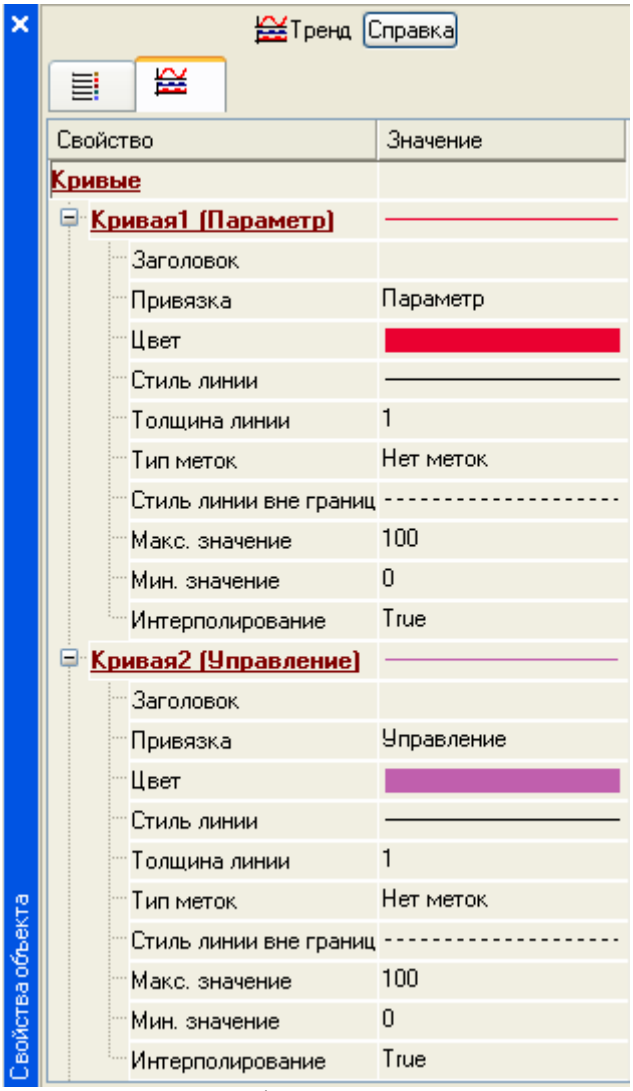


Рисунок 39 – Свойства графического элемента Тренд ГЭ примет вид рисунок 40.

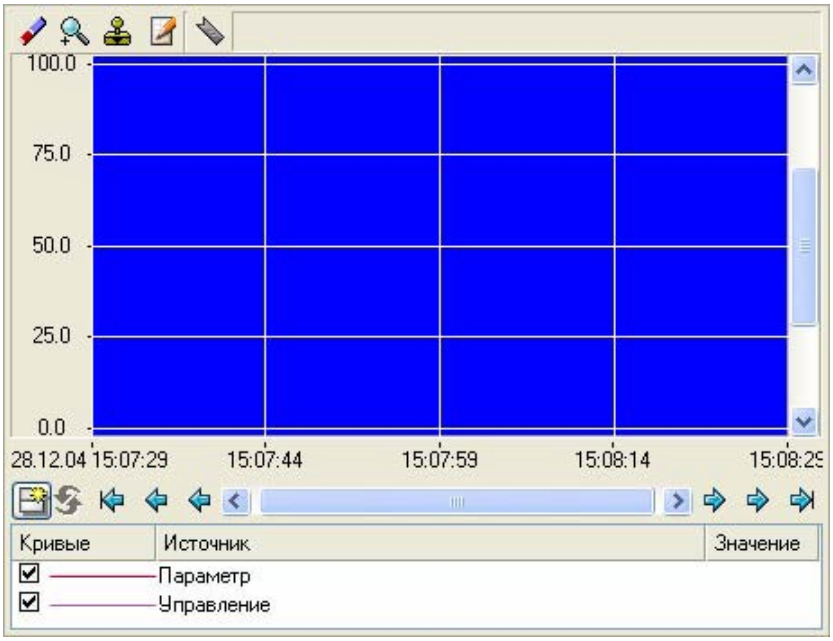


Рисунок 40 – Вид графического элемента Тренд

Запуск проекта

Сохранить проект с помощью иконки . На инструментальной панели выбрать команду и скомпилировать тем самым проект для запуска в реальном времени. Выбрать иконку на инструментальной панели и запустить режим исполнения. С помощью кнопки «Управление» ввести величину «управляющего воздействия» и наблюдать результат на соседнем индикаторе и тренде.

Простейшая обработка данных

С помощью создания нового компонента проекта – шаблона программы свяжем операцией сложения два имеющихся канала. Будем суммировать реальные значения каналов Параметр и Управление, а результат помещать во вновь созданный аргумент экрана Сумма (с отображением на ГЭ Текст и Тренд) без создания дополнительного канала в узле проекта.

Скопировать два первых ГЭ – «Значение параметра» и «text» и разместить их ниже ГЭ Кнопка рисунок 41.

Рисунок 41 – Вид графического экрана Изменить статический текст первого ГЭ на «Сумма :» рисунок 42.

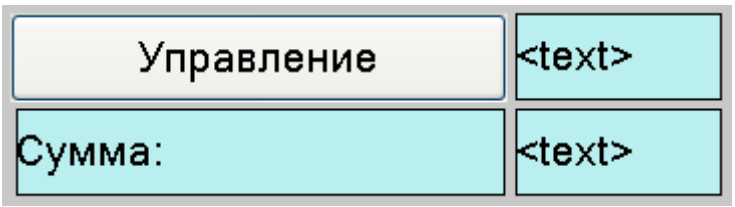


Рисунок 42 – Вид графического экрана

А динамику второго ГЭ привязать к третьему аргументу экрана типа **IN** с именем **Сумма**, который создать в процессе привязки рисунок 43.

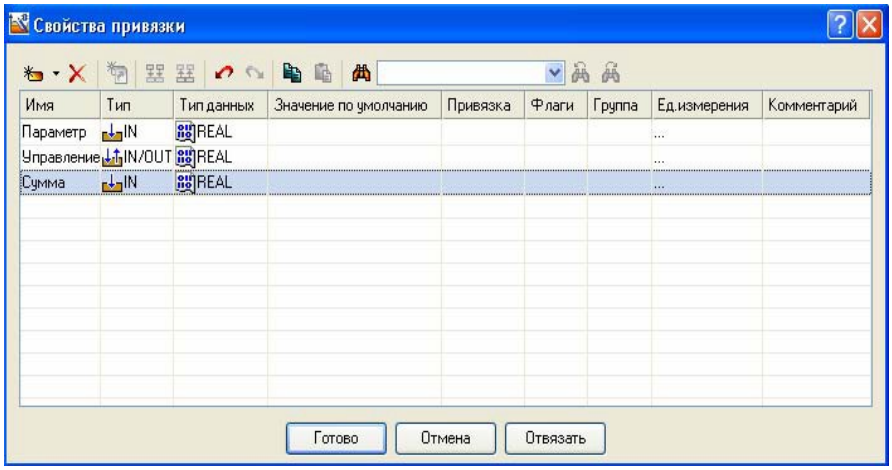


Рисунок 43 – Свойства привязки

Добавить еще одну кривую на тренд с привязкой к аргументу **Сумма** рисунок 44.

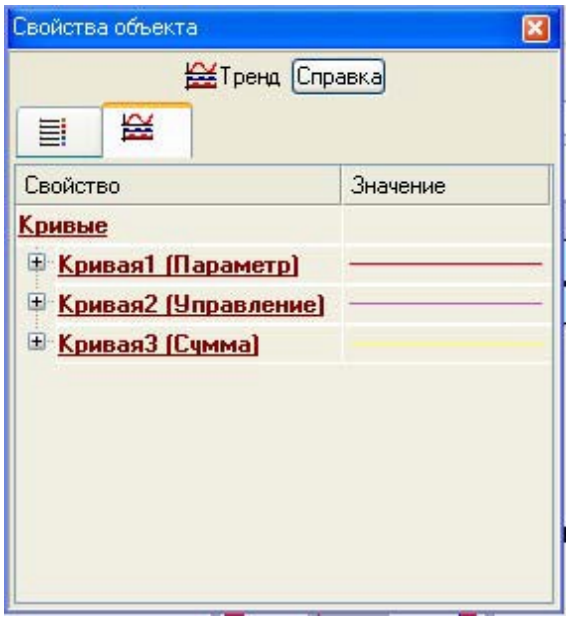


Рисунок 44 – Свойства графического элемента Тренд

Создание программы на языке Техно ST

Создадим программу, в которой сумма двух аргументов, связанных с атрибутами **Реальное значение** каналов Параметр и Управление, будет помещается в третий аргумент с именем Сумма. В дальнейшем, воспользуемся возможностью связывания аргументов шаблонов для вывода на экран результата работы программы без создания дополнительного канала.

Двойным щелчком ЛК открыть узел RTM_1 и создать в нем компонент **Программа** рисунок 45.

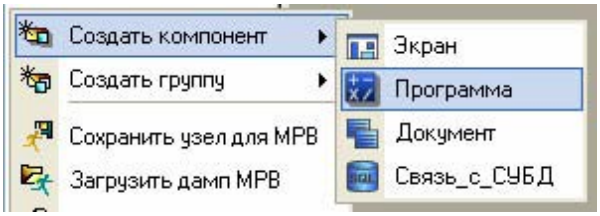


Рисунок 45 – Создание компонента программа

выделить компонент Программа#1 и ПК вызвать контекстное меню, выбрав в котором ЛК пункт **Редактировать шаблон**, перейти в режим редактирования программы рисунок 46.

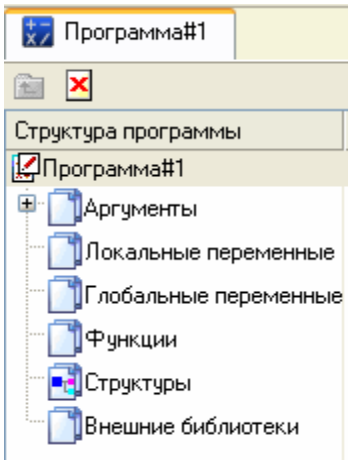
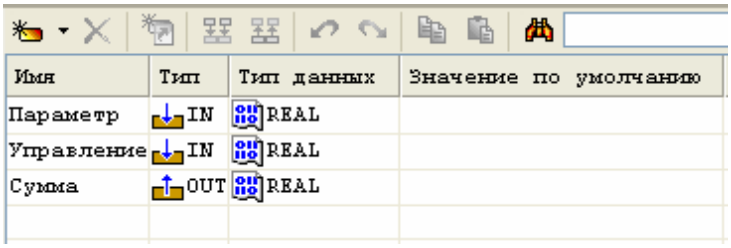


Рисунок 46 – Режим редактирования программы

Выделением ЛК в дереве шаблона Программа#1 строки **Аргументы** вызвать табличный редактор аргументов. Иконкой  создать в редакторе аргументов три аргумента с именами Параметр, Управление и Сумма. При этом первые 2 аргумента должны быть типа **IN**, а третий – **OUT** рисунок 47.



Имя	Тип	Тип данных	Значение по умолчанию
Параметр	IN	REAL	
Управление	IN	REAL	
Сумма	OUT	REAL	

Рисунок 47 – Создание каналов программы

Выделить в дереве шаблона строку Программа#1 и в открывшемся диалоге **Выбор языка** выбрать язык ST рисунок 48.

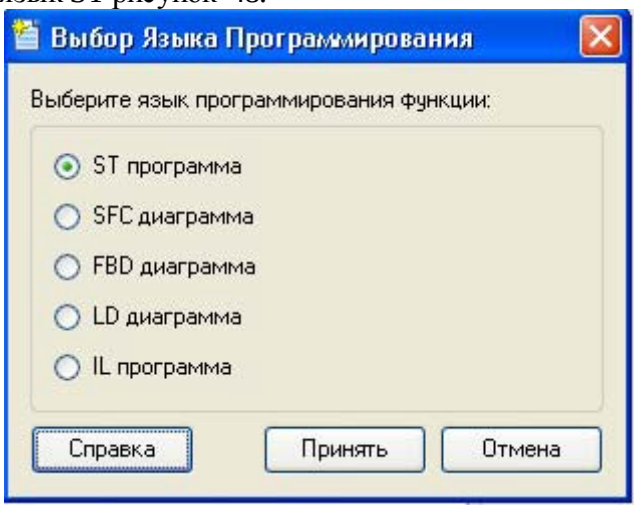


Рисунок 48 – Выбор языка программирования

По нажатию экранной кнопки **Принять** в открывшемся окне редактора программ с объявленными переменными набрать следующую строку рисунок 49.

```
PROGRAM
    VAR_INPUT Параметр : REAL; END_VAR
    VAR_INPUT Управление : REAL; END_VAR
    VAR_OUTPUT Сумма : REAL; END_VAR

Сумма=Параметр+Управление;

END_PROGRAM
```

Рисунок 49 – Вид программы на языке ST

С помощью иконки  на инструментальной панели редактора или «горячей клавишей» F7 скомпилировать программу и убедиться в успешной компиляции в окне **Выход (Output)**, вызываемого из инструментальной панели с помощью иконки  рисунок 50.

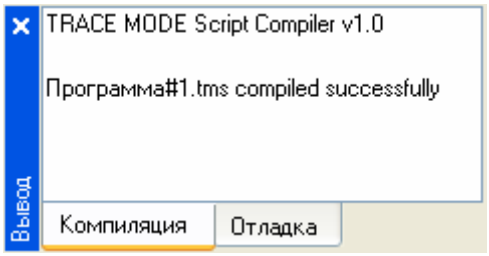


Рисунок 50 – Результат удачной компиляции программы

Привязка аргументов программы

Выполним привязку аргументов программы к атрибутам каналов. Вызвать свойства компонента Программа#1 через контекстное меню. Выбрать вкладку **Аргументы**. Двойным нажатием в поле Привязка привязать аргументы программы к атрибутам каналов – аргумент **Параметр** к реальному значению канала Параметр, аргумент Управление к реальному значению канала **Управление** рисунок 51.

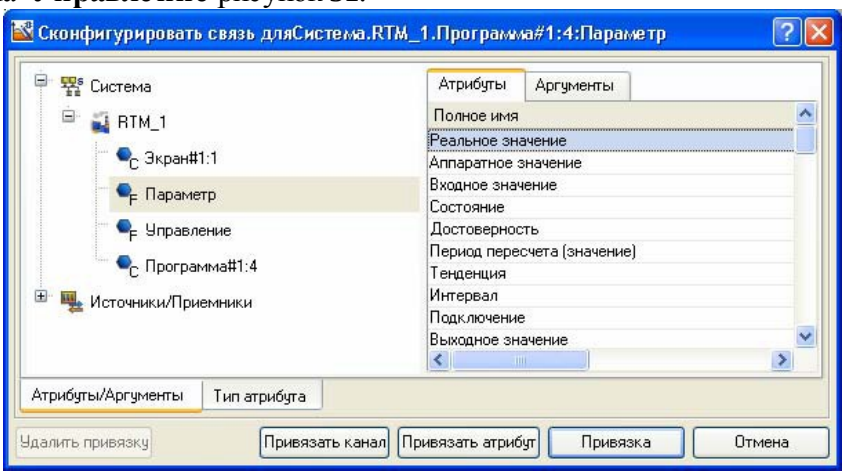


Рисунок 52 – Привязка аргументов программы к атрибутам каналов Двойным щелчком в поле **Привязка** аргумента программы **Сумма** вызвать окно настройки связи, выбрать в левом окне канал класса **Вызов** Экран#1, а в правом окне выбрать вкладку **Аргументы** и указать в ней аргумент **Сумма** и кнопкой **Привязка** подтвердить связь рисунок 53.

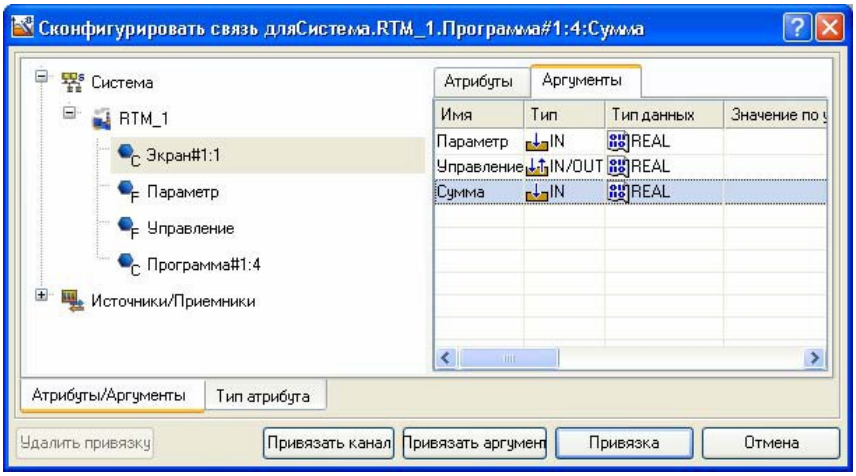


Рисунок 53 – Привязка аргументов программы к атрибутам каналов

В результате, будем иметь рисунок 54.

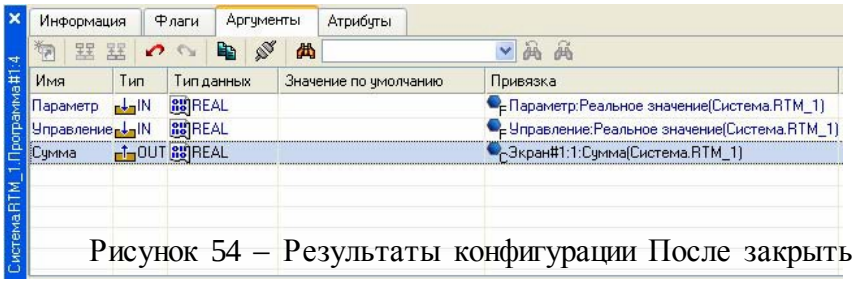


Рисунок 54 – Результаты конфигурации После закрыть окно свойств компонента Программа#1.

Запуск проекта

Сохранить проект с помощью иконки  . На инструментальной панели выбрать команду  и скомпилировать тем самым проект для запуска в реальном времени. Выбрать иконку  на инструментальной панели и запустить режим исполнения. С помощью кнопки «Управление» ввести «управляющее воздействие» и наблюдать соответствующее изменение сигнала «Управление» и смещение сигнала «Сумма» рисунок 55.

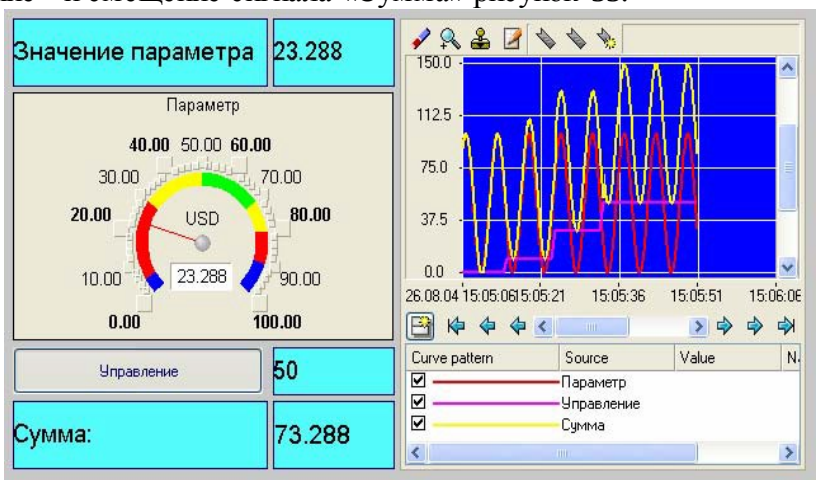


Рисунок 55 – Результат имитационного запуска проекта

Связь по протоколу DDE с приложением MS Windows на примере Excel MPB как DDE – сервер

Организуем запрос реальных значений каналов узла разработанного проекта приложением MS Windows в качестве которого выберем книгу MS Excel.

Запустить MS Excel. Записать в двух ячейках первого столбца запросы на получение данных:

=RTM0|GET!Парамет р

=RTM0|GET!Управление

где 0 – индивидуальный номер узла в проекте;

Запустить на исполнение узел АРМ RTM_1. В меню таблицы MS Excel **Правка** выбрать команду **Связи**, выделить все три параметра и нажать кнопку **Обновить**, после чего закрыть окно кнопкой **ОК**. Убедиться, что значения в ячейках книги Excel изменяются вместе с соответствующими реальными значениями каналов узла (значения канала **Параметр** меняется постоянно, а канала **Управление** – после введения нового значения с помощью ГЭ Кнопка) рисунок 56.

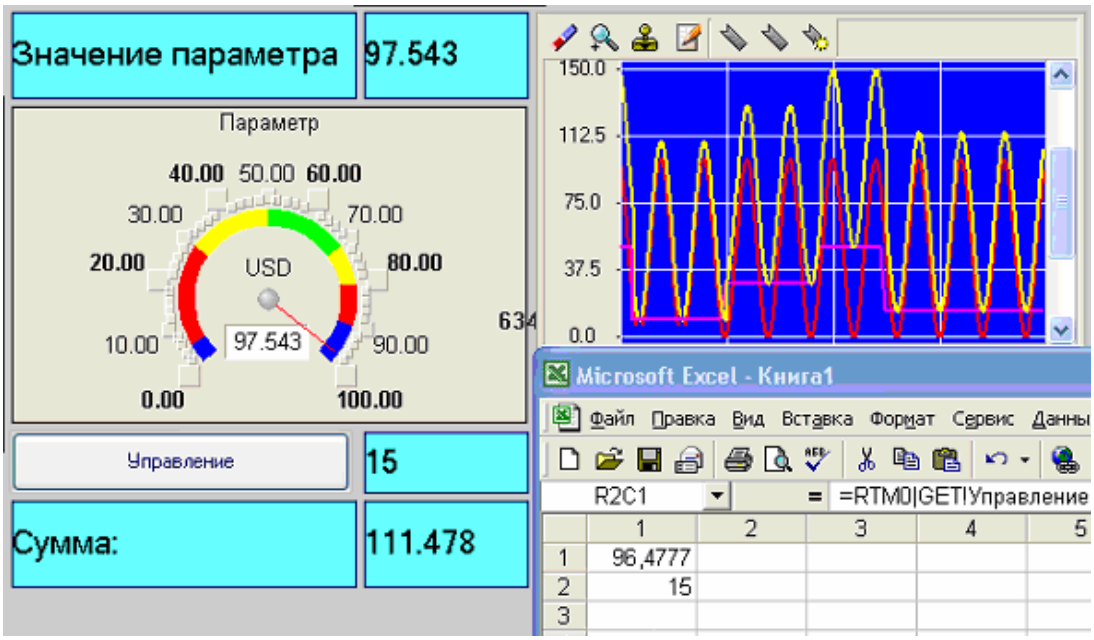


Рисунок 56 – Результат имитационного запуска проекта

МРВ как DDE-клиент

В том случае, когда требуется получать данные от внешнего приложения по протоколу DDE, МРВ должен выступать в роли DDE-клиента. Например, если необходимо вводить во вновь создаваемый канал (в его атрибут **Входное значение**) Из_таблицы узла RTM_1 данные из ячейки R3C3 книги MS Excel, надо в слое **Источники/Приемники** создать новую группу DDE, а в ней – компонент DDE#1 и отредактировать его следующим образом рисунок 57.

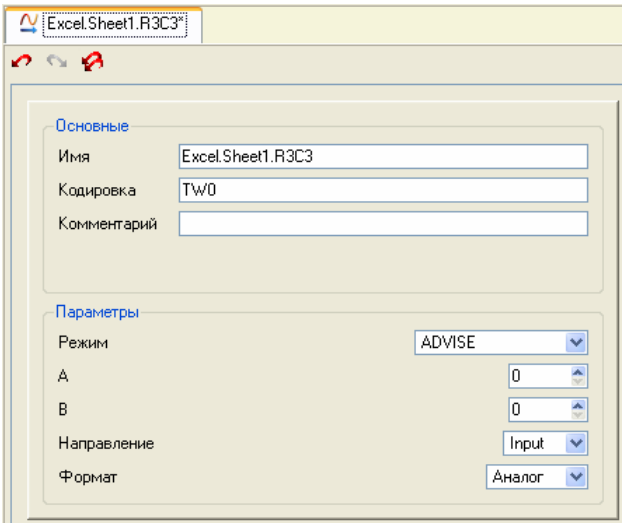


Рисунок 57 – Создание группы DDE#1

где в атрибуте **Имя**: Excel – имя приложения; Sheet1 – имя листа книги Excel; R3C3 – адрес ячейки.

ADVISE – режим посылки клиенту значения при каждом его изменении. После необходимо создать канал класса **Float** типа **Input** с именем

Из_таблицы и привязать к нему с помощью механизма **drag-and-drop** источник Excel.Sheet1.R3C3. После процедур сохранения проекта и подготовки его к запуску в реальном времени, запустим Excel, а затем узел АРМ RTM_1. Вводя в ячейку R3C3 произвольные значения, их можно наблюдать в атрибутах канала Из_таблицы с помощью окна просмотра компонентов, открываемое через основное меню отладчика рисунок 58.

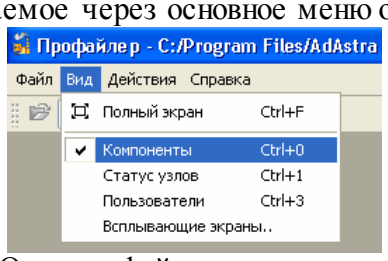


Рисунок 58 – Окно профайлера

Таким образом, в результате будем иметь следующее рисунок 59

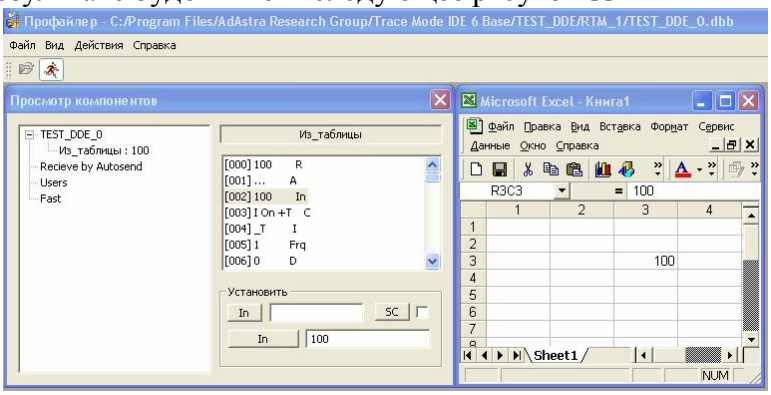


Рисунок 59 – Результат имитационного запуска проекта

Для составления отчета необходимо в меню **Файл** выполнить команду **Документировать проект** полученный *.html файл необходимо распечатать.

Контрольные вопросы

1. Системы мониторинга и управления технологическими процессами
2. Этапы создания систем управления на базае SCADA–систем
3. Функциональные характеристики SCADA-систем
4. Функциональные возможности
5. Программно-аппаратные платформы SCADA-систем

6. Средства сетевой поддержки
7. Встроенные командные языки
8. Поддерживаемые базы данных
9. Графические возможности Тренды и архивы в SCADA-системах

Лабораторная работа №2 Реализация логических функций при помощи scada–системы TRACE MODE

Цель работы: освоить методику программирования логических функций при помощи SCADA–системы TRACE MODE на языке Техно FBD.

Порядок выполнения лабораторной работы

Разработка любого проекта автоматизации всегда начинается с запуска Интегрированной среды разработки (ИСР). Для ее запуска необходимо выполнить команду TRACE MODE IDE 6 (base) из группы установки инструментальной системы в меню Программы WINDOWS или двойным щелчком ЛК мыши по иконке  рабочего стола Windows рисунок 60;

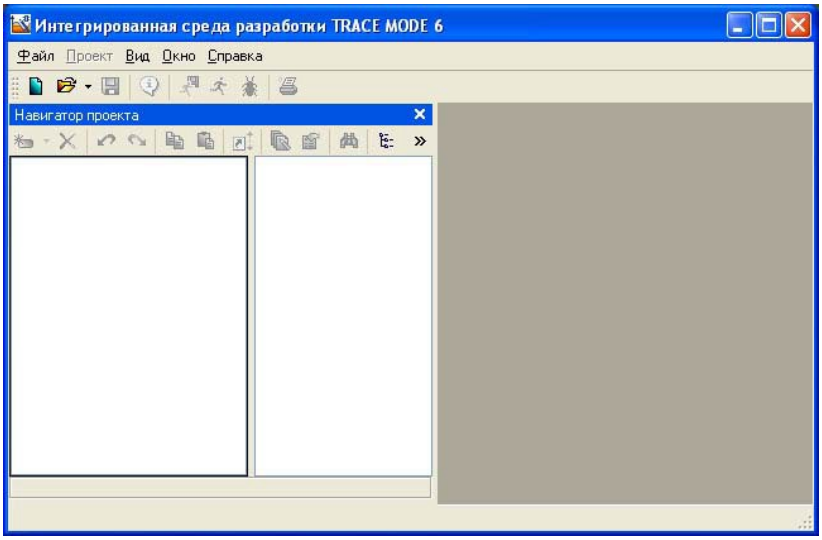


Рисунок 60– Интегрированная среда разработки

После запуска ИСР в меню **Файл** выбрать команду **Настройки ИС…**В появившемся окне выбрать **Уровень сложности** и настроить как показано на рисунке 61, а затем выбрать **Отладка** и настроить как на рисунке 62.

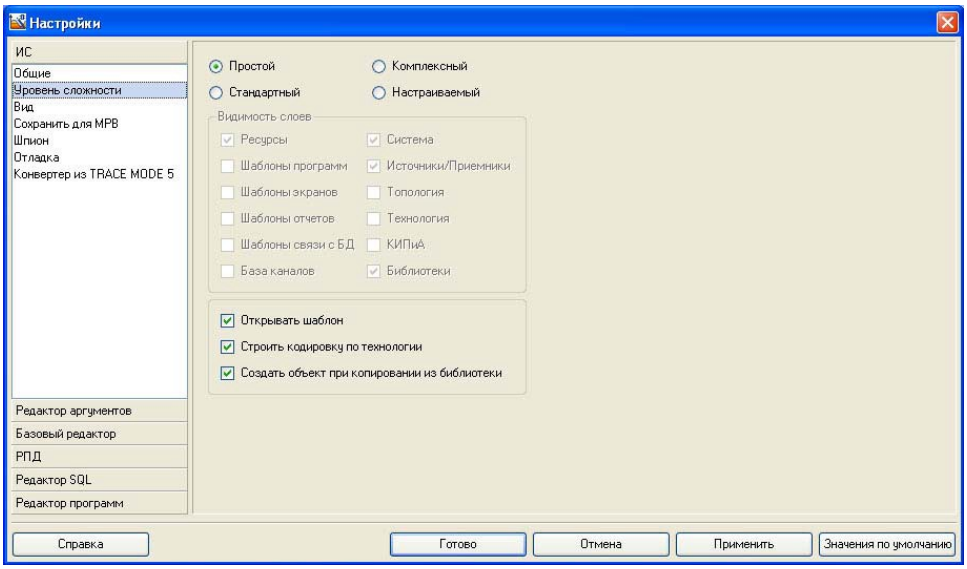


Рисунок 61 – Настройки ИСР

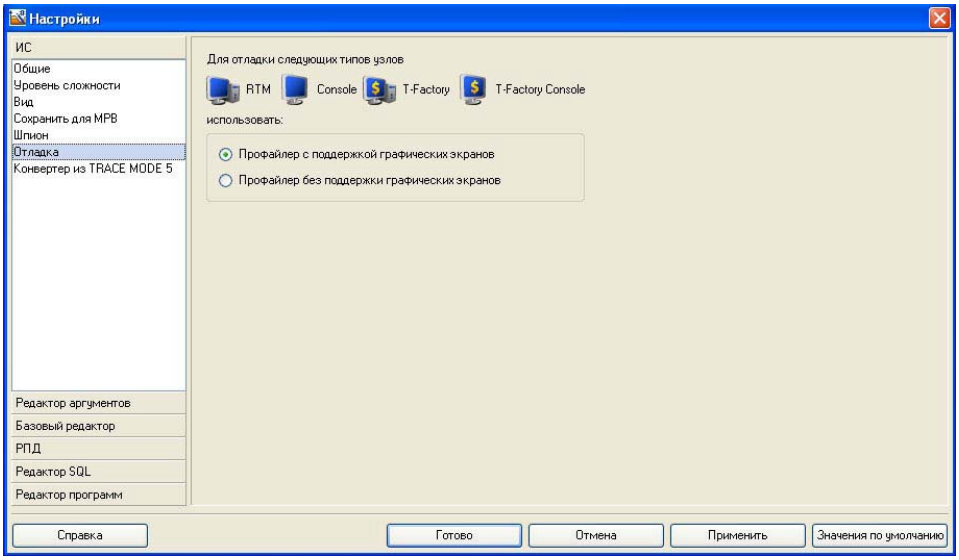


Рисунок 62 – Настройки ИСР

После проведенных настроек ИСР нажать кнопку Готово.

С помощью иконки  инструментальной панели создадим новый проект при этом в открывшемся на экране диалоге рисунок 63.

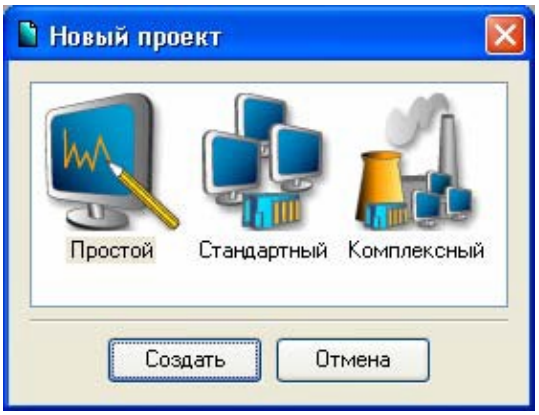


Рисунок 63 – Выбор типа проекта

Выберем **Простой** стиль разработки. После нажатия левой клавиши мыши (ЛК) на экранной кнопке **Создать**, в левом окне Навигатора проекта появится дерево проекта с созданным узлом **АРМ RTM_1**. Откроем узел **RTM_1** двойным щелчком ЛК, в правом окне **Навигатора проекта** отобразится содержимое узла – пустая группа **Каналы** и один канал класса «Вызов» **Экран#1**, предназначенный для отображения на узле **АРМ** графического экрана рисунок 64.

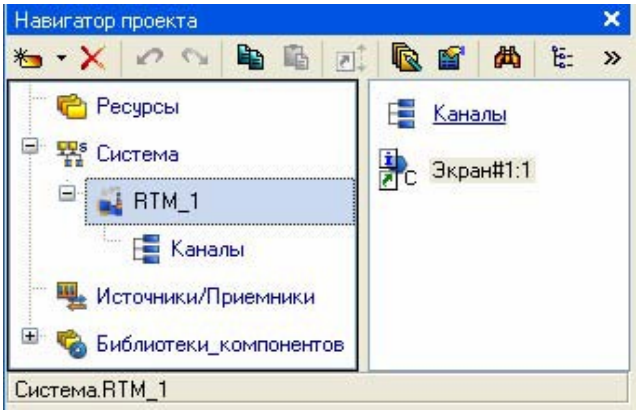


Рисунок 64 – Навигатор проекта

Создание графического экрана

Двойным щелчком ЛК на компоненте **Экран#1** открыть окно графического редактора.

Подготовим на экране вывод динамического текста для отображения численного значения входных переменных X1, X2, X3, X4 и выходного значения Y путем указания динамизации атрибута графического элемента (ГЭ). Определим назначение аргумента шаблона экрана.

Создадим и разместим четыре ГЭ 

как показано на рисунке 65.

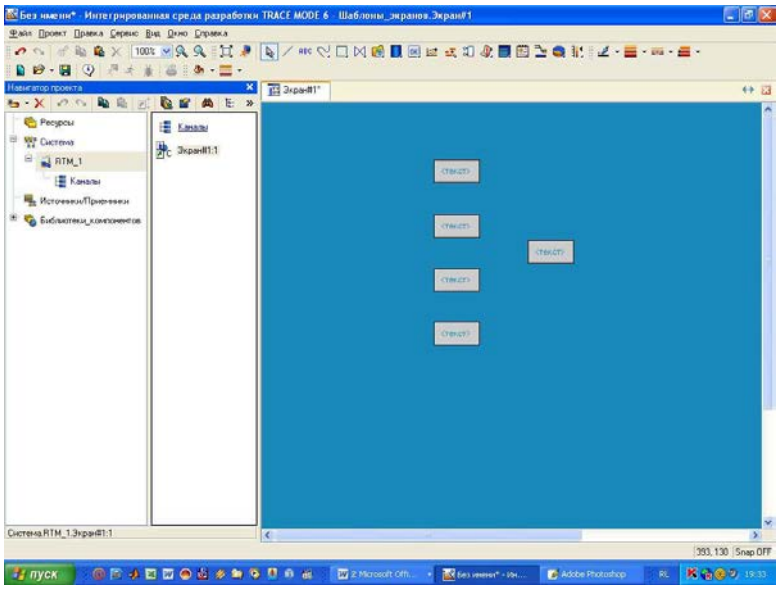


Рисунок 65 – Создание графических элементов

Двойным щелком ЛК на строке Текст вызвать меню **Вид индикации** рисунок 66;

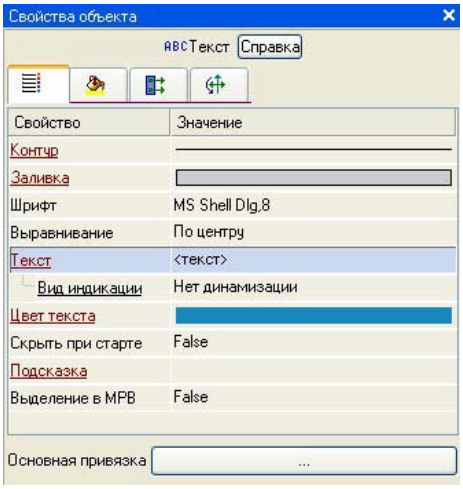


Рисунок 66 – Настройка индикации

В правом поле строки нажать ЛК и вызвать список доступных типов. Выбрать тип **Значение** рисунок 67.

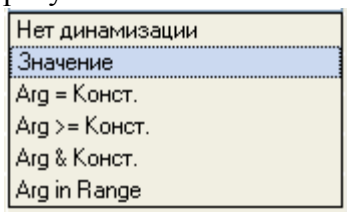


Рисунок 67 – Выбор типа динамизации

В открывшемся меню настройки параметров динамизации рисунок 68.

Рисунок 68 – Окно привязки Выбрать свойство **Привязка.**

В открывшемся окне **Свойство привязки**, нажав кнопку  на его панели инструментов, создать пять аргументов экрана 4 входных и один выходной рисунок 69.

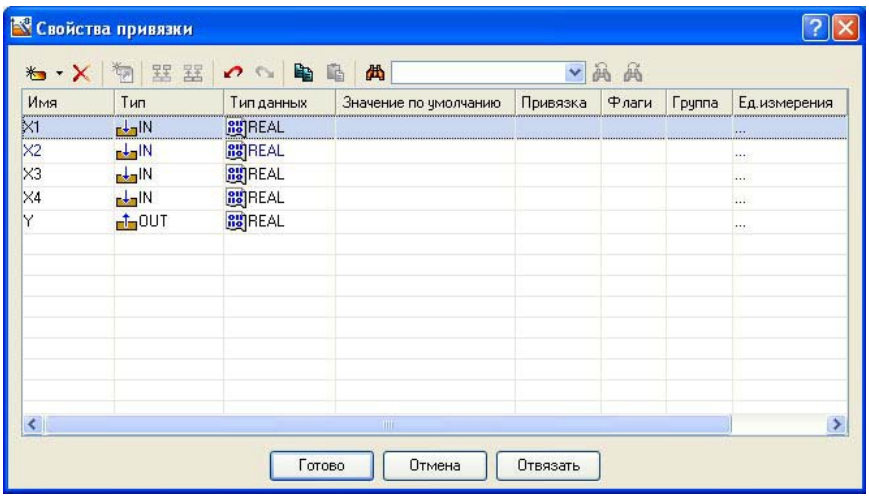


Рисунок 69 – Создание аргументов экрана

Двойным щелчком ЛК выделить имя аргумента и изменить его, введя с клавиатуры «X1»

(завершить ввод нажатием клавиши Enter). Подтвердить связь с этим аргументом нажатием кнопки Готово; Аналогичным образом настроить все входные и выходные аргументы.
Введем в состав графического экрана средство, позволяющее реализовать ввод числовых значений с клавиатуры. Создадим новый аргумент шаблона экрана для их приема.

Выбрать на инструментальной панели графического редактора иконку ГЭ Кнопка -  . С помощью мыши разместить его в поле экрана как показано на рисунке 70.

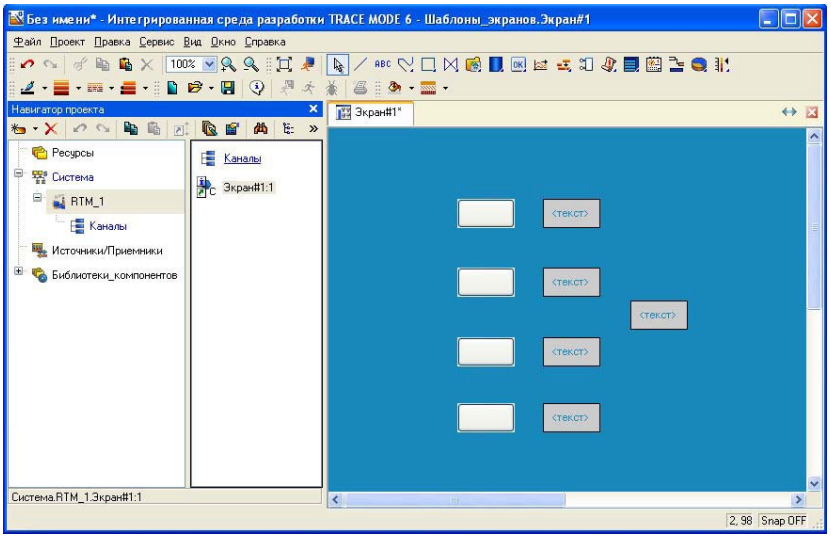


Рисунок 70 – Графический экран

Перейти в режим редактирования  первой кнопки, вызвать окно свойств ГЭ  рисунок 71.

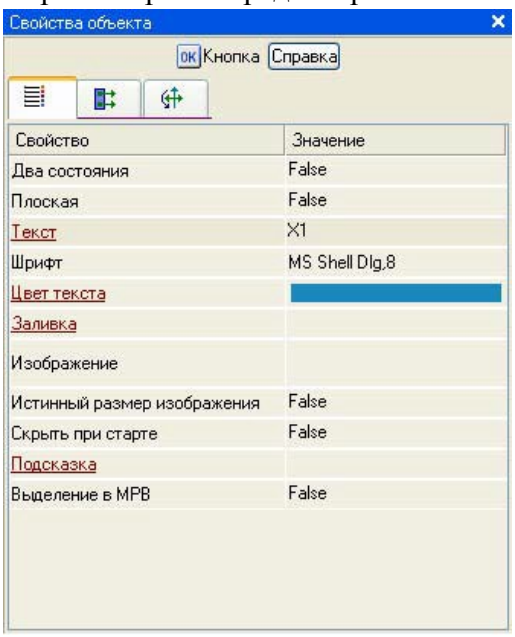


Рисунок 71 – Свойства графического элемента

В поле **Текст** ввести «X1». Открыть бланк **События**  и ПК раскрыть меню **По нажатию (pressed)**. Выбрать из списка команду **Добавить Передать значение**, раскрыть меню настроек выбранной команды рисунок 72.

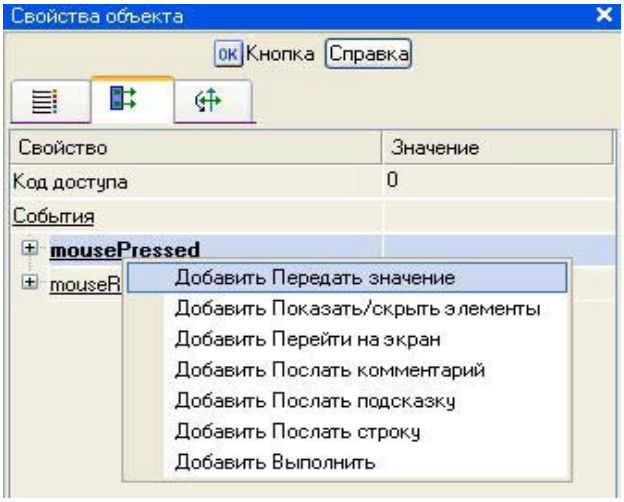


Рисунок 72 – Настройка типа передачи

В поле **Тип передачи (Send Type)** выбрать из списка **Ввести и передать (Enter & Send)** рисунок 73.

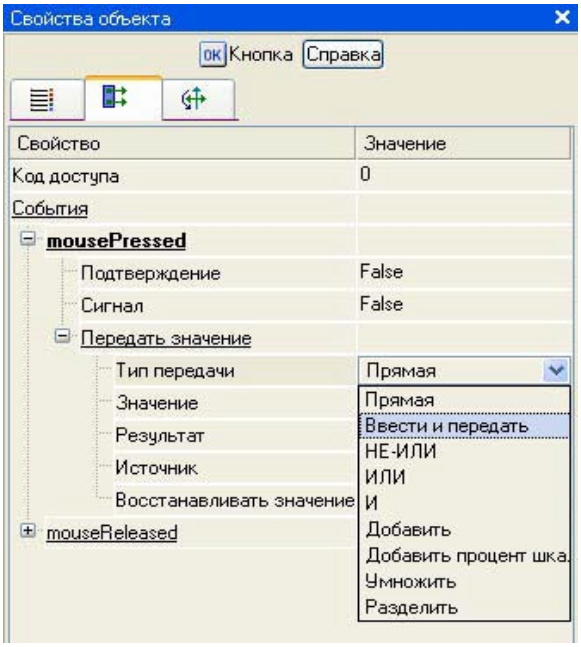


Рисунок 73 – Настройка типа передачи

ЛК в поле **Результат** вызвать табличный редактор аргументов и произвести привязку к аргументу X1. После привязки окно Свойства объекта выглядит следующим образом рисунок 74.

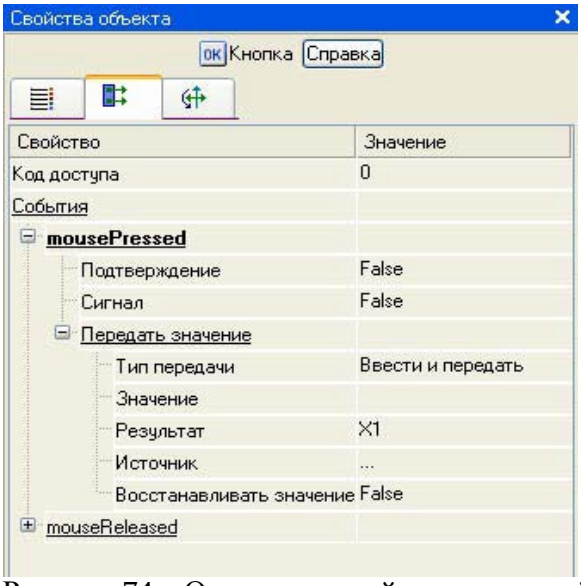


Рисунок 74 – Окончательный вид окна свойств графического элемента Аналогичным образом настроить остальные кнопки x2, x3, x4.

Привязка аргумента экрана к каналу

Создадим по аргументам X1, X2, X3, X4 и выходу Y шаблона экрана новые каналы и отредактируем их привязку. В слое **Система** открыть узел **RTM_1**. С помощью ПК вызвать через контекстное меню свойства компонента **Экран#1** рисунок 75.

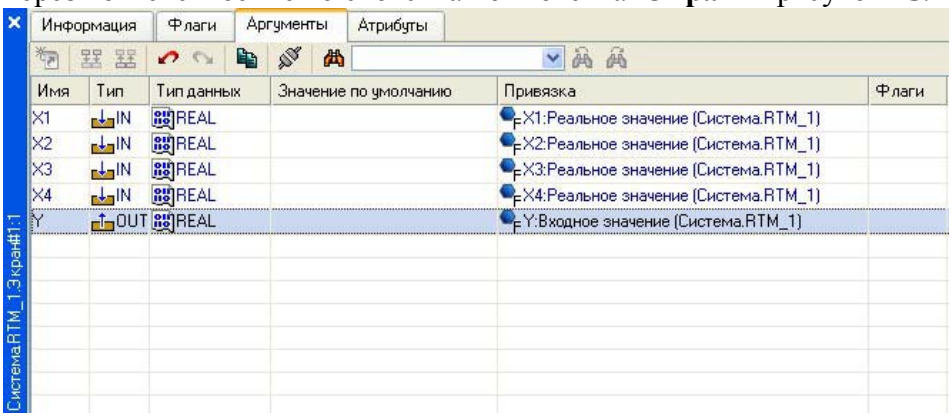


Рисунок 75 – Окно свойств экрана

Выбрать вкладку Аргументы, ЛК выделить аргументы X1, X2, X3, X4 и выход Y и с помощью иконки  создать новый канал. В результате, в узле **RTM_1** ,будут автопостроены следующие каналы рисунок 75.

Рисунок 75 – Автопостроены каналы

3.2.1 Создание программы на языке ТехноFBD

Создадим программу, которая будет реализовывать заданную логическую функцию. Двойным щелчком ЛК открыть узел **RTM_1** и создать в нем компонент **Программа** рисунок 76.

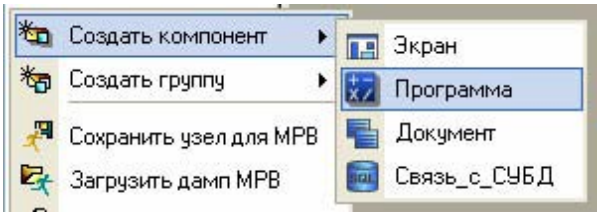


Рисунок 76 – Создание компонента Программа

Выделить компонент Программа#1 и ПК вызвать контекстное меню, выбрав в котором ЛК пункт **Редактировать шаблон**, перейти в режим редактирования программы рисунок 77.

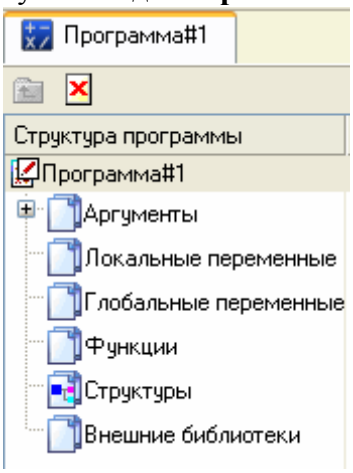


Рисунок 77 – Редактирование аргументов Программы

Выделением ЛК в дереве шаблона **Программа#1** строки Аргументы вызвать табличный редактор аргументов иконкой  создать в редакторе аргументов четыре аргумента X1, X2, X3, X4 и выход Y. При этом первые 4 аргумента должны быть типа **IN**, а последний – **OUT** рисунок 78.

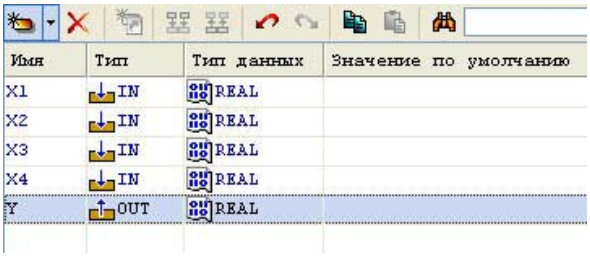


Рисунок 78 – Аргументы программы

Выделить в дереве шаблона строку **Программа#1** и в открывшемся диалоге Выбор языка выбрать язык **FBD** рисунок 79.

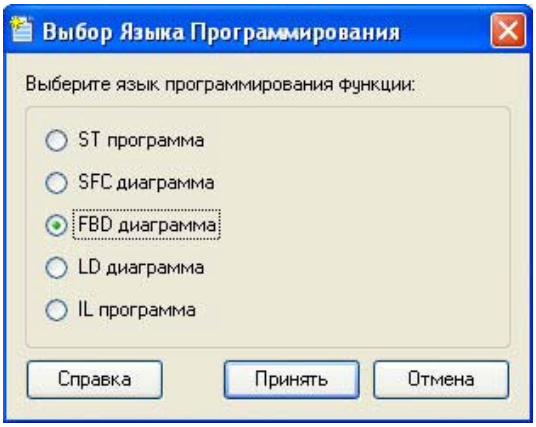


Рисунок 79 – Выбор языка программирования

По нажатию экранной кнопки **Принять** в открывшемся окне редактора программ с объявленными переменными создать программу в соответствии с заданием. Для выбора палитры FBD блоков необходимо ЛК мыши нажать на кнопку  после чего появится окно выбора FBD блоков рисунок 80. При разработки программы верхние входы FBD блоков не используются т.к. они предназначены для изменения порядка пересчета блоков, а информационными входами, являются входы начиная со второго.

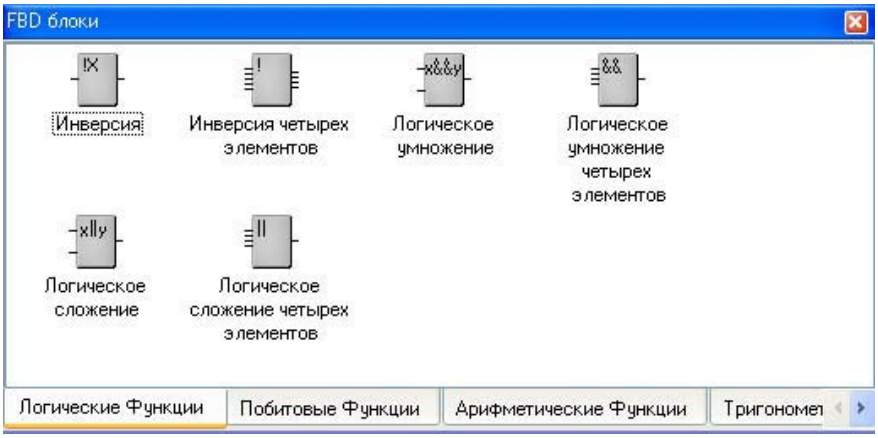


Рисунок 80 – Палитра FBD блоков

Для реализации логической функции выберем следующие блоки: из раздела *Логические Функции* **FBD блоки** инверсия (!X), логическое умножение (X&&Y и &&), логическое сложение (||). После размещения всех блоков для рассматриваемого примера программа будет выглядеть следующим образом рисунок 81.

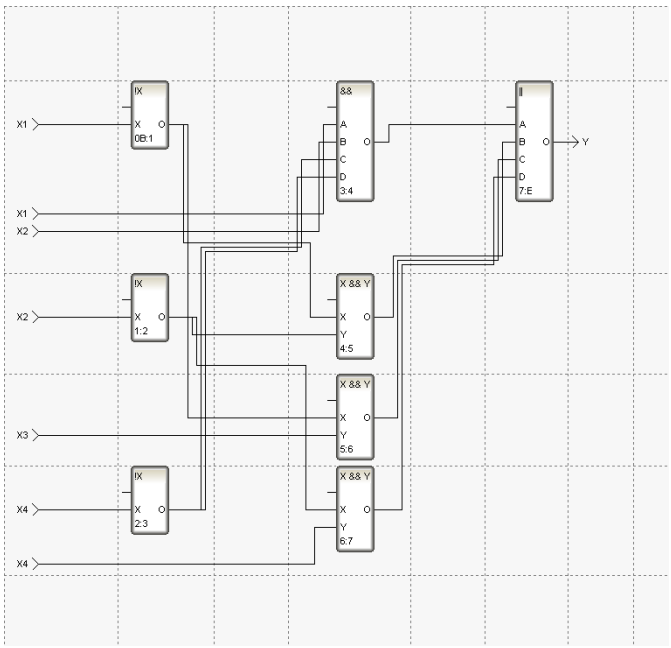


Рисунок 81 – Программа на языке **Техно FBD**

С помощью иконки  на инструментальной панели редактора или «горячей клавишей» F7 скомпилировать программу и убедиться в успешной компиляции в окне Выход (Output), вызываемого из инструментальной панели с помощью иконки  рисунок 82.

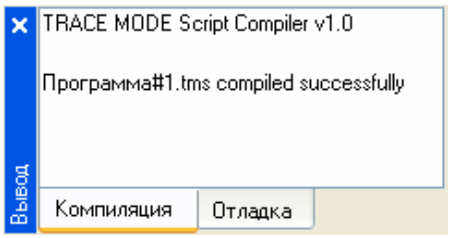


Рисунок 82 – Результат успешной компиляции программы

Привязка аргументов программы

Выполним привязку аргументов программы к атрибутам каналов. Вызвать свойства компонента **Программа#1** через контекстное меню. Выбрать вкладку **Аргументы**. Двойным нажатием в поле **Привязка** привязать аргументы программы к атрибутам каналов – аргументы X1, X2, X3, X4 к реальному значению каналов X1, X2, X3, X4 рисунок 83.

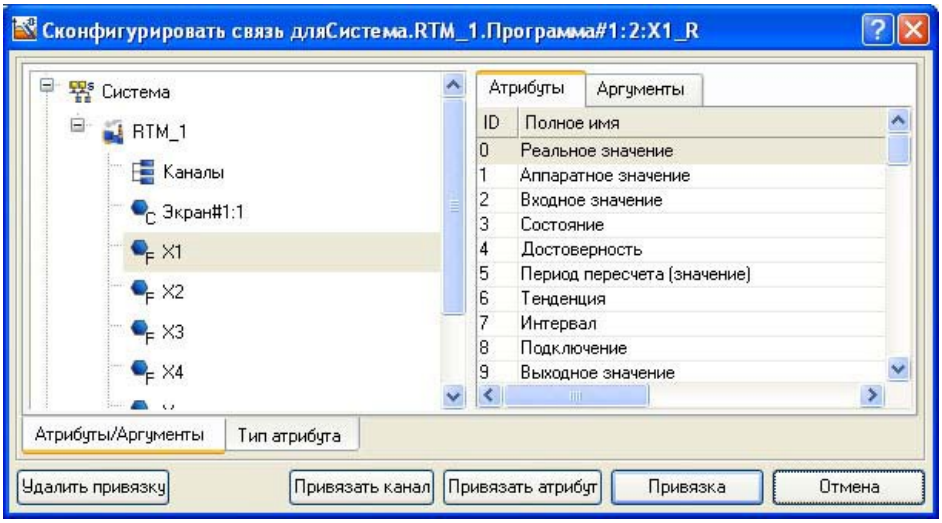


Рисунок 83 – Связь аргументов программы с аргументами экрана Двойным щелчком в поле **Привязка** аргумента программы Y вызвать окно
настройки связи, выбрать в левом окне канал класса «Вызов» **Экран#1**, а в правом окне выбрать вкладку **Аргументы** и указать в ней аргумент Y и кнопкой **Привязка** подтвердить связь рисунок 84.

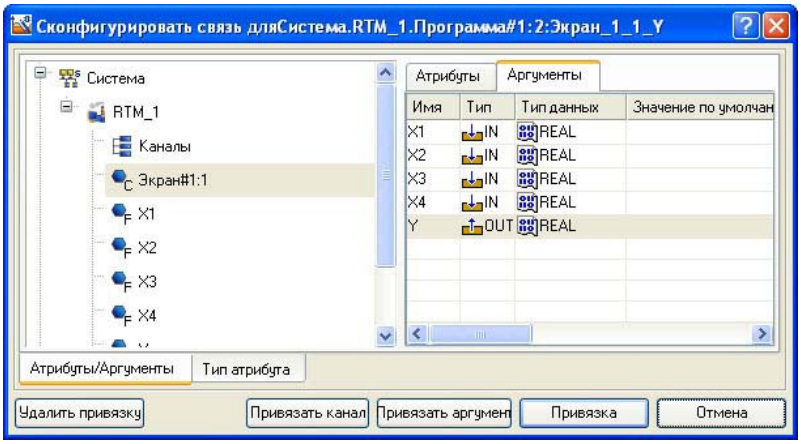


Рисунок 84 – Связь выходных аргументов В результате, будем иметь рисунок 85.

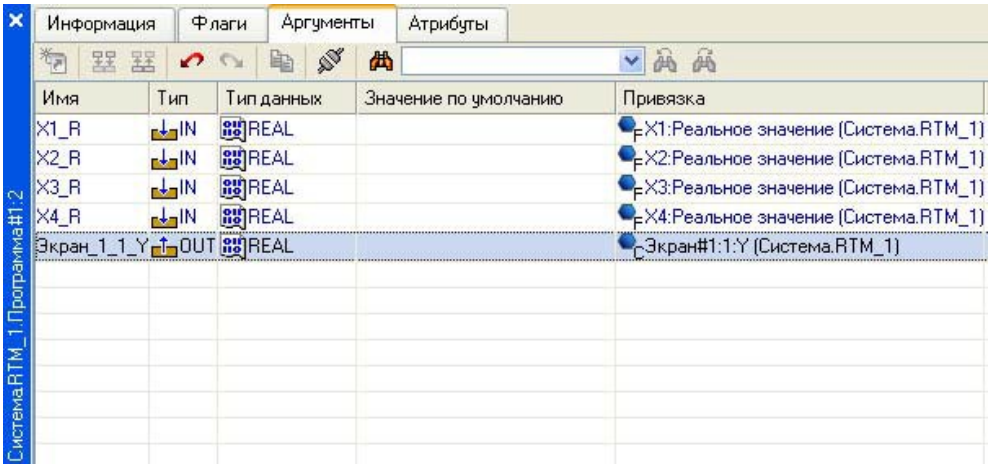


Рисунок 85 – Окончательная настройка связи После закрыть окно свойств компонента **Программа#1.**

Запуск проекта

Сохранить проект с помощью иконки . На инструментальной панели выбрать команду и скомпилировать тем самым проект для запуска в реальном времени. В навигаторе проекта выделить узел RTM_1 рисунок 86.

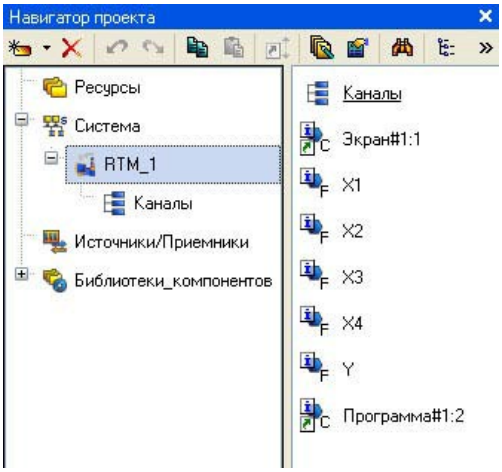


Рисунок 86 – Навигатор проекта

Выбрать иконку на инструментальной панели и запустить режим исполнения. Для составления отчета необходимо в меню **Файл** выполнить команду **Документировать проект** полученный *.html файл необходимо распечатать.

Контрольные вопросы

1. SCADA система TRACE MODE
2. Принцип работы монитора. Канал TRACE MODE 6
3. Обеспечение работы распределенных АСУ
4. Резервирование
5. Автопостроение
6. Математическая обработка данных
7. Архивирование каналов узла
8. Архивирование каналов проекта

Лабораторная работа №3 Реализация одноконтурной системы автоматического регулирования при помощи SCADA–системы TRACE MODE

Цель работы: освоить методику программирования одноконтурной системы автоматического регулирования при помощи SCADA–системы TRACE MODE на языке Техно FBD.

Порядок выполнения лабораторной работы

Рассмотрим реализацию одноконтурной системы автоматического регулирования при помощи SCADA–системы TRACE MODE на языке Техно FBD.

Разработка любого проекта автоматизации всегда начинается с запуска Интегрированной среды разработки (ИСР). Для ее запуска необходимо выполнить команду: TRACE MODE IDE 6 (base) из группы установки инструментальной системы в меню Программы WINDOWS или двойным щелчком ЛК мыши по иконке  рабочего стола Windows рисунок 87.

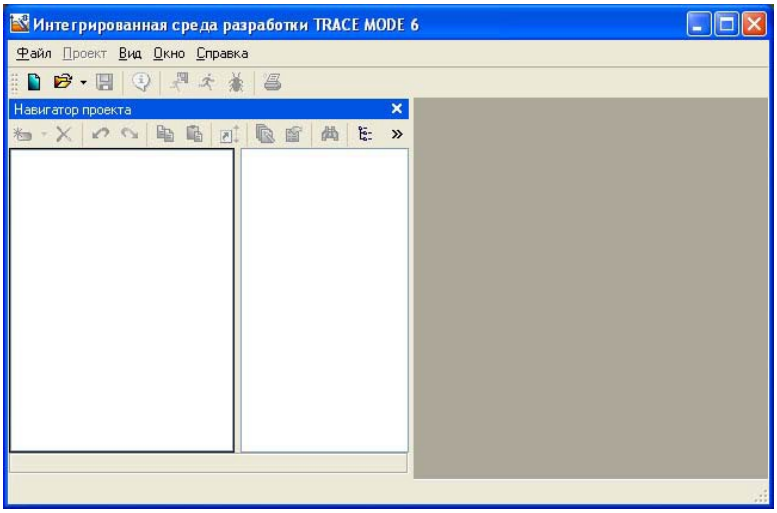


Рисунок 87– Интегрированная среда разработки

После запуска ИСР в меню **Файл** выбрать команду *Настройки ИС…*

В появившемся окне выбрать *Уровень сложности* и настроить как показано на рисунке 88:

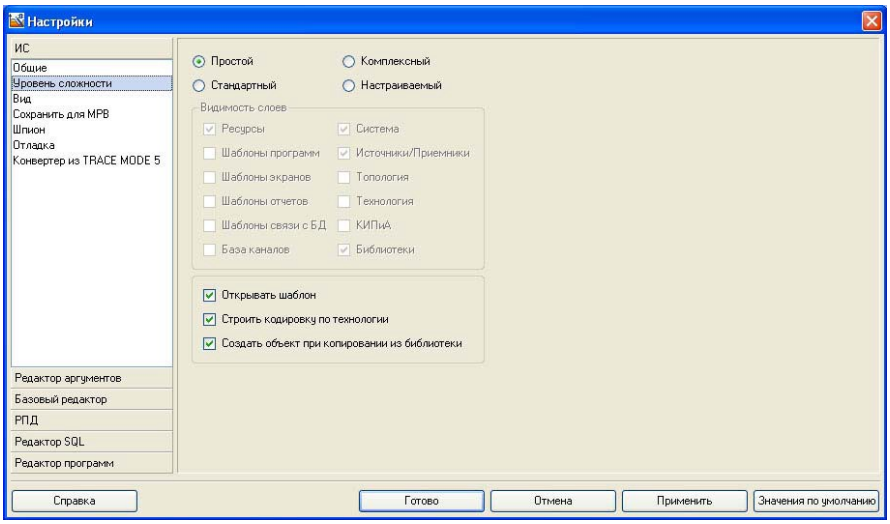


Рисунок 88 – Настройки ИСР

затем выбрать *Отладка* и настроить как показано на рисунке 89.

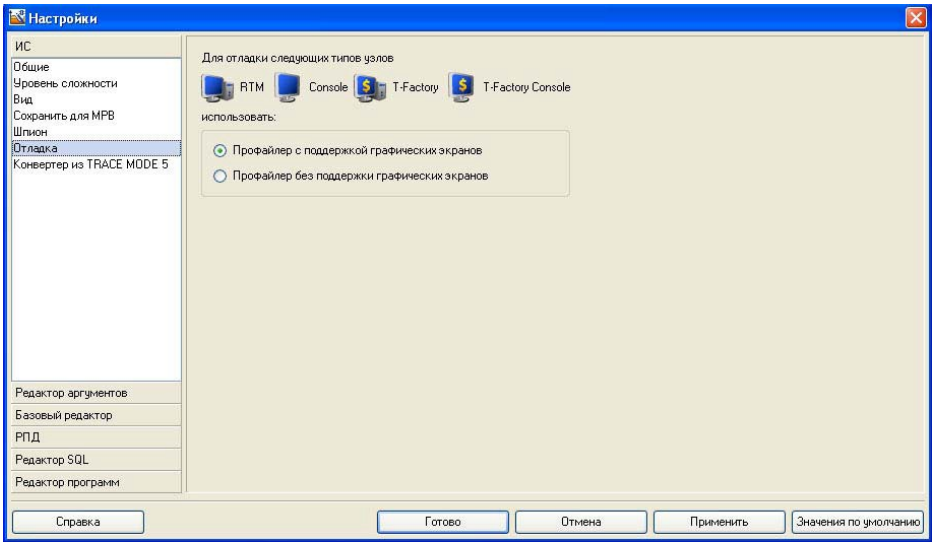


Рисунок 89 – Настройки ИСР

После проведенных настроек ИСР нажать кнопку Готово.

Нажатием левой клавиши мыши по иконке  инструментальной панели создадим новый проект при этом в открывшемся на экране диалоге рисунок 90.

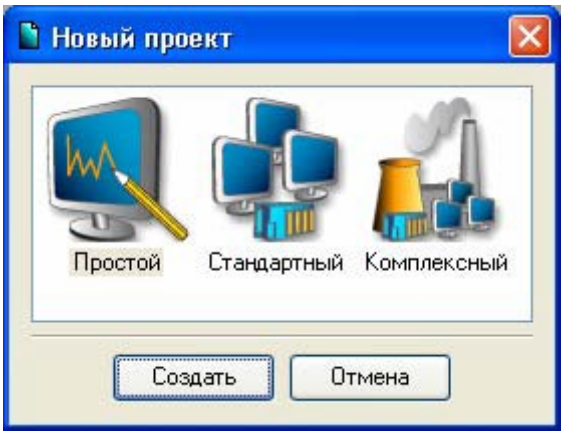


Рисунок 90 – Выбор типа проекта

Выберем **Простой** стиль разработки. После нажатия левой клавиши мыши (ЛК) на экранной кнопке ***Создать***, в левом окне Навигатора проекта появится дерево проекта, с созданным узлом **АРМ RTM_1**. Откроем узел **RTM_1** двойным щелчком ЛК, при этом, в правом окне **Навигатора проекта** отобразится содержимое узла – пустая группа **Каналы** и один канал класса «Вызов» **Экран#1**, предназначенный для отображения на узле **АРМ** графического экрана рисунок 91.

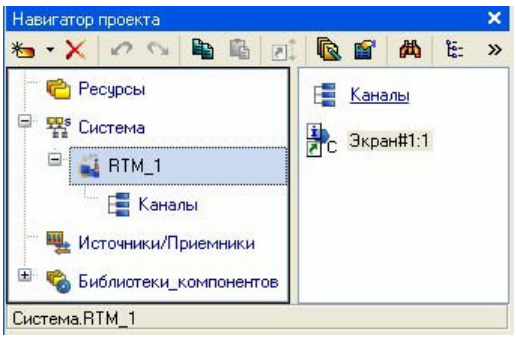


Рисунок 91 – Навигатор проекта

Создание графического экрана

Двойным щелчком ЛК на компоненте **Экран#1** открываем окно графического редактора. Подготовим на экране вывод динамического текста для отображения численного значения входных переменных апериодического звена первого порядка: коэффициента усиления объекта (Kob), постоянной времени объекта (Tob), времени запаздывания (Tau); ПИД регулятора: коэффициент усиления регулятора (Kr), времени интегрирования (Ti), времени дифференцирования (Td); и величины задания (Zad).

Выбрать на инструментальной панели графического редактора иконку ГЭ Кнопка -  . С помощью мыши разместить семь элементов в поле экрана как показано на рисунке 92.

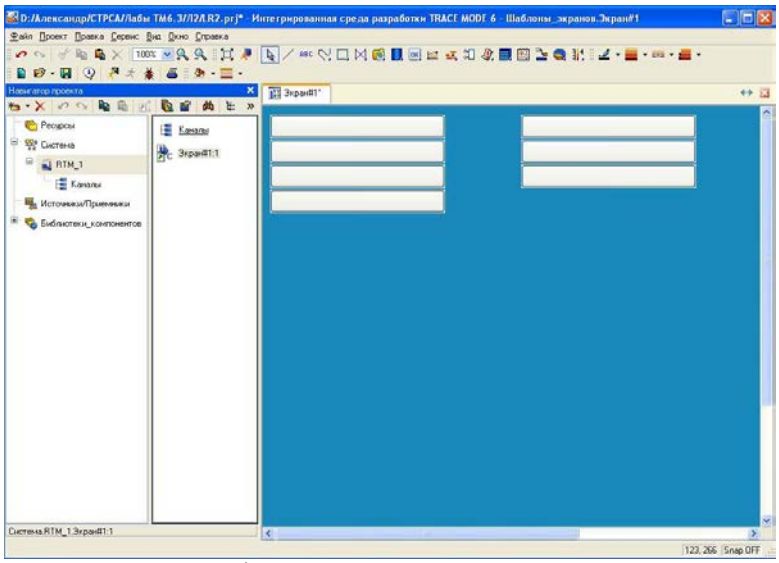


Рисунок 92 – Графический экран

Перейти в режим редактирования нажатием  кнопки, затем двойным щелчком ЛК вызвать окно свойств первого ГЭ рисунок 93.

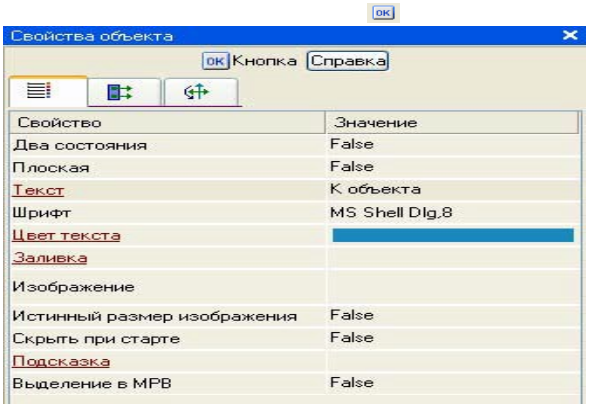


Рисунок 93 – Свойства графического элемента

В поле **Текст** ввести «К объекта». Открыть бланк **Действия**  и (правой кнопкой мыши) ПК раскрыть меню События **По нажатию** (**mousePressed**). Выбрать из списка команду **Добавить Передать значение**, раскрыть меню настроек выбранной команды рисунок 94.

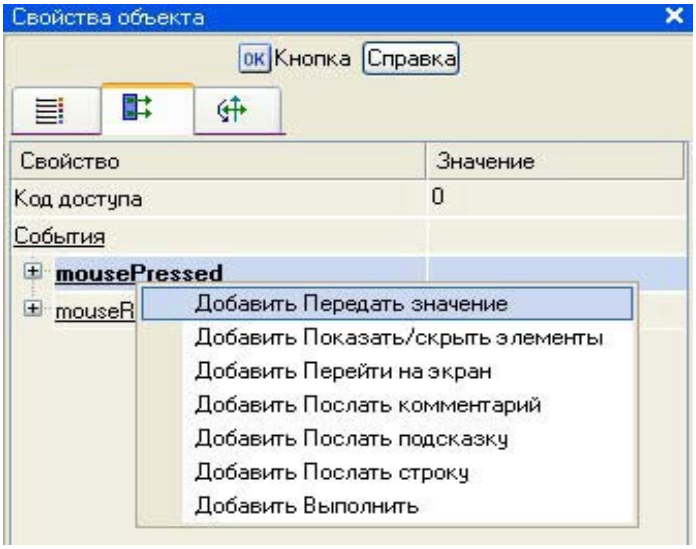


Рисунок 94 – Настройка типа передачи

В поле **Тип передачи (Send Type)** выбрать из списка **Ввести и передать (Enter & Send)** рисунок 95.

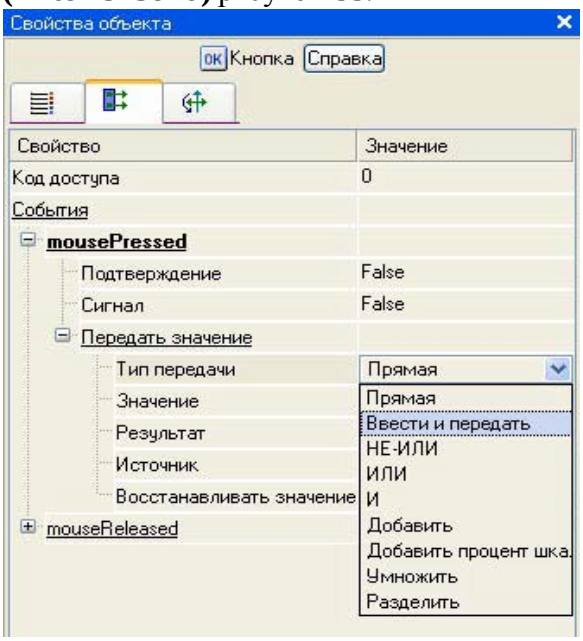


Рисунок 95 – Настройка типа передачи

ЛК в поле **Результат** вызвать табличный редактор аргументов. В открывшемся окне **Свойство привязки**, нажав кнопку  на его панели инструментов, создать восемь аргументов экрана 7 входных и один выходной. Входной (IN) или Выходной (OUT) будет аргумент, выбирается из низпадающего меню, появляющегося при двойном щелчке ЛК по типу аргумента (рисунок 96). Двойным щелчком ЛК выделить имя аргумента и изменить его, введя с клавиатуры «Kob» (завершить ввод нажатием клавиши Enter). Подтвердить связь с этим аргументом нажатием кнопки **Готово** Аналогичным образом настроить все остальные аргументы.

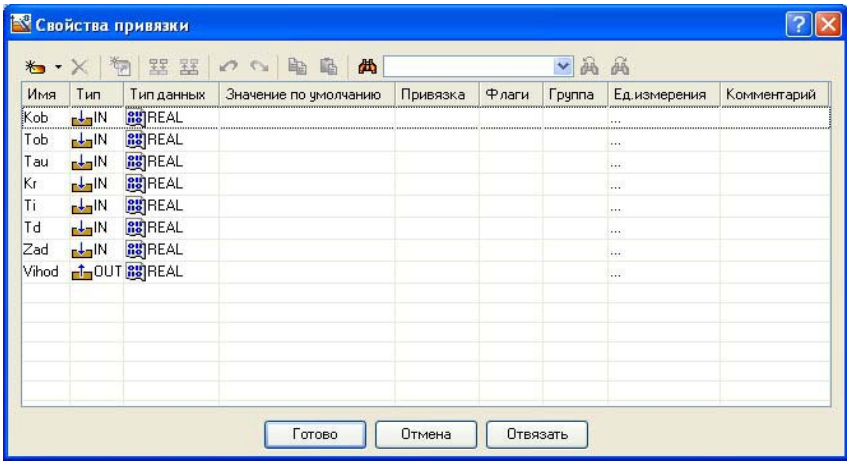


Рисунок 96 – Создание аргументов экрана

После привязки окно Свойства объекта выглядит следующим образом рисунок 97.

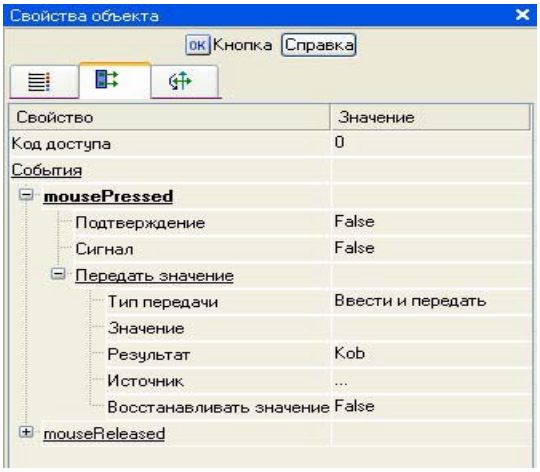


Рисунок 97 – Окончательный вид окна свойств графического элемента

Zad.

Аналогичным образом настроить остальные кнопки Tob, Tau, Kr, Ti, Td,

Для отображения числового значения входных аргументов создадим и разместим семь ГЭ как показано на рисунке 98.

Для этого активировав ГЭ ABC нажатием ЛК, рисуем области вывода динамического текста, задавая левый верхний и правый нижний углы щелчком ЛК.

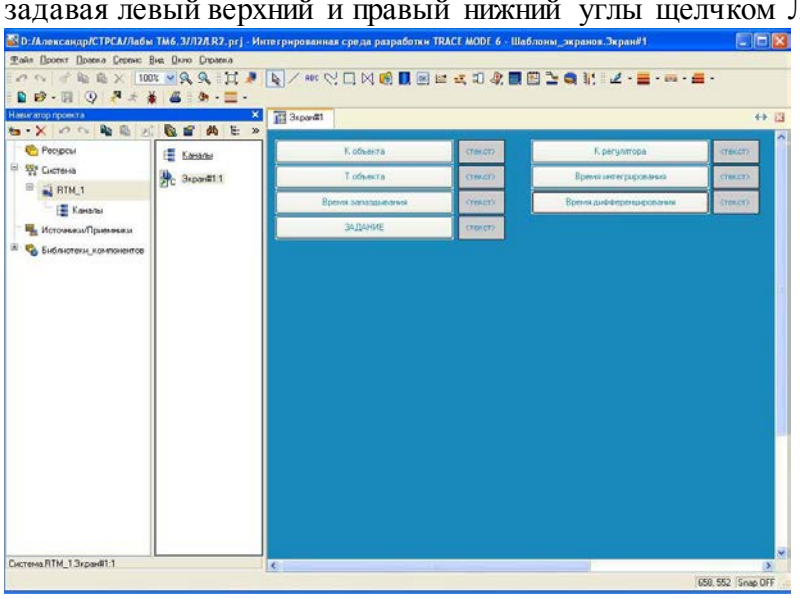


Рисунок 98 – Создание графических элементов

После этого необходимо переключиться в режим редактирования активацией ГЭ  . Двойным щелком ЛК на строке Текст вызвать меню **Вид индикации** рисунок 99;

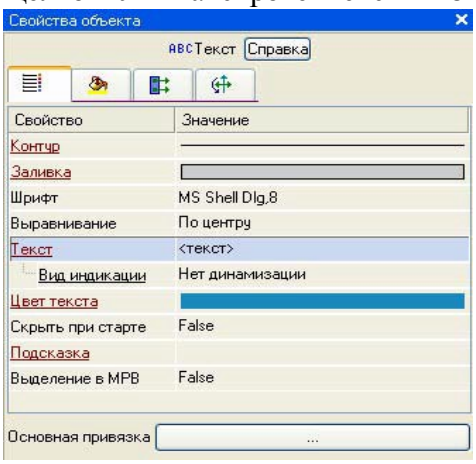


Рисунок 99 – Настройка индикации

В окне «Свойства объекта», двойным щелчком ЛК по свойству **Текст** развернуть его. В появившемся подпункте **Вид индикации** щелкнув по значению **Нет динамизации** выбрать его тип **Значение** (рисунок 100).

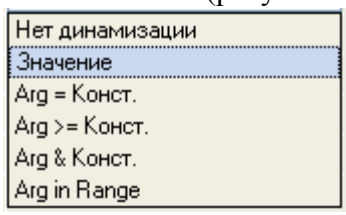


Рисунок 100 – Выбор типа динамизации

В открывшемся меню настройки параметров динамизации рисунок 101.

Рисунок 101 – Окно привязки Выбрать свойство **Привязка** и связать с Kob.

Дополним созданный экран новым ГЭ для совместного просмотра изменений значений каналов узла во времени и отслеживании предыстории – трендом.

На экрана разместим ГЭ Тренд для вывода значений **Zad** и **Vihod** рисунок 102. Основные свойства ГЭ оставим заданными по умолчанию. Перейдем во вкладку  и, выделив ЛК строку **Кривые**, с помощью ПК  создадим две новые кривые. Настроим их привязки к аргументам, толщину и цвет линий:

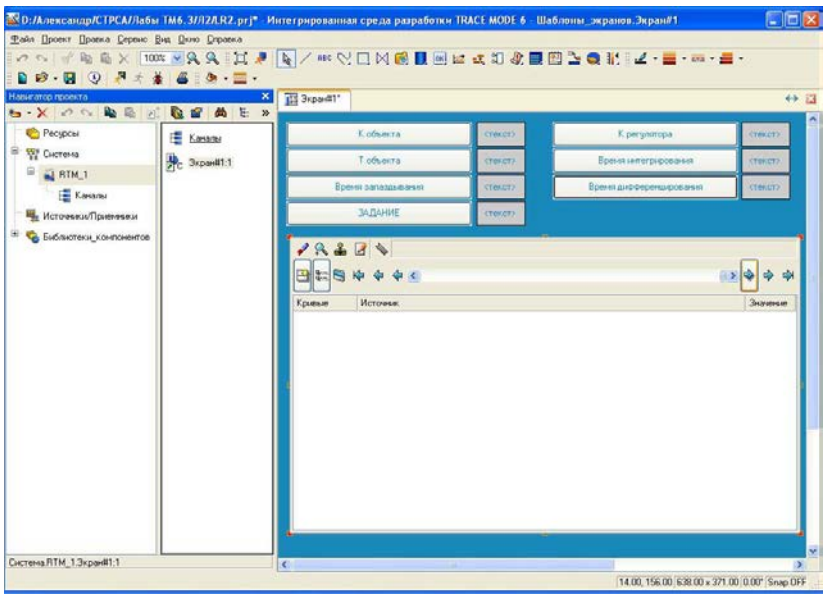


Рисунок 102 – Размещение тренда

Двойным щелчком ЛК мыши в поле тренда открываем окно **Свойства тренда** и переходим на вкладку **Кривые**. Нажимая ПК мыши в поле кривые задаем две кривые Zad и Vihod, а в поле привязка привязываем созданные кривые к заданию и выходу как показано на рисунке 103.

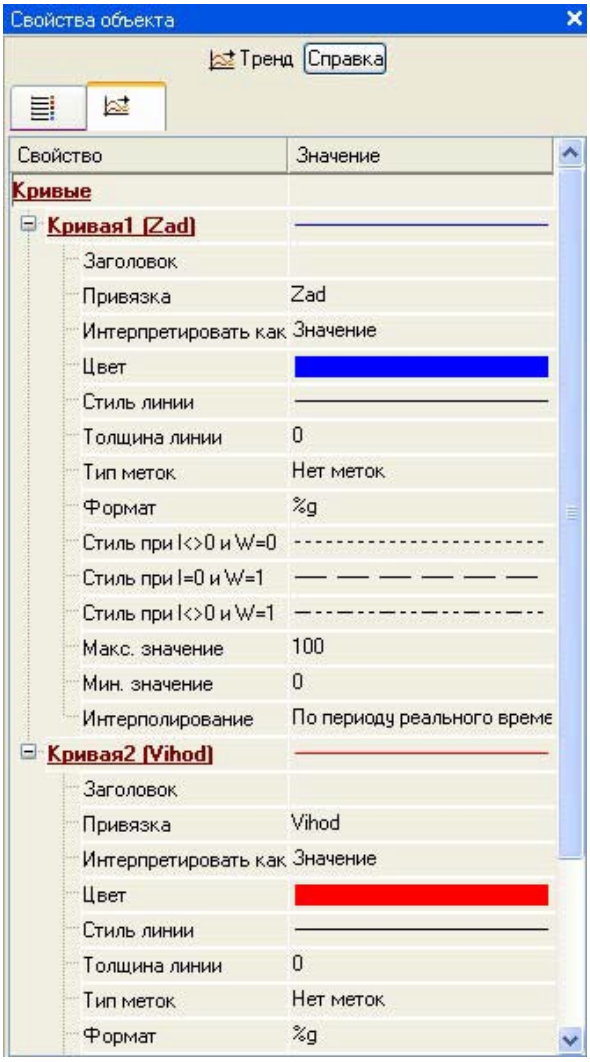


Рисунок 103 – Свойства тренда

Привязка аргумента экрана к каналу

Создадим по аргументам Kob, Tob, Tau, Kr, Ti, Td, Zad, Vihod, шаблона экрана новые каналы и отредактируем их привязку. В слое Система открыть узел RTM_1. С помощью ПК вызвать контекстное меню свойства компонента Экран#1 и выбрать команду Свойства рисунок 104. Выбрать вкладку Аргументы, ЛК, при нажатой клавише Cntrl, выделить аргументы Kob, Tob, Tau, Kr, Ti, Td, Zad, Vihod и с помощью иконки создать новые каналы.

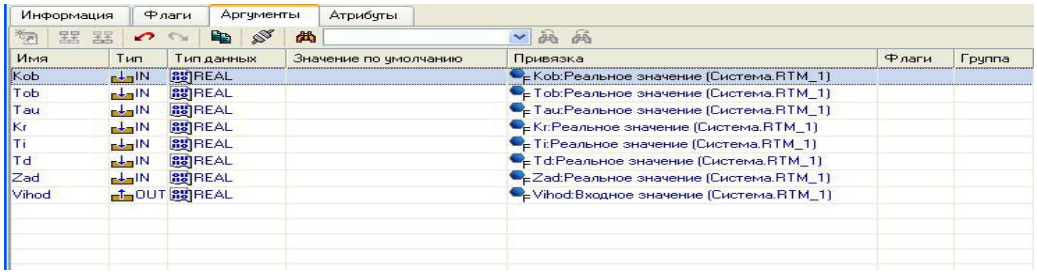


Рисунок 104 – Окно свойств экрана

В результате, в узле **RTM_1**, будут автопостроены следующие каналы рисунок 105.

Рисунок 105 – Автопостроены каналы

Создание программы на языке Техно FBD

Для создания FBD-программы и подключения ее к проекту нужно выполнить следующие операции:

– разместить необходимые функциональные блоки в рабочем поле FBD-редактора;

– соединить нужные входы и выходы блоков, образовав единую диаграмму;

– задать аргументы, переменные и константы программы;

– привязать входы/выходы FBD-диаграммы к аргументам, переменным и константам программы;

– скомпилировать программу

Создадим программу, которая будет реализовывать одноконтурную систему автоматического регулирования. Для этого ПК по узлу **RTM_1 вызвать контекстное меню выбрать подменю «Создать компонент»** и выбрать ЛК в нем компонент **Программа** рисунок 106.

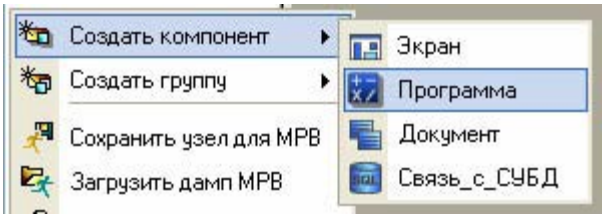


Рисунок 106 – Создание компонента Программа

Выделить компонент Программа#1 и ПК вызвать контекстное меню, выбрав в котором ЛК пункт **Редактировать шаблон**, перейти в режим редактирования программы рисунок 107.

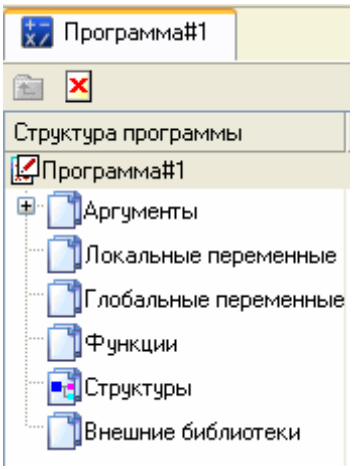


Рисунок 107 – Редактирование аргументов Программы

Выделением ЛК в дереве шаблона **Программа#1** строки **Аргументы** вызвать табличный редактор аргументов иконкой  и создать в редакторе аргументов следующие аргументы программы Kob, Tob, Tau, Kr, Ti, Td, Zad,. При этом аргументы Kob, Tob, Tau, Kr, Ti, Td, Zad должны быть типа **IN**, а Vihod – **OUT** рисунок 108.

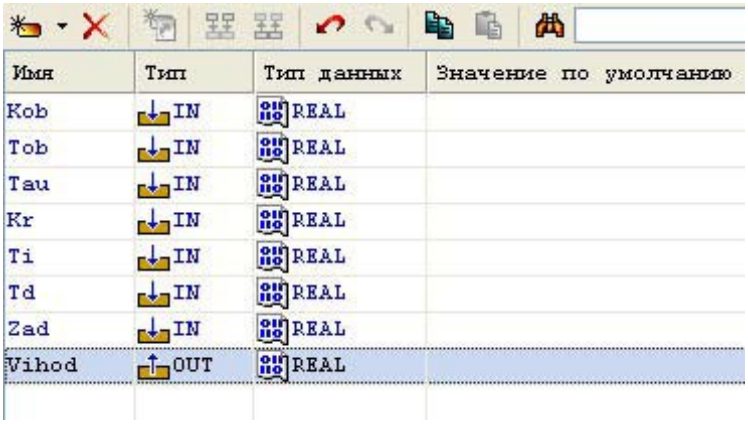


Рисунок 108 – Аргументы программы

Щелкнув ЛК в дереве шаблона строку **Программа#1** и в открывшемся диалоге Выбор языка выбрать язык **FBD** рисунок 109.

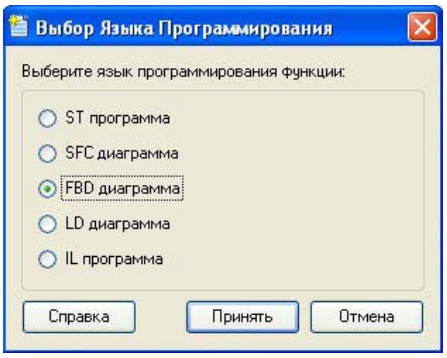


Рисунок 109 – Выбор языка программирования

По нажатию экранной кнопки **Принять** в открывшемся окне редактора программ с объявленными переменными создать программу в соответствии с заданием. Для выбора палитры FBD блоков необходимо ЛК мыши нажать на кнопку  после чего появится окно выбора FBD блоков рисунок 110. При разработки программы верхние входы FBD блоков не используются т.к. они предназначены для изменения порядка пересчета блоков, а информационными входами, являются входы начиная со второго.

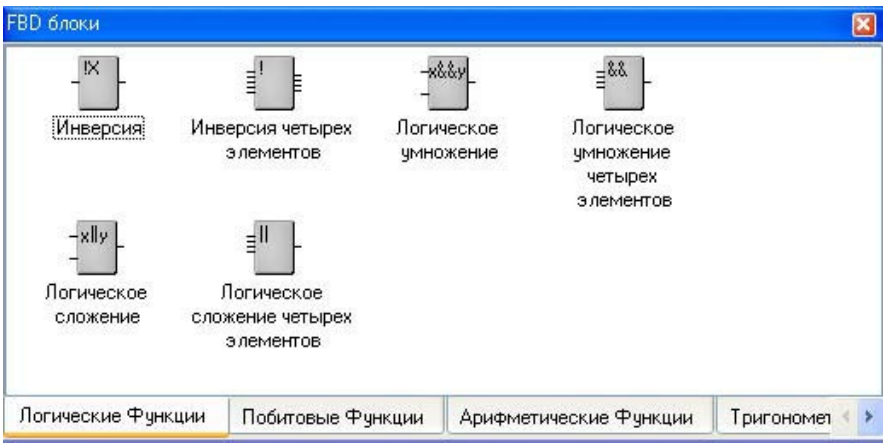


Рисунок 110 – Палитра FBD блоков

Для создания системы управления выберем следующие блоки: из раздела *Арифметические Функции* **FBD блок** вычитание (X–Y); из раздела *Регулирование* **FBD блок** модель объекта (OBJ) и звено PID (PID). После размещения всех блоков программа будет выглядеть следующим образом рисунок 111.

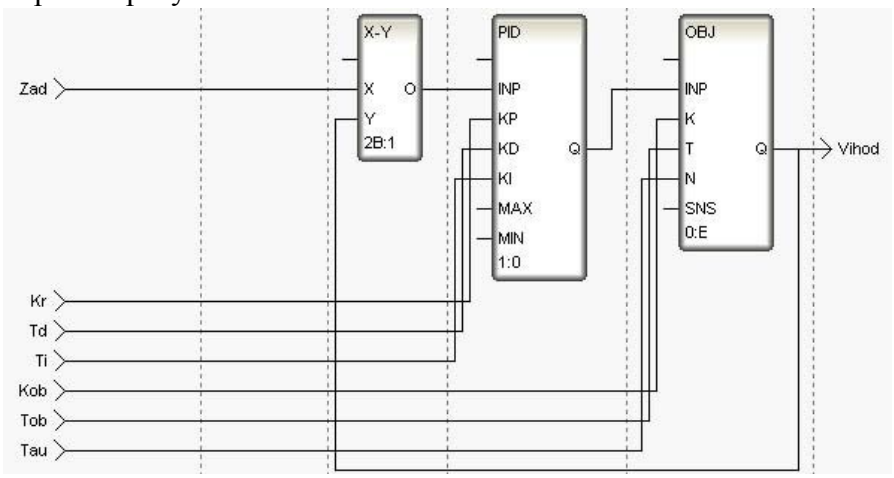


Рисунок 111 – Программа реализации системы управления

С помощью иконки  на инструментальной панели редактора или «горячей клавишей» F7 скомпилировать программу и убедиться в успешной компиляции в окне Выход (Output), вызываемого из инструментальной панели с помощью иконки  рисунок 112.

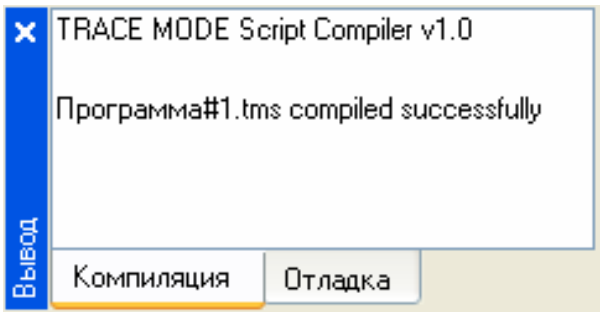


Рисунок 112 – Результат успешной компиляции программы

3.3.4 Привязка аргументов программы

Выполним привязку аргументов программы к атрибутам каналов. Вызвать свойства компонента **Программа#1** через контекстное меню. Выбрать вкладку **Аргументы**.

Двойным щелчком в поле **Привязка** аргумента программы Y вызвать окно настройки связи, выбрать в левом окне канал класса «Вызов» **Экран#1**, а в правом окне выбрать вкладку **Аргументы** и указать в ней аргумент Vihod и кнопкой **Привязка** подтвердить связь рисунок 113.

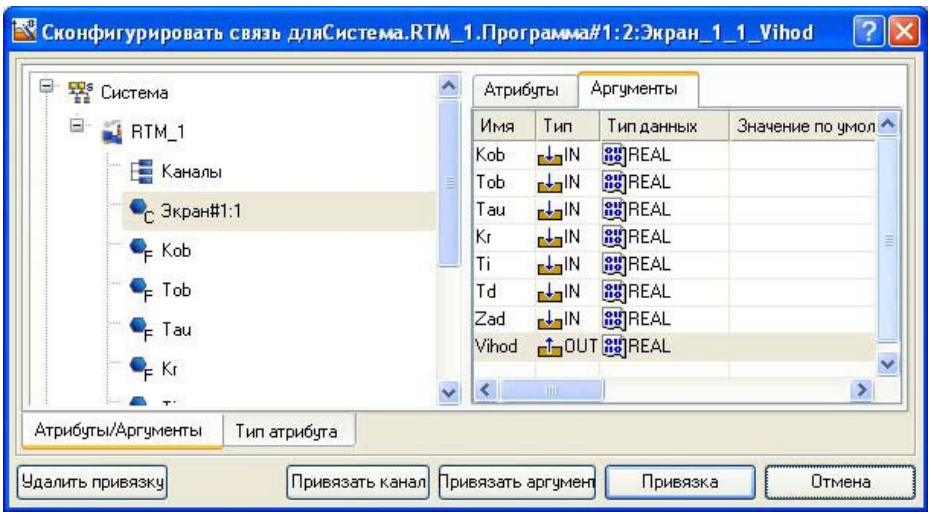


Рисунок 113 – Связь выходных аргументов

Аналогично в поле **Привязка** привязать аргументы программы к атрибутам каналов – аргументы Kob, Tob, Tau, Kr, Ti, Td, Zad к реальному значению каналов Kob, Tob, Tau, Kr, Ti, Td, Zad рисунок 114.

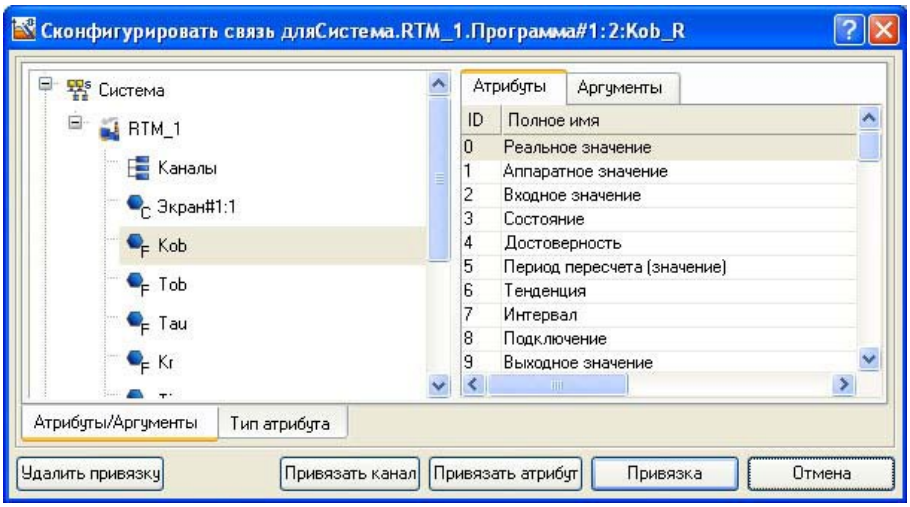


Рисунок 114 – Связь аргументов программы с аргументами экрана В результате, будем иметь рисунок 115.

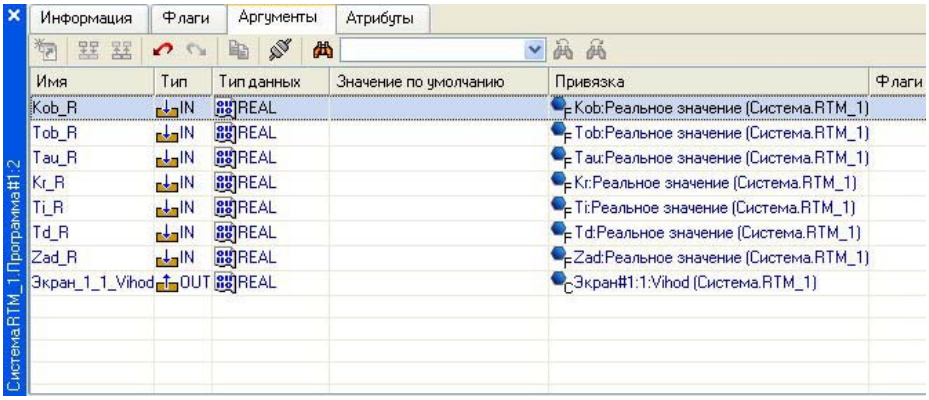


Рисунок 115 – Окончательная настройка связи После закрыть окно свойств компонента **Программа#1.**

Запуск проекта

Сохранить проект с помощью иконки  . На инструментальной панели выбрать команду  (сохранить для МРВ) и скомпилировать тем самым проект для запуска в реальном времени. В навигаторе проекта выделить узел RTM_1 рисунок 116.

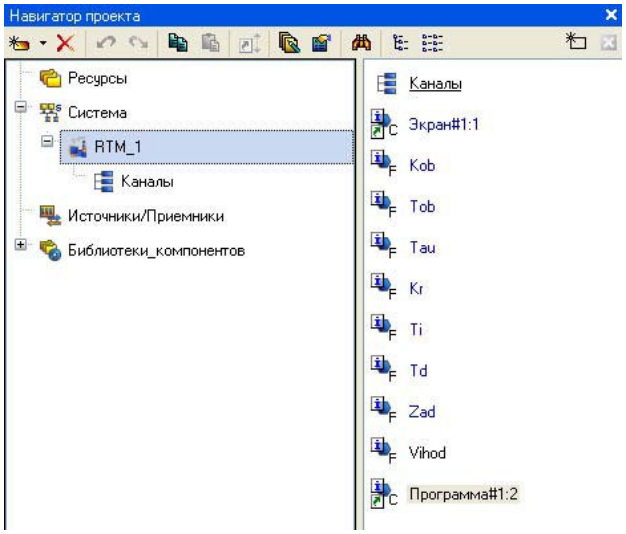


Рисунок 116 – Навигатор проекта

Выбрать иконку  на инструментальной панели и запустить профайлер.

В режиме исполнения запустить программу с помощью иконки  .

Для составления отчета необходимо в меню **Файл** выполнить команду **Документировать проект** полученный *.html файл необходимо распечатать.

Контрольные вопросы

1. Принцип работы монитора. Канал TRACE MODE 6
2. Обеспечение работы распределенных АСУ
3. Резервирование
4. Автопостроение
5. Математическая обработка данных
6. Архивирование каналов узла
7. Архивирование каналов проекта
8. Отчет тревог и генерация сообщений
9. Графический интерфейс оператора

Лабораторная работа №4 Синтаксис Техно IL

Лаб 4
Синтаксис Техно IL

Программа на языке **Техно IL** представляет собой последовательность **инструкций**. Каждая инструкция должна начинаться с новой строки и должна содержать **оператор** с опциональным **модификатором** и, для некоторых операций, один или более **операндов**, разделенных пробелами. Между инструкциями могут располагаться пустые строки. Компилятор не чувствителен к регистру, т.е. инструкции **add var_002** и **ADD VAR_002** равнозначны.

Примеры IL-инструкций

```
ADD VAR_000 2.6
LT VAR_000 VAR_001
JMPC label1
GT VAR_001 20
JMPC label2
LD 278
label1: CAL FUNCTION_000(VAR_000, VAR_001)
label2: ST VAR_001
```

Под **аккумулятором** в **Техно IL** понимается хранилище текущего результата вычислений (в этом качестве выступает один из регистров процессора). Далее в описании языка **Техно IL** значение аккумулятора обозначается словом **result**. Функция на языке **Техно IL** возвращает **result**.

Техно IL поддерживает одноадресный и двухадресный режимы записи инструкций, которые оперируют с двумя операндами. В первом случае первым операндом является аккумулятор, который опускается при записи, во втором случае указываются два операнда.

Пример

В данном примере представлена запись процедуры a = a + b в одноадресном и двухадресном режиме. Одноадресный режим:

```
LD a    //result = a
ADD b   //result = result + b
ST a    //a = result
```

Двухадресный режим позволяет записать ту же операцию компактнее:

```
ADD a b  // a = a + b
```

В IL-программе могут использоваться метки и комментарии. Правила их задания аналогичны правилам **Техно ST**.

Операторы и модификаторы Техно IL

К ним относятся:
– модификаторы Техно IL;
– операторы обмена с аккумулятором;
– логические операторы Техно IL;
– арифметические операторы Техно IL;
– операторы сравнения Техно IL;
– операторы перехода и вызова функции Техно IL.

Модификаторы Техно IL

Модификаторы **Техно IL** – это литеры **N, C** и **X**, которые могут быть приписаны справа к имени ряда операторов.

Модификатор **N** обозначает логическое отрицание операнда. Например, инструкция

AND a
интерпретируется как **result = result AND a**, а инструкция
ANDN a
интерпретируется как **result = result AND NOT a**.
Для операторов **JMP**, **CAL** и **RET**:
– модификатор **C** обозначает, что инструкция выполняется в том случае, если результат предыдущей операции сравнения истинен;
– модификатор **X** обозначает, что инструкция выполняется в том случае, если **result** = TRUE.

Операторы обмена с аккумулятором

В таблице 1 представлены операторы обмена с аккумулятором.

Таблица 1

Синтаксис	Допустимый модификатор	Действие
LD operand	N	result := operand
ST operand	N	operand := result

Знак ":=" в таблице 1 обозначает операцию присваивания.
В качестве операнда может использоваться численная или булева переменная. В качестве операнда оператора **LD** может использоваться число.
Отличное от нуля значение аккумулятора интерпретируется как TRUE, нулевое – как FALSE, поэтому значение аккумулятора может быть присвоено как численной, так и булевой переменной.

Пример

```
VAR VAR_000 : INT := 10; END_VAR
VAR VAR_001 : BOOL := TRUE; END_VAR
VAR VAR_002 : BOOL; END_VAR
LD 8          //result := 8
ST VAR_000    //VAR_000 := 8
ST VAR_002    //VAR_002 := TRUE
LD 0          //result := 0
ST VAR_001    //VAR_001 := FALSE
LD VAR_001    //result := FALSE
ST VAR_002    //VAR_002 := FALSE
ST VAR_000    //VAR_000 := 0
```

Логические операторы Техно IL

В таблице 2 представлены логические операторы.

Таблица 2

Синтаксис	Допустимый модификатор	Действие
S operand		operand := TRUE (см. примечание)
R operand		operand := FALSE (см. примечание)
AND operand1 operand2	N	result := operand1 := operand1 AND operand2
OR operand1	N	result := operand1 := operand1 OR

Синтаксис	Допустимый модификатор	Действие
operand2		operand2
XOR operand1 operand2	N	result := operand1 := operand1 XOR operand2

Примечание. Оператор выполняется только тогда, когда **result** = TRUE.

В качестве операндов могут использоваться булевы переменные. Вторым операндом может быть число (но не численная переменная), которое интерпретируется следующим образом: не равно 0 – TRUE; равно 0 – FALSE.

Выполнение операторов **R** и **S** не изменяет значения аккумулятора.

Пример

```
VAR VAR_001 : BOOL := TRUE; END_VAR
VAR VAR_002 : BOOL; END_VAR
VAR VAR_004 : INT := 0; END_VAR
LD 1              //result:=1
S VAR_002         //VAR_002:=TRUE
R VAR_002         //VAR_002:=FALSE
AND VAR_001 VAR_002  //result:=VAR_001:=FALSE
LD 1              //result:=1
S VAR_001         //VAR_001:=TRUE
OR VAR_002 VAR_001   //result:=VAR_002:=TRUE
XOR VAR_002 VAR_001  //result:=VAR_002:=FALSE
OR VAR_002 10        //result:=VAR_002:=TRUE
```

Арифметические операторы Техно IL

В таблице 3 представлены арифметические операторы.

Таблица 3

Синтаксис	Действие
ADD operand1 operand2	result := operand1 := operand1 + operand2
SUB operand1 operand2	result := operand1 := operand1 – operand2
MUL operand1 operand2	result := operand1 := operand1 * operand2
DIV operand1 operand2	result := operand1 := operand1 : operand2

В качестве операндов используются численные переменные, в качестве второго операнда может использоваться число.

Арифметические операторы не допускают использования модификаторов.

Пример

```
VAR VAR_000 : REAL := 20; END_VAR
VAR VAR_001 : LREAL := 30; END_VAR
ADD VAR_000 10       //result := VAR_000 := 30
MUL VAR_001 9        //result := VAR_001 := 270
SUB VAR_001 VAR_000  //result := VAR_001 := 240
DIV VAR_001 VAR_000  //result := VAR_001 := 8
```

Операторы сравнения Техно IL

В таблице 4 представлены операторы сравнения.

Таблица 4

Синтаксис	Действие
GT operand1 operand2	result := TRUE, если operand1 > operand2
GE operand1 operand2	result := TRUE, если operand1 >= operand2
EQ operand1 operand2	result := TRUE, если operand1 == operand2
NE operand1 operand2	result := TRUE, если operand1 <> operand2
LE operand1 operand2	result := TRUE, если operand1 <= operand2
LT operand1 operand2	result := TRUE, если operand1 < operand2

В качестве операндов используются численные переменные, в качестве второго операнда может использоваться число.

Операторы сравнения не допускают использования модификаторов.

Операторы сравнения, как правило, предшествуют операторам **JMPC**, **CALC** и **RETC**. Если результат сравнения ложен, инструкции сравнения не изменяют значения аккумулятора, а последующий оператор **JMPC**, **CALC** или **RETC** игнорируется (даже если **result** = TRUE).

Пример

```
VAR VAR_000 : INT := 20; END_VAR
VAR VAR_001 : INT := 30; END_VAR
VAR VAR_002 : BOOL; END_VAR
LD 1              //result := TRUE
GT VAR_000 VAR_001 //результат сравнения ложен,
//т.к. (20<30), аккумулятор сохраняет свое
//значение (TRUE)
RETC              //RETC игнорируется
LD VAR_002        //result := FALSE
LT VAR_000 VAR_001 //результат сравнения истинен
//result := TRUE
CALC fff(VAR_000)  //вызов функции произойдет
```

Операторы перехода и вызова функции Техно IL

В таблице 5 представлены операторы перехода и вызова функции.

Таблица 5

Синтаксис	Допустимый модификатор	Действие
JMP имя_метки	C, X	переход к строке с указанной меткой
CAL имя_функции(val1, ... valN)	C, X	вызов функции или функции-блока
RET	C, X	выход из программы, функции или функции-блока

Операторы перехода выполняются, если строка с указанной меткой находится в том же программном компоненте.

CAL и **CALL** являются равнозначными операторами. В круглых скобках через запятую указываются значения, передаваемые в функцию. Между именем функции и круглыми скобками пробел необязателен. Число передаваемых в функцию значений должно быть равно числу аргументов, заданных для этой функции.

При выполнении оператора **RET** и его разновидностей функция возвращает значение **result.**

Определены следующие модификации данных операторов:

JMP, CAL, RET – соответственно оператор безусловного перехода, безусловного вызова и безусловного выхода.

JMPX, CALX, RETX – соответственно оператор условного перехода, условного вызова и условного выхода. Инструкция, содержащая любой их этих операторов, выполняется только тогда, когда **result**=TRUE, в противном случае игнорируется.

JMPC, CALC, RETC – соответственно оператор условного перехода, условного вызова и условного выхода. Эти операторы следуют непосредственно за оператором сравнения. Инструкция, содержащая любой их этих операторов, выполняется только тогда, когда результат предыдущей операции сравнения истинен, в противном случае игнорируется.

Определение переменных и констант

Вид константы или переменной (глобальная, локальная) задается оператором, с помощью которого данная переменная (константа) определяется. Синтаксис операторов определения переменных предполагает обязательное указание типа данных:

```
//определение локальной строковой
//переменной myVar
VAR myVar: STRING; END_VAR
```

Тип данных определяет размер выделяемой памяти. Для указания типа в **Техно ST** определены следующие ключевые слова (в круглых скобках указано соответствие типу данных С):

BOOL (**bool**) – булево значение размерностью 1 байт (**true (1)** или **false (0**));

SINT (**__int8**) – целое со знаком размерностью 1 байт (**-128 ... 127**);

USINT (**unsigned __int8**) – целое без знака размерностью 1 байт (**0 ... 255**);

INT (**short**) – целое со знаком размерностью 2 байта (**-32768 ... 32767**);

UINT (**unsigned short**) – целое без знака размерностью 2 байта (**0 ... 65535**);

DINT (**long**) – целое со знаком (4 байта) (**-2147483648 ... 2147483647**);

UDINT (**unsigned long**) – целое без знака (4 байта) (**0 ... 4294967295**);

TIME, DATE, TIME_OF_DAY, DATE_AND_TIME – соответствуют **DINT.** Значения переменных этих типов задаются аналогично соответствующим временным константам;

REAL (**float**) – вещественное число (4 байта) (максимальное значение **3.402823466e+38**);

LREAL (**double**) – вещественное число (8 байт) (максимальное значение **1.7976931348623158e+308**);

STRING (**char** []) – 256 символов в кодировке UTF-8 (512 байт);

HANDLE – специальный тип, используемый для хранения внешних данных в виде числа, имеет размерность 4 байта, не может быть использован в арифметических, логических и т.п. операциях.

Кроме указанных типов, переменной может быть присвоен структурный тип, созданный пользователем. Такая переменная является конкретным объектом указанного типа.

При определении переменной может быть задано ее значение:

```
VAR i: INT:=0; END_VAR
```

Если при определении переменной ее значение не задано, то этой переменной по умолчанию присваивается следующее начальное значение:

– числовая переменная – 0;

– переменная типа **BOOL** – **FALSE**;

– переменная типа **STRING** – пустая строка;

– переменная типа **HANDLE** – 16#00000000 (0 в формате HEX);

– переменная типа **TIME, DATE, TIME_OF_DAY** или **DATE_AND_TIME** – 0.

При определении константы задание ее значения обязательно:

```
VAR CONSTANT myConst: INT:=13; END_VAR
```

В отличие от переменной, значение константы в программе изменять нельзя.

Особенности присвоения значений переменным

При присвоении значения переменной типа TYPE1 переменной типа TYPE2 нужно учитывать следующее:

присвоение корректно только в том случае, если тип TYPE2 заключает в себе все числа типа TYPE1:

```
VAR a: REAL := -1564.343;  END_VAR
VAR b: USINT := 50; END_VAR
a = b;  //корректная операция
b = a;  //некорректная операция
```

присвоение корректно, если один из типов – **BOOL**, а другой – любой численный. Логическое значение TRUE соответствует единице, FALSE – нулю; нуль соответствует FALSE, любое ненулевое значение, в том числе отрицательное, соответствует TRUE:

```
VAR a: BOOL; END_VAR
VAR b: SINT := -50; END_VAR
a = b; //a = TRUE, корректная операция
b = a; //b = 1, корректная операция
```

Операторы

Определены следующие операторы, образующие предложения **Техно ST**:

return
if
case
while
repeat
for
break
exit
continue
операторы определения переменных
оператор индексирования элементов массива
goto

Оператор return

Определены 2 варианта задания данного оператора:

return {выражение}

Действие: выход из функции **Техно ST**. Значением функции является значение {выражения};

return

Действие: выход из функции-блока **Техно ST.**

Пример

```
RETURN (2 + ARG_000 ** 2);
```

Оператор if-then-else

Данный оператор начинается с ключевого слова **if** и заканчивается ключевым словом **end_if.** Определены 3 варианта задания данного оператора:

Вариант 1

if {выражение} then {последовательность предложений} end_if

Действие: если {выражение} истинно, выполняется {последовательность предложений}, иначе никаких действий не производится.

Вариант 2

if {выражение} then {последовательность предложений1}

else {последовательность предложений2} end_if

Действие: если {выражение1} истинно, выполняется {последовательность предложений1}, иначе выполняется {последовательность предложений 2}.

Вариант 3

if {выражение1} then {последовательность предложений1}
elsif {выражение2} then {последовательность предложений2}
…
elsif {выражениеN} then {последовательность предложенийN}
else {последовательность предложений} end_if

Действие: выполняется первая по порядку {последовательность предложений}, для которой соответствующее {выражение} истинно. Если все {выражения} ложны, выполняется {последовательность предложений}, следующая за ключевым словом **else.**

Количество блоков "**elsif** {выражение} **then** {последовательность предложений}" не ограничено.

Пример

В результате выполнения следующего кода переменной VAR_000 присваивается значение 200. Выполняется только одно, первое по порядку действие, для которого условие истинно, поэтому действие, следующее за конструкцией **ELSIF...THEN**, выполнено не будет, несмотря на то, что условие VAR_002 < 1 истинно:

```
VAR VAR_000 : INT; END_VAR
VAR VAR_002 : REAL := 0.5; END_VAR
IF VAR_002 < 2 THEN
VAR_000 = 200;
ELSIF VAR_002 < 1 THEN
VAR_000 = 500;
ELSE
VAR_000 = 300;
END_IF;
```

Оператор case

Определены 2 варианта задания данного оператора.

Вариант 1

case {выражение} of
{список значений}: {последовательность предложений}
...
{список значений}: {последовательность предложений}
end_case

Вариант 2

case {выражение} of
{список значений}: {последовательность предложений}
...
{список значений}: {последовательность предложений}
else {последовательность предложений}
end_case

Список значений представляет собой набор целых чисел или набор диапазонов целых чисел, разделенных запятой. Диапазон задается в виде

{нижняя граница} .. {верхняя граница}

Действие: если результат вычисления {выражения} принадлежит множествам заданным {списками значений}, выполняется соответствующая {последовательность предложений}. Если

результат вычисления {выражения} не принадлежит ни одному из заданных множеств, выполняется {последовательность предложений}, следующая за ключевым словом **else.**

Пример

В результате выполнения следующего кода VAR_001=500:

```
VAR VAR_000 : INT; END_VAR
VAR VAR_001 : INT; END_VAR
CASE VAR_000 + 4 OF
0 .. 2 : VAR_001 = 200;
3, 4, 5 : VAR_001 = 500;
END_CASE;
```

Оператор while

Синтаксис:

while {выражение} do {последовательность предложений} end_while

Действие: пока {выражение} истинно, выполняется {последовательность предложений}.

Пример

После выполнения следующего кода VAR_001=16:

```
VAR VAR_000 : INT := 10; END_VAR
VAR VAR_001 : INT; END_VAR
WHILE VAR_000 > 2 DO VAR_000 = VAR_000 - 1;
VAR_001 = VAR_001 + 2;
END_WHILE;
```

Оператор repeat

Синтаксис:

repeat {последовательность предложений} until {выражение} end_repeat

Действие: пока {выражение} истинно, выполняется {последовательность предложений}. Если {выражение} ложно, {последовательность предложений} выполняется 1 раз.

Пример

После выполнения следующего кода VAR_001=20:

```
VAR VAR_000 : INT :=10; END_VAR
VAR VAR_001 : INT; END_VAR
REPEAT VAR_001 = VAR_001 + 2; VAR_000 = VAR_000 + 1; UNTIL VAR_000 < 20
END_REPEAT;
```

Оператор for

Синтаксис:

for {инициализация переменной цикла} to {выражение1} by {выражение2} do {последовательность предложений}

end_for

Инициализация переменной цикла имеет вид:

{имя переменной}:={выражение}

Действие: пока значение переменной цикла меньше или равно значению {выражения1} выполняется {последовательность предложений}. По завершении каждого цикла к переменной цикла прибавляется значение {выражения2}; если оно не задано, прибавляется 1.

С помощью оператора **for** нельзя построить цикл с убывающим счетчиком. Для создания таких циклов нужно использовать операторы **while** и **repeat.**

Пример

После выполнения следующего кода VAR_001=22:

```
VAR VAR_000 : INT :=10; END_VAR
VAR VAR_001 : INT; END_VAR
FOR VAR_000 = 10 TO 20 DO VAR_001 = VAR_001 + 2; END_FOR;
```

Операторы break и exit

Операторы **break** и **exit** эквивалентны.

Синтаксис:

break

exit

Действие: выход за пределы цикла. В случае вложенных циклов выход осуществляется только из текущего цикла и не затрагивает внешние.

Оператор continue

Синтаксис:

continue

Действие: переход в конец цикла, т.е. выражения, следующие за оператором **continue** до конца цикла, не выполняются.

Операторы определения переменных

Операторы определения переменных могут быть заданы вручную (кроме операторов определения глобальных переменных) или с помощью табличного редактора.

В языке **ST** определены следующие операторы данного типа:

var
{определение переменной}
...
{определение переменной}
end_var

var_global
{определение переменной}
...
{определение переменной}
end_var

var_arg
{определение переменной}
...
{определение переменной}
end_var

var_input
{определение переменной}
...
{определение переменной}
end_var

var_output
{определение переменной}
...
{определение переменной}
end_var

var_inout
{определение переменной}
...
{определение переменной}
end_var

После ключевого слова **end_var** точка с запятой не ставится.

Действие: определяет новую переменную. При использовании совместно с **constant** задает константу.

Оператор **var ... end_var** используется для создания локальных переменных и структур; может использоваться в основной программе или ее компоненте (функции).

Оператор **var_global ... end_var** используется для создания глобальных переменных; может использоваться вне основной программы и ее компонентов (функций).

Оператор **var_arg(var_input) ... end_var** используется для определения аргументов (основной программы или ее функций), передаваемых по значению. Определение аргумента с помощью этого оператора равнозначно заданию аргумента типа **вход** в табличном редакторе.

Оператор **var_output(var_inout) ... end_var** используется для определения аргументов (основной программы или ее функций), передаваемых по ссылке. Определение аргумента с помощью оператора **var_output...end_var** равнозначно заданию в табличном редакторе аргумента типа **выход**, а определение аргумента с помощью оператора **var_inout...end_var** равнозначно заданию аргумента типа **вход/выход**.

Создание аргументов вручную с помощью указанных операторов может использоваться только в отладочных программах – для таких аргументов нельзя задать привязку. Аргументы рабочей программы следует создавать с помощью табличного редактора.

Выражение **{определение переменной}** имеет вид:

{имя переменной}: {тип переменной};
{имя переменной}: {тип переменной}:={выражение};
{имя переменной}: array [] of {тип переменной};
{имя переменной}: array [{размерности массива}] of {тип переменной};
{имя переменной}: array [{размерности массива}] of {тип переменной}:={начальные значения};

Выражения {размерности массива} задаются в виде диапазонов изменения индексов массива, разделенных запятой.

Диапазон изменения индексов массива имеет вид

{нижняя граница} .. {верхняя граница}

или

{размер массива}

обозначающий диапазон от **0** до **{размер массива}-1**. В случае, если размерность массива не указана, он считается пустым и ожидается его инициализация в ходе выполнения программы.

Выражения {начальные значения} имеют вид списка начальных значений элементов массива, разделенных запятой. Каждое начальное значение имеет вид выражения, вычисление которого дает реальное начальное значение, или конструкции

{целочисленная константа} ({выражение})

где {целочисленная константа} задает количество элементов, которым присваивается это значение. При присвоении начальных значений элементам массива первым изменяется последний индекс массива.

Область действия имени переменной определяется по следующим правилам:

- **глобальные** переменные действуют в рамках программы и сохраняют свое значение между вызовами программы. В частности, глобальными являются переменные FBD- и LD-блоков;
- **локальные** переменные и **аргументы** действуют в рамках объекта (программы, функции, структуры), в котором определены.

При привязке переменной по имени она ищется в следующем порядке: локальные, аргументы

функции, переменные-члены структуры, глобальные.

Оператор индексирования элементов массива

Синтаксис:

{имя} [{индекс 1}, ... {индекс N}]

где **{имя}** – имя переменной или функции, возвращающей массив, а **{индекс k}** – целое неотрицательное число или целочисленная переменная (кроме **UINT** и **USINT**). Количество индексов зависит от размерности массива.

Действие: возвращает ссылку на элемент массива, которая может быть использована в левой и правой части оператора присваивания.

Оператор goto

Синтаксис:

goto {метка строки}

Действие: безусловный переход к строке кода с указанной меткой. Оператор **goto** и метка, на которую этот оператор ссылается, должны находиться в одном и том же программном компоненте (программе, функции и т.п.). Метка должна начинаться с буквы и отделяться от кода программы двоеточием:

```
…
goto myLabel2;
…
myLabel2:
END_PROGRAM
```

Числовые константы

Десятичные целочисленные константы состоят из ненулевой цифры, за которой следует последовательность десятичных цифр:

123, 456, 7890

Двоичные целочисленные константы начинаются с префикса **2**#, за которым следуют цифры 0 или 1:

2#1001, 2#1100

Восьмеричные целочисленные константы начинаются с префикса **8**#, за которым следуют цифры от 0 до 7:

8#777, 8#0123

Шестнадцатеричные константы начинаются с префикса **16**#, за которым следуют цифры или буквы **a...f**. Буквы можно задавать как в нижнем, так и в верхнем регистре (**A...F):**

16#123, 16#EA7

Вещественные константы состоят из целой и дробной части, разделенной точкой. Либо целая, либо дробная часть может отсутствовать. Числа могут задаваться в формате с плавающей точкой, при этом они сопровождаются суффиксом E с указанием десятичного порядка:

1.23, 123., .123, 0.123E3, .123e-3, 123.E+5

Временные интервалы состоят из префикса **t#** или **time**#, за которым следует запись в виде

<дни>d<часы>h<минуты>m<секунды>s<миллисекунды>ms

Любая составляющая может быть опущена (например, запись **t#1h10s** является корректной и означает 1 час 10 секунд). Временной интервал приводится к целочисленному виду, означающему количество миллисекунд в заданном временном интервале.

Дата состоит из префикса **d#** или **date**#, за которым следует запись в виде **yyyy-mm-dd** (год, месяц, день). Приводится к целочисленному виду, означающему количество секунд, прошедшее с 0 часов 1 января 1971 года до 0 часов заданной даты.

Время дня состоит из префикса **tod#** или **time_of_day**#, за которым следует запись в виде **hh:mm:ss** (час, минута, секунда). Приводится к целочисленному виду, означающему количество секунд, прошедшее с 0 часов текущего дня.

Константа "Дата и время" состоит из префикса **dt#** или **date_and_time**#, за которым следует запись в виде **yyyy-mm-dd-hh:mm:ss** (год, месяц, день, час, минута, секунда). Приводится к целочисленному виду, означающему количество секунд, прошедшие с 0 часов 1 января 1971 года до заданных даты и времени.

Строковые константы

Строковые константы представляют собой набор символов, заключенных в одинарные или двойные кавычки: **'первая строка', "вторая строка"**. В строке недопустимы управляющие символы, включая переводы строки, а также кавычки и символ $.

Для размещения в строках произвольных символов применяется механизм эскейп-последовательностей, начинающихся с символа $. Определены следующие последовательности:

$r – возврат каретки, код 16#0D;

$n – перевод строки, код 16#0A;

$t – табуляция, код 16#09;

$uXXXX – UNICODE-символ ('X' – шестнадцатеричная цифра);

$x – символ **x** ('x' – любой символ).

Пример

"Строка с кавычкой: **$'**, символом **$u0410** и переводом строки **$n**"

Особенности вычислений

Целочисленность результата арифметических вычислений в программе имеет высший приоритет – даже в том случае, когда этот результат присваивается переменной с плавающей точкой.

Пусть, например, в программе объявлена переменная **float**:

```
VAR VAR_000 : REAL; END_VAR
```

Тогда:

```
VAR_000 = 2 / 10          //VAR_000 = 0
VAR_000 = 2. / 10         //VAR_000 = 0.2
VAR_000 = 2. / 10  + 2 /10   //VAR_000 = 0.2
```

Контрольные вопросы

1. Охарактеризуйте язык списка инструкций IL.
2. В чем назначение модификаторов в языке IL?
3. Назовите три способа для вызова функциональных блоков в языке IL.
4. Для чего служат модификаторы в языке IL?
5. Что такое оператор в языке IL?
6. С помощью какой команды производят прямое объявление адресов в языке IL?

Лабораторная работа №5 Разработка графического интерфейса интегрированных систем управления

ЛАБ 5
Разработка графического интерфейса

Графическое представление хода выполнения техпроцесса, а также управление техпроцессом с помощью графических средств являются одними из главных задач, решаемых TRACE MODE 6. Для разработки графического интерфейса оператора в интегрированную среду встроены **редактор представления данных (РПД)** и его модификация – **еРПД** (рис. 20):

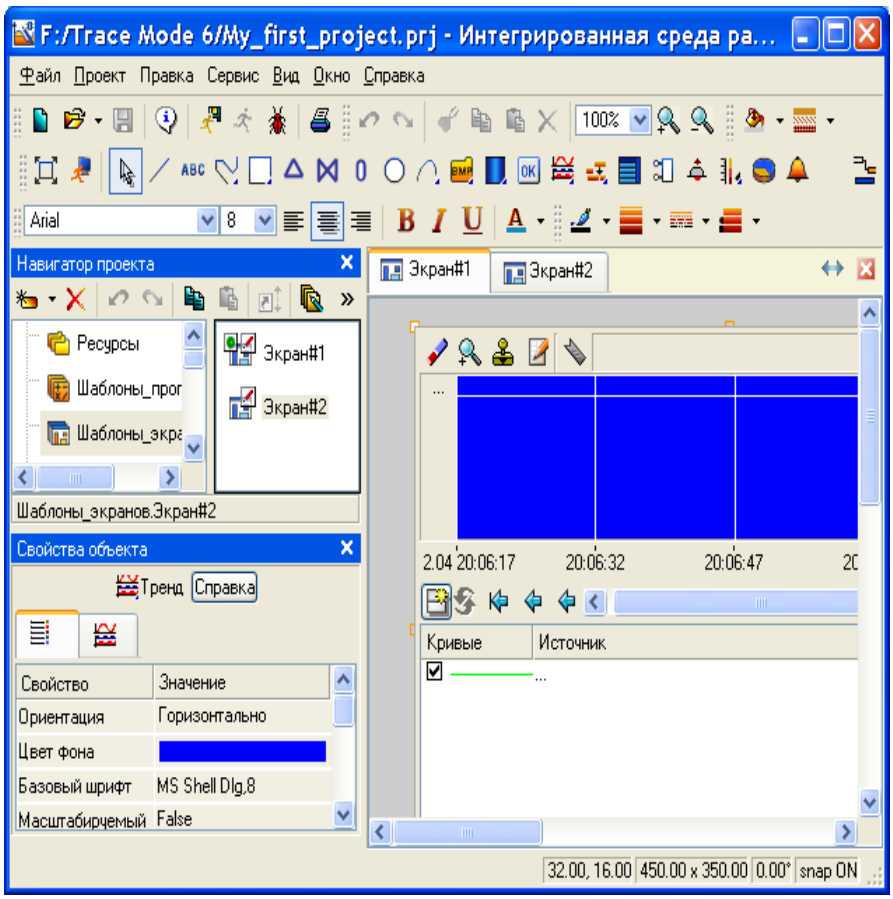


Рис. 20. Редактор представления данных

Отрезки, примыкающие к блоку справа, обозначают выходы блока (возвращаемые функцией значения).

Графический интерфейс оператора разрабатывается в **РПД** в виде набора **графических экранов** или в **еРПД** в виде набора **графических панелей** , являющихся компонентами проекта.

Разработка псевдографического интерфейса оператора для узлов, которые исполняются мониторами в среде DOS, описана в разделе **Мнемосхемы**.

Для создания шаблона экрана нужно выполнить команду **Экран**, для создания шаблона панели – команду **Графическая панель** из контекстного меню слоя **Шаблоны экранов**.

С целью взаимодействия с другими компонентами проекта для экрана/панели могут быть заданы аргументы.

Совокупность графических экранов/панелей узла образует его **графическую базу**. Совокупность графических баз всех узлов разрабатываемого проекта АСУТП образует **графическую часть** проекта.

Графический экран/панель может содержать один или несколько графических **слоев**, каждый из которых, в свою очередь, может содержать один или несколько **подслоев**.

В слоях графического экрана/панели размещаются **графические элементы** (ниже соответственно **ГЭ** и **еГЭ**). Графические элементы имеют наборы настраиваемых **атрибутов, динамических свойств** и **функций управления**. Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени. **РПД** и **еРПД** содержат большое количество встроенных графических элементов, позволяющих изобразить практически любой техпроцесс, вывести на дисплей всю необходимую информацию о ходе его выполнения, а также управлять техпроцессом.

Главное меню и панели инструментов РПД

Панель инструментов 'Графические элементы'

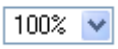
С помощью инструментов этой панели выбираются графические элементы для размещения их в графических слоях экранов. При выборе ГЭ редактор переходит в режим размещения.

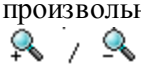
С помощью кнопки  данной панели можно перейти в режим редактирования, с помощью кнопки  – в режим эмуляции.

Кнопка  предназначена для переключения режима отображения графических экранов (обычный/полноэкранный).

Меню и панель инструментов 'Правка'

Меню и панель инструментов **Правка** содержат ряд типовых инструментов для редактирования графических экранов. Данные инструменты доступны также из контекстного меню ГЭ. Контекстное меню ГЭ содержит также команду **Копировать в избранное**.

В списке 

(**Масштаб**) можно выбрать предустановленный масштаб или вручную задать произвольный. Для выбора предустановленного масштаба можно также использовать кнопки  или сочетания клавиш **CTRL+ПЛЮС/МИНУС** на цифровой клавиатуре. При выделении некоторой области **AxB** экрана с помощью мыши с удержанием клавиши **Z** экран масштабируется в **MIN{C/A, D/B}** раз, где **CxD** – размеры видимой области. Во всех случаях масштабирование производится относительно центра видимой области.

Меню 'Сервис' и панель инструментов 'Топология экрана'

Данные панель инструментов и меню содержат команды для позиционирования и тиражирования выделенного графического элемента.

Меню **Сервис** содержит дополнительно команду **Параметры экрана.**

Панель инструментов 'Параметры текста'

В режиме редактирования с помощью типовых инструментов данной панели задаются параметры текста в выделенном графическом элементе (выделенной группе ГЭ). Данные команды применимы только к такому тексту, который может быть введен/отредактирован с помощью клавиатуры.

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры линии'

В режиме редактирования с помощью инструментов этой панели задаются параметры линии (линии контура) выделенного графического элемента (выделенной группы ГЭ):

 – выбор **цвета линии**. По этой команде на экран выводится стандартный диалог выбора цвета;

 – выбор **толщины линии.**

 – выбор **стиля линии**. По этой команде открывается список стилей, содержащий в том числе опцию **Без линии** (при выборе этой опции линия невидима).

 – выбор **края линии** (**плоский, квадратный, круглый**). Ниже на рисунках показан один и тот же графический элемент **Текст** с различными краями линии контура:

- **плоский;**
- **квадратный;**
- **круглый;**

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры заливки'

В режиме редактирования с помощью инструментов этой панели задаются параметры заливки выделенного графического элемента (выделенной группы ГЭ):

 – выбор **цвета заливки**. По этой команде на экран выводится стандартный диалог выбора цвета;

 – выбор **стиля заливки.**

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Ресурсные библиотеки'

Инструменты данной панели предназначены для операций с библиотеками строк, рисунков и других ресурсов, которые могут быть использованы при разработке графических экранов.

Меню 'Вид'

Команды этого меню управляют видимостью редактора аргументов экрана, окна **Слои**, таблицы графических элементов и окна **Избранное**, а также панелей инструментов **Топология экрана** и **Параметры текста.**

Задание типовых свойств графических элементов

В этом разделе описаны типовые свойства графических элементов и инструменты их задания.

Специфические свойства отдельных ГЭ

рассматриваются при описании этих ГЭ.

Графические элементы имеют следующие настраиваемые свойства:

- **Атрибуты**
- **Динамические свойства**
- **Функции управления**

Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени:

Для задания свойств ГЭ (группы ГЭ) используется окно **Свойства объекта**, содержащее различное число вкладок для разных элементов (рис. 21).

Чтобы открыть это окно, нужно выполнить команду **Свойства** из контекстного меню выделенного ГЭ (выделенной группы ГЭ) или дважды нажать ЛК мыши на ГЭ. При снятии выделения ГЭ окно его свойств автоматически закрывается. Окно **Свойства объекта** недоступно, если графический элемент расположен в слое, редактирование которого запрещено.

Атрибуты – это простейшие свойства графического элемента. Они задаются на вкладке (**Основные свойства**) окна **Свойства объекта.**

В окне свойств атрибуты могут быть сгруппированы – наименования таких групп выделены подчеркиванием, при двойном нажатии на них ЛК раскрывается список свойств.

Существуют 2 вида атрибутов ГЭ:

- **статические** – атрибуты, которые не изменяются при работе в реальном времени;
- **динамизируемые** – атрибуты, которые могут быть как статическими, так и динамическими (изменяющимися при работе в реальном времени в зависимости от значения привязанного аргумента). Разделы конфигурирования таких атрибутов выделены красным цветом и содержат пункт **Вид индикации.**

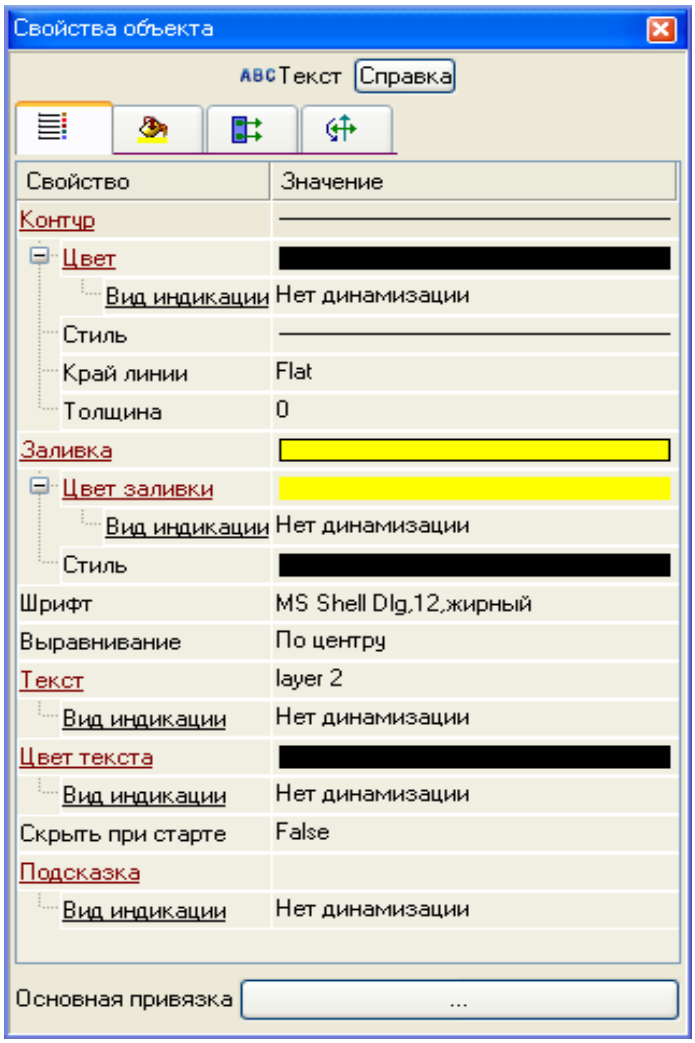


Рис. 21. Задание свойств группы ГЭ

Статические атрибуты графических элементов

В данном разделе описано задание типовых статических атрибутов ГЭ с помощью вкладок окна **Свойства объекта.**

Некоторые типовые статические атрибуты ГЭ могут быть также заданы с помощью панелей инструментов РПД.

Видимость при старте

При переходе в режим реального времени (т.е. при запуске проекта в МРВ или при переходе в режим эмуляции) графические элементы по умолчанию видимы. Чтобы скрыть ГЭ при запуске режима реального времени, нужно установить для него атрибут **Скрыть при старте.**

В режиме редактирования видимы все ГЭ.

В реальном времени управлять видимостью ГЭ можно с помощью функции управления ГЭ и с помощью ГЭ **Свободные формы.**

Всплывающая подсказка

В разделе **Подсказка** для ГЭ задается всплывающая подсказка, отображаемая на экране при наведении курсора на ГЭ в режиме реального времени.

Цветовые атрибуты

Разделы конфигурирования цветовых атрибутов ГЭ (**Цвет, Цвет линии, Цвет заливки, Цвет текста** и т.п.) в окне **Свойства объекта** содержат типовой инструмент задания цвета – кнопку

При нажатии этой кнопки на экран выводится диалог выбора предустановленных цветов (рис. 22).

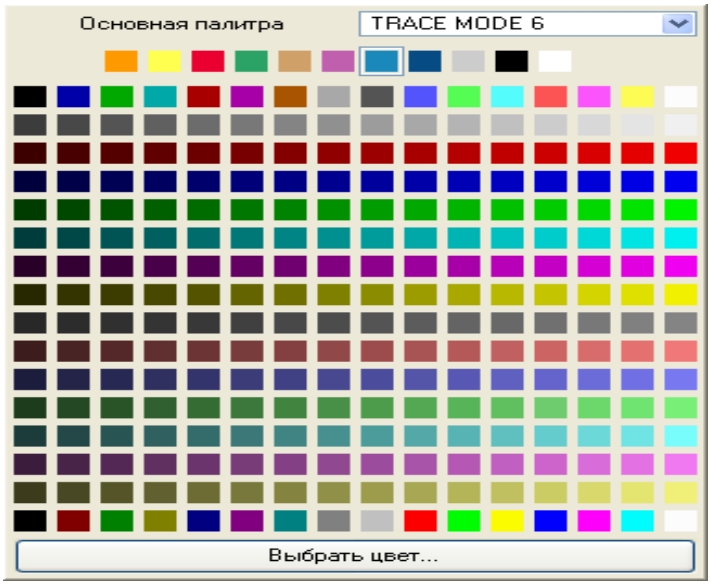


Рис. 22. Диалог выбора цветов

Раскрывающийся список меню **Основная палитра** позволяет выбрать различные стили с предустановленным набором базовых цветов.

При нажатии кнопки **Выбор цвета** этого диалога на экран выводится стандартный диалог выбора цвета (рис. 23).

Выбранный цвет выводится на кнопку соответствующего раздела:

Системный цвет

Подобный атрибут может быть корневым в окне свойств ГЭ или располагаться в разделе конфигурирования некоторого цветового атрибута

Если значение атрибута – TRUE, то в первом случае он задает использование соответствующих системных цветов Windows для всех
цветовых атрибутов ГЭ, во втором – только для атрибута, в разделе которого находится.

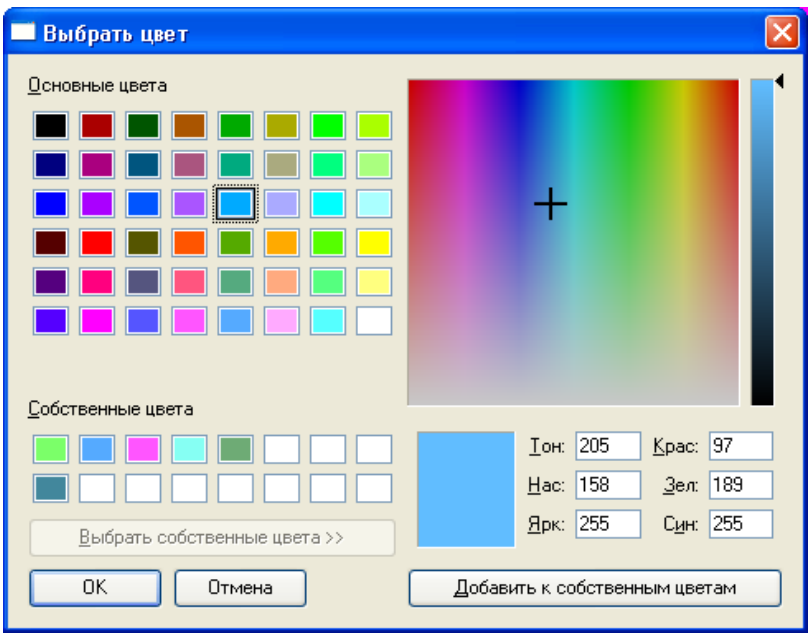


Рис. 23. Задание параметров цвета

Если значение атрибута – FALSE, используются цвета, заданные в окне свойств ГЭ.

Толщина линии

Раздел конфигурирования атрибута **Толщина** содержит инструмент выбора толщины линии и окно отображения численного значения толщины (в пикселях).

Стиль линии

Раздел конфигурирования атрибута **Стиль линии** содержит кнопку  .

При нажатии этой кнопки на экране появляется меню выбора линии контура.

Выбранный вид линии выводится на кнопку раздела.

Стиль заливки

Раздел конфигурирования атрибута **Стиль заливки** содержит кнопку  .

При нажатии этой кнопки на экране появляется меню выбора стиля заливки.

Выбранный стиль выводится на кнопку раздела.

Прозрачность заливки

При значении TRUE (значение по умолчанию) этот атрибут задает прозрачность фона ГЭ при одном из линейных стилей заливки. Если атрибут имеет значение FALSE, цвет фона ГЭ – белый.

При точечных стилях заливки этот атрибут не работает (фон ГЭ – белый).

Тип заливки

Раздел конфигурирования заливки ГЭ содержит атрибут **Тип заливки**, для которого может быть установлено значение **Цвет** (значение по умолчанию) или **Изображение.**

При значении **Цвет** параметры заливки определяются ее цветовыми атрибутами.

При значении **Изображение** в окне свойств отображается атрибут **Изображение**. При нажатии кнопки в разделе его конфигурирования на экране появляется диалог выбора картинки из ресурсной библиотеки изображений (рис. 24).

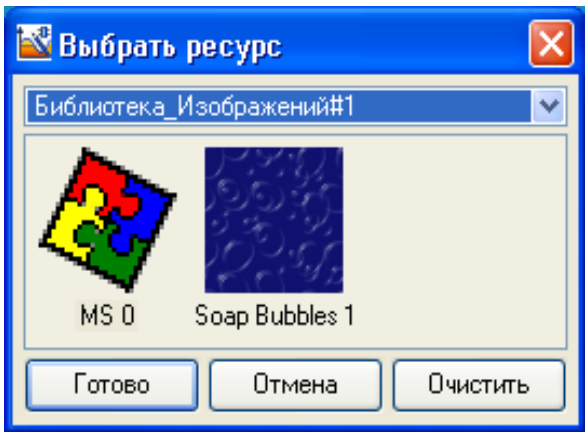


Рис. 24. Диалог выбора изображения

Выбранная картинка отображается в разделе конфигурирования атрибута **Изображение** и устанавливается в качестве фона ГЭ.

Текстовые атрибуты

Раздел конфигурирования текстовых атрибутов (**Текст**, **Надпись** и т.п.) содержит окно обычного текстового редактора для задания значения атрибута (для некоторых ГЭ подобному атрибуту по умолчанию присваивается некоторое значение).

Шрифт

Раздел конфигурирования атрибута **Шрифт** содержит кнопку, при нажатии которой на экран выводится меню выбора параметров шрифта.

По команде **Размеры** этого меню на экране появляется меню выбора размера шрифта, по команде **Шрифт** – стандартный диалог задания параметров шрифта (рис. 25).

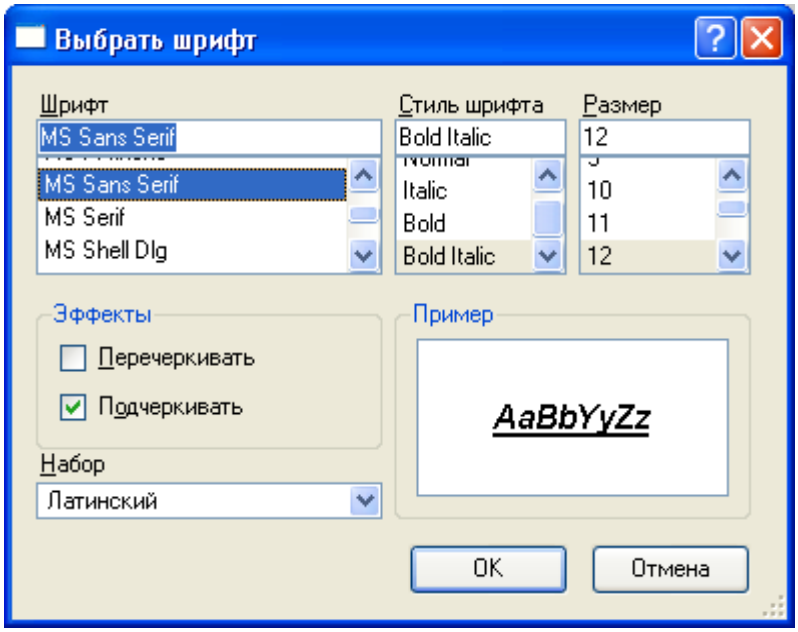


Рис. 25. Диалог задания параметров шрифта

Выбранные параметры шрифта отображаются на кнопке раздела.

По умолчанию задан шрифт **MS Shell Dlg** с размером 8.

Выравнивание

Раздел конфигурирования атрибута **Выравнивание** содержит стандартные инструменты выравнивания текста: **Влево**, **По центру** и **Вправо.**

Прозрачность

Атрибут **Прозрачность** определяет степень прозрачности ГЭ. Этот параметр задается в процентах (0-100), 0 соответствует абсолютной непрозрачности.

Выделение в МРВ

Атрибут **Выделение в МРВ** разрешает/запрещает выделение графического элемента в реальном времени по нажатию ЛК.

Выделение работает только в том случае, если для ГЭ задана функция управления.

Если ГЭ входит в состав графического объекта (для объекта должно быть разрешено выделение), то по нажатию ЛК на ГЭ выделяется объект.

Для оконных ГЭ эта функция отключена.

Глобальное разрешение выделения графических элементов в реальном времени и параметры выделения (цвет, отступы, толщина и стиль контура) задаются в файле **gr_settings.xml** (тег **<last_clicked_attrs**>), расположенном в директории МРВ. Параметры выделения могут быть также заданы при конфигурировании РПД.

Временные атрибуты

Дата и время должны задаваться в форматах, заданных в региональных настройках ОС. Для перехода к заданию времени нужно нажать ЛК в соответствующем поле.

При нажатии кнопки 

открывается меню, содержащее команды **Сброс** (задать нулевое время, т.е. 01.01.1970) и **Текущее** (задать текущее время).

Динамизация атрибута графических элементов

Динамизацией атрибута называется задание условий его изменения в зависимости от значения привязанного аргумента. При динамизации атрибута графический элемент становится индикатором выполнения заданных условий.

При размещении ГЭ на экране все его динамизируемые атрибуты по умолчанию статические, и разделы их конфигурирования на вкладке **Основные свойства** окна свойств содержат инструмент задания соответствующего статического параметра. Например, при размещении ГЭ **Текст** раздел динамизируемого атрибута **Цвет текста** содержит инструмент выбора цвета .

Чтобы динамизировать атрибут, нужно дважды нажать на названии ЛК мыши, и в раскрывшемся списке настроить динамические свойства с помощью раздела **Вид индикации** (рис. 26).

Вид условия (и, соответственно, вид индикатора, создаваемого из ГЭ), выбирается в разделе **Вид индикации**:

Значение – индикация значения аргумента;

Arg = Конст. – индикация равенства аргумента заданной константе;

Arg >= Конст. – индикация превышения аргументом заданного порога;

Arg & Конст. – индикация состояния битов значения аргумента, заданных маской **Константа**. Если хотя бы один такой бит установлен, индицируется ИСТИНА, иначе – ЛОЖЬ;

Arg в диапазоне – индикация нахождения аргумента в заданных диапазонах;

Arg в интервале – индикация нахождения аргумента в интервалах привязанного канала.

В зависимости от выбранного вида индикации меняются инструменты его конфигурирования.

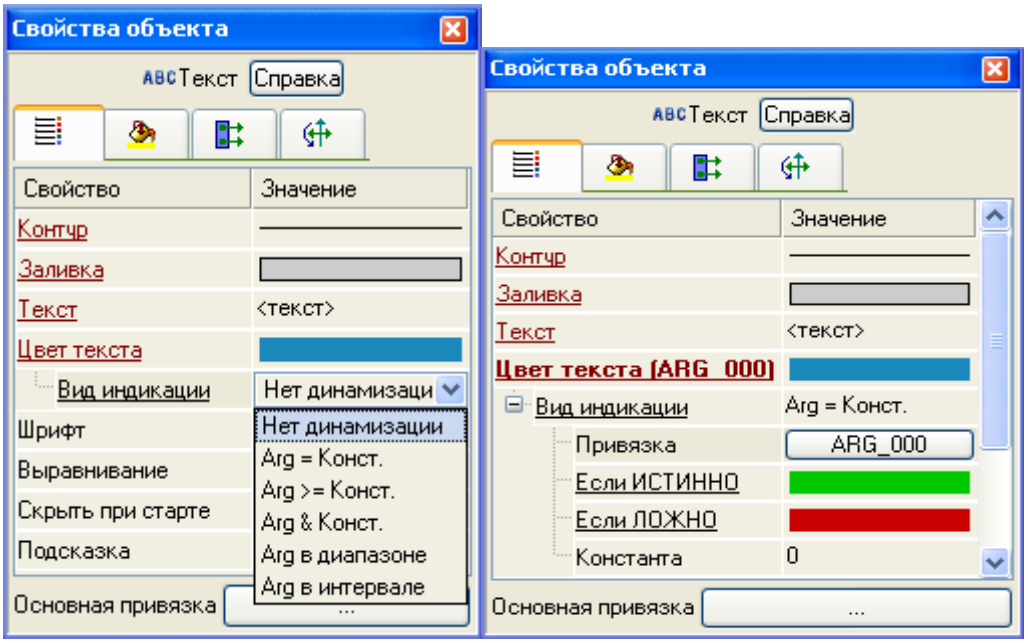


Рис. 26. Динамизация атрибутов

Индикация значения

Вид индикации **Значение** может быть задан при динамизации атрибута **Текст**. Формат вывода выбирается в списке **Формат** (рис. 27).

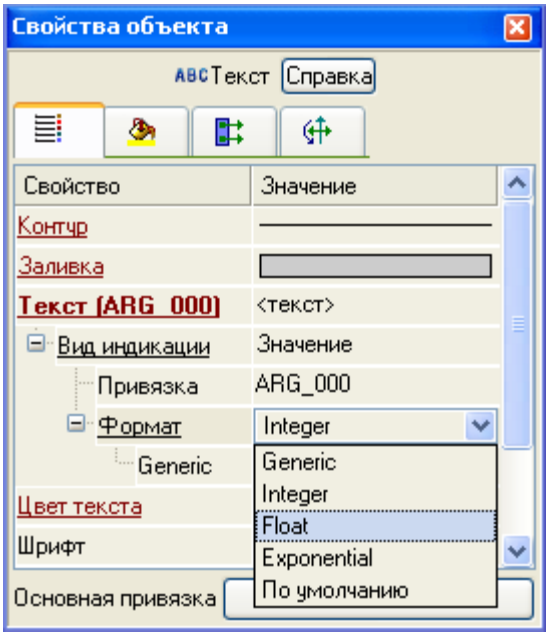


Рис. 27. Динамизация атрибута Текст

Для каждого формата можно задать более точное описание в нотации языка Си.

При выборе формата **По умолчанию** ГЭ будет отображать значение аргумента в формате, заданном МРВ.

Индикация интервала для значений каналов FLOAT и DOUBLE FLOAT

Вид индикации **Arg в интервале** может быть задан при динамизации цветовых атрибутов (**Цвет текста**, **Цвет заливки** и др.). Он служит для визуального отображения интервала, в котором находится значение канала класса **FLOAT** или **DOUBLE FLOAT.**

Для выбора аргумента служит атрибут **Привязка**, при нажатии на значение которого открывается стандартное окно выбора аргумента.

Для правильной работы данного вида индикации выбираемый аргумент должен быть связан с атрибутом **Интервал** (7, **P**) канала класса **FLOAT** или **DOUBLE FLOAT.**

При нахождении значения привязанного канала в пределах предупредительных границ (интервал 0), динамизируемый атрибут принимает цвет, установленный по умолчанию.

Атрибут **Предупреждение** служит для выбора цвета при нахождении значения канала за пределами предупредительных, но в пределах аварийных границ (интервал 1 или 2).

Атрибут **Авария** служит для выбора цвета при нахождении значения канала за пределами аварийных границ, но в пределах границ шкалы (интервал 3 или 4).

Атрибут **Вне границ** служит для выбора цвета при нахождении значения канала за пределами границ шкалы (интервал 5 или 6).

Другие виды индикации

Наборы инструментов конфигурирования других видов индикации зависят от динамизируемого атрибута. Если параметр индикации при динамизации атрибута может быть задан как вручную, так и выбран из ресурсной библиотеки, набор инструментов содержит переключатель **Использовать ресурсы** (FALSE – вручную, TRUE – из библиотеки).

При конфигурировании видов индикации **Arg = Конст.**, **Arg >= Конст.** и **Arg & Конст.** в полях **Если ИСТИННО** и **Если ЛОЖНО** задаются значения, которые должен принимать динамизируемый атрибут при выполнении заданного условия (**ИСТИННО**) и в противном случае (**ЛОЖНО**). Если задано **Доп. значение для ИСТИННО** (**ЛОЖНО**), то на каждом такте обновления графического экрана значение атрибута, заданное в поле **Если ИСТИННО** (**ЛОЖНО**), сменяется значением, заданным в соответствующем поле **Доп. значение**…. Чтобы отобразить дополнительное значение, надо выполнить соответствующую команду из контекстного меню поля **Если ИСТИННО** (**ЛОЖНО**). Значение константы, с которой сравнивается аргумент, задается в поле **Константа.**

При конфигурировании вида индикации **Arg в диапазоне** строки описания диапазонов создаются/удаляются с помощью контекстного меню, вызываемого нажатием ПК мыши на атрибуте **Диапазоны** – для создания нового диапазона, и на созданном атрибуте **Диапазон** – для его удаления. В

полях **Мин**. и **Макс**. задаются границы диапазонов (**Макс** должно быть больше **Мин**). Индикатор этого вида работает по следующему алгоритму: при изменении значения привязанного аргумента ищется первый по списку диапазон, которому удовлетворяет аргумент (**Мин <= arg < Макс**) и атрибуту присваивается значение, заданное в соответствующем поле **Значение**. Создание и назначение поля **Доп. значение** – такое же, как при конфигурировании видов индикации **Arg = Конст.** и **Arg >= Конст**..

Если в поле **Использовать ресурсы** установлено значение **True**, поля **Если ИСТИННО (ЛОЖНО)** и **Доп. значение…** содержат кнопки, при нажатии которых открываются навигаторы соответствующих библиотек для выбора ресурса (рис. 28).

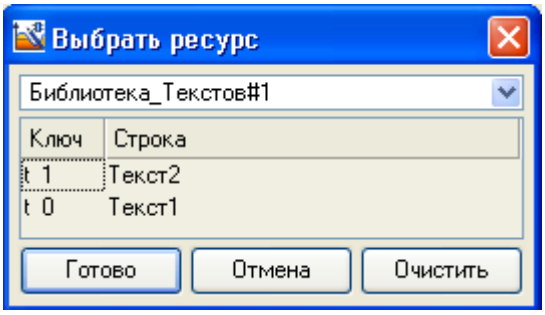


Рис. 28. Навигатор выбора ресурса

Для типовых атрибутов поля **Если ИСТИННО (ЛОЖНО)** и **Доп. Значение…** содержат кнопки, при нажатии которых открываются стандартные диалоги задания параметра (например, цвета).

Основная привязка

Окно свойств некоторых ГЭ содержит раздел **Основная привязка**.

При нажатии на кнопку выводится стандартный диалог выбора аргумента.

При динамизации атрибутов они, как правило, автоматически привязываются к основной привязке (если она задана), однако для ряда ГЭ основная привязка имеет более широкое назначение.

Пример создания индикатора значения

В данном примере показана динамизация атрибутов **Текст** и **Цвет текста** графического элемента **Текст**, в результате которой ГЭ становится индикатором текущего значения генератора, и при превышении заданного порогового значения меняет цвет текста.

Привязать атрибут ГЭ к генератору (или каналу) напрямую нельзя. Чтобы использовать генератор для динамизации атрибута, нужно этот атрибут привязать к аргументу, который, в свою очередь, привязать к генератору (или каналу).

Создадим в слое **Источники/Приемники** группу **Генераторы** с генератором **Пила**.

В слое **Система** создадим группу **RTM**, и в ней канал **Экран**. В результате создастся канал **Экран#1** класса **Вызов** со ссылкой на **Экран#1** в слое **Шаблоны экранов**.

В табличном редакторе аргументов создадим для экрана **Экран#1** аргумент **ARG_000** (чтобы открыть табличный редактор аргументов, нужно выполнить команду **Аргументы** из меню **Вид)** (рис. 29).

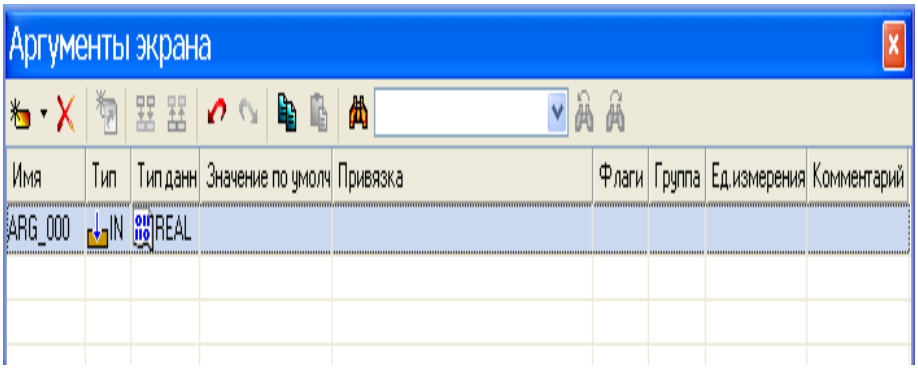


Рис. 29. Создание аргументов экрана

Дважды нажмем ЛК в поле **Связь** табличного редактора аргументов и выберем для привязки генератор **Пила** (рис. 30)..

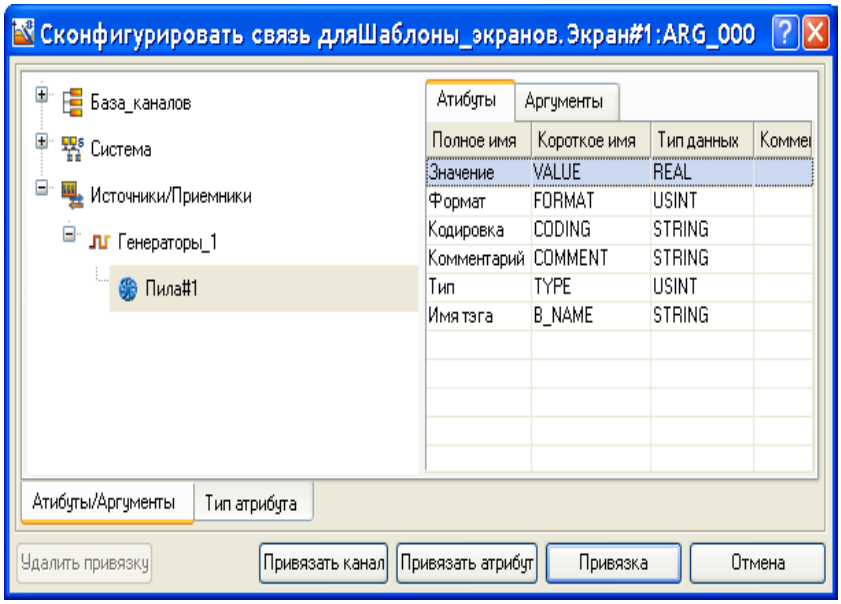


Рис. 30. Создание привязки аргумента

Заданная привязка отобразится в табличном редакторе аргументов.

Разместим на экране ГЭ **Текст** и зададим его статические атрибуты, как показано на рисунке (рис. 31).

Чтобы при работе в реальном времени ГЭ отображал текущее значение аргумента, нужно выполнить следующие действия:

– дважды нажать ЛК мыши на поле **Текст** в окне свойств – по этой команде раскрывается список с параметрами индикации атрибута. В поле **Тип индикации** выбрать **Значение**.

– нажать кнопку в поле **Привязка** и задать аргумент.

При нажатии кнопки ОК диалог конфигурирования динамического свойства закрывается, а привязка к аргументу отображается в окне свойств ГЭ.

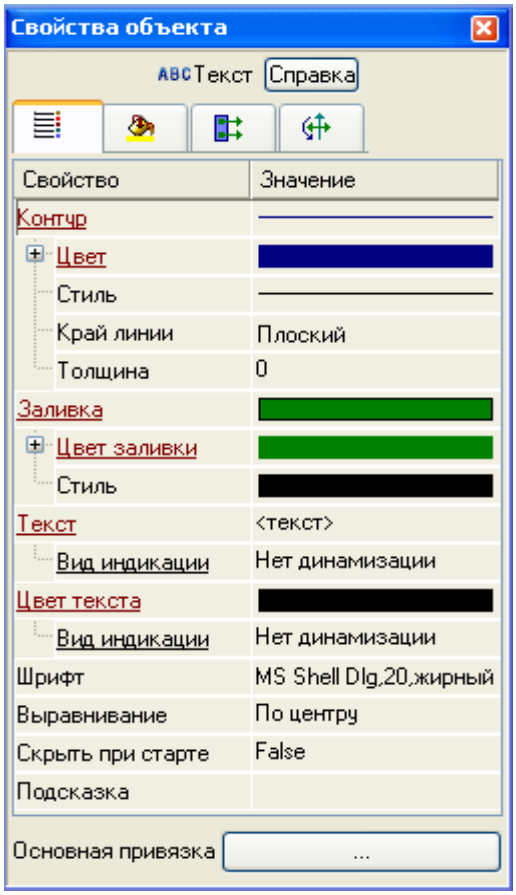


Рис. 31. Задание статических атрибутов текста

Теперь настроим изменение цвета текста ГЭ при превышении заданного порогового значения. Для этого нужно выполнить следующие действия:

– дважды нажать ЛК мыши на поле **Цвет Текста** в окне свойств – по этой команде раскрывается список с параметрами индикации атрибута. В поле **Тип индикации** выбрать **Arg >= Конст.**;

– нажать кнопку в поле **Привязка** и задать аргумент появившемся диалоге.

В поле **Константа** ввести пороговое значение 80, при превышении которой мы зададим изменение цвета текста ГЭ.

В поле **Значение для TRUE** при помощи стандартного диалога выбора цвета зададим цвет текста для значений, превышающих пороговое (в данном случае красный). В поле **Значение для FALSE** зададим цвет текста для значений меньше порогового (в данном случай черный).

Теперь сохраним проект для запуска в мониторе реального времени, откроем и запустим его в профайлере с поддержкой графических экранов – ГЭ отобразит изменяющуюся амплитуду пилообразного сигнала. При превышении заданного порогового значения 80 цвет текста изменится с черного на красный.

Динамические свойства

Динамическая заливка ГЭ

При использовании данного свойства ГЭ отображает значение привязанного аргумента числового формата в виде закрашенной области (такая область далее называется **слоем**). Поддерживаются два вида динамической заливки – **однослойная** (отображает значение одного аргумента) и **многослойная** (отображает значения нескольких аргументов). Оба вида настраиваются на вкладке **Динамическая заливка** ( ) окна **Свойства объекта**. Для использования динамической заливки нужно на этой вкладке установить флаг **Разрешено**.

Для добавления/удаления слоя используется контекстное меню, вызываемое нажатием ПК мыши на названиях пунктов **Слои**/**Слой** соответственно. Настройки для всех создаваемых слоев имеют одинаковое назначение.

Вкладка содержит следующие инструменты конфигурирования заливки.

Направление – направление заливки (**вверх**, **вниз**, **вправо**, **влево**).

Имя – имя слоя. Для перехода к редактированию нужно дважды нажать ЛК в этом поле.

Привязка – выбор аргумента, к которому привязывается слой. При нажатии ЛК в данном поле на экране появляется диалог выбора аргумента . Слой, для которого привязка к аргументу не задана, считается привязанным к 0. Если заливка многослойная, значения привязываемых аргументов должны быть неотрицательными.

Тип заливки – выбор типа заливки. На вкладке доступны типовые атрибуты конфигурирования заливки выбранного типа.

Мин, Макс – числа, которые ставятся в соответствие границам ГЭ, используемым в качестве пределов шкалы. Например, если выбрано направление заливки справа налево, **Мин** соответствует правой, а **Макс** – левой границе графического элемента. Если флаг **Мин = LL, Макс = HL** не установлен, значения **Мин** и **Макс** могут быть заданы вручную.

Мин = LL, Макс = HL – если выбранный аргумент привязан к значению канала, то при установке этому атрибуту значения TRUE пределы шкалы устанавливаются равными соответственно нижнему и верхнему пределам канала. Если заливка многослойная, то пределы шкалы установятся равными соответственно **LL** и **HL** первой по порядку привязки.

Цвета для диапазонов – если выбранный аргумент привязан к значению канала, то при установке этому атрибуту значения TRUE с помощью цветовых атрибутов **Предупреждение**, **Авария**, **Вне границ** можно задать дополнительные цвета заливки, соответствующие нахождению значения канала в определенном диапазоне - при многослойной заливке значение атрибута **Цвета для диапазонов** должно быть FALSE.

В зависимости от значений **Мин** и **Макс** возможны 2 варианта индикатора, создаваемого из ГЭ.

Вариант 1: **Мин**=0, **Макс** больше или равно максимально возможной сумме аргументов. В этом случае ГЭ отображает абсолютный вклад аргументов в их общую сумму (ниже показано применение многоуровневой заливки к ГЭ **Прямоугольник**, каждый из аргументов изменяется от 0 до 100, текущие значения аргументов отображают графические элементы **Текст**).

Вариант 2: **MAX** > **MIN** > 0.

Этот вариант предназначен для решения специальных задач отображения. Примером такой задачи может служить отображение в заданном диапазоне уровней несмешивающихся жидкостей в емкости, если в аргументы передаются толщины слоев жидкостей (обратите внимание на инверсный порядок слоев в списке относительно их расположения в емкости). Диапазон отображаемых уровней задается параметрами **МИН** и **МАКС**.

Динамическое перемещение ГЭ

Задание траектории перемещения

Задание режима перемещения

Это свойство настраивается в разделе **Перемещать** вкладки **Динамическая трансформация** () окна **Свойства объекта**.

Чтобы использовать данное динамическое свойство, надо установить флаг **Перемещать**.

При работе в реальном времени графический элемент перемещается вдоль траектории, которая задается как ломаная линия (количество узлов ломаной не ограничено). Текущее положение ГЭ зависит от значения привязанного аргумента (числовой аргумент для привязки выбирается в списке **Привязка**), от значений, заданных для узлов траектории, и флага **Перемещать плавно**.

Задание траектории перемещения

Под заданием траектории понимается задание положения ее узлов и задание значений для этих узлов.

По умолчанию траектория динамического перемещения представляет собой отрезок от точки привязки ГЭ до центра ограничивающего прямоугольника, т.е. имеет 2 узла. Значения для этих узлов устанавливаются равными 0 и 100 соответственно (значение, заданное для узла, отображается в окне **Значение узла** при наведении курсора на узел).

С помощью метода **drag-and-drop** положение узлов траектории на экране можно изменять (при наведении на узел курсор принимает вид  , для выделения узла нужно нажать на нем ЛК).

Чтобы добавить новый узел, нужно выделить один из имеющихся узлов и далее использовать метод **drag-and-drop** при нажатой клавише **CTRL** (при этом для узла-потомка устанавливается значение, которое задано для узла-родителя).

Узел, первоначально размещенный в точке привязки ГЭ (этот узел обозначается красной точкой), остается крайним узлом при любых манипуляциях с траекторией и в дальнейшем называется первым узлом.

Значения для крайних узлов траектории задаются вручную. Для этого нужно выделить крайний узел, ввести число в окне **Значение узла** и нажать кнопку **Установить для узла.**

Значения для промежуточных узлов траектории могут быть заданы вручную или рассчитаны автоматически.

Чтобы задать значения для промежуточных узлов вручную, надо установить флаг **Использовать значения промежуточных узлов** и далее задавать значение для каждого промежуточного узла аналогично заданию значения для крайнего узла.

Значения для узлов должны монотонно возрастать (убывать) от одного крайнего узла до другого.

Чтобы задать значения для промежуточных узлов автоматически, надо сбросить флаг **Использовать значения промежуточных узлов** или при установленном флаге **Использовать значения промежуточных узлов** нажать кнопку **Рассчитать значения узлов** – в обоих случаях значения для промежуточных узлов рассчитываются исходя из значений, заданных для крайних узлов, и общей длины траектории.

Задание режима перемещения

Если флаг **Перемещать плавно** не установлен, при работе в реальном времени графический элемент скачкообразно перемещается от узла к узлу, располагаясь в каждый момент времени на том узле, для которого задано значение, ближайшее к текущему значению привязанного аргумента.

Если флаг **Перемещать плавно** установлен, автоматически рассчитывается значение для каждого пикселя траектории, при этом расчет зависит от флага **Использовать значения промежуточных узлов**:

Если флаг **Использовать значения промежуточных узлов** не установлен, значение для пикселей рассчитывается исходя из значений, заданных для крайних узлов и общей длины траектории.

Если флаг **Использовать значения промежуточных узлов** установлен, значение для пикселей рассчитывается на каждом отрезке траектории исходя из его длины и значений, заданных для его узлов.

При установленном флаге **Перемещать плавно** положение графического элемента привязано к пикселю, значение которого имеет наименьшее отклонение от текущего значения привязанного аргумента.

Динамический контур ГЭ

Динамический контур представляет собой прокручиваемый по часовой стрелке пунктир (под прокруткой здесь подразумевается дискретное перемещение с шагом, равным длине штриха). Это свойство настраивается на вкладке **Динамический контур** окна **Свойства объекта.**

На вкладке размещены следующие инструменты настройки динамического контура:

Привязка – задание привязки к аргументу экрана (аргумент должен иметь числовой формат). От значения привязанного аргумента зависит скорость прокрутки контура. Если аргумент равен 1, контур перемещается на 1 шаг на каждом такте обновления экрана; если аргумент равен 2, контур перемещается на 1 шаг 1 раз за 2 такта, и т.д. Если аргумент равен 0, контур не прокручивается.

Цвет штриха – выбор цвета штриха.

Цвет промежутка – выбор цвета промежутка.

Длина штриха– задание длины штриха (и, соответственно, шага перемещения контура) в пикселях (2-100).

Промежуток/штрих – задание отношения длины промежутка к длине штриха (1-10).

Функции управления графическими элементами

Функции управления ГЭ – это действия, заданные для графических элементов на этапе редактирования проекта АСУ; выполнение этих действий при работе в реальном времени инициализируется оператором с помощью мыши. Задание функций управления для графических элементов придает графическим экранам свойство интерактивности и обеспечивает одно из важнейших качеств АСУ – управление техпроцессом с помощью графических средств.

Функции управления задаются на вкладке **Действия** () окна **Свойства объекта**:

Определены следующие **события**, по которым инициализируется выполнение действий в реальном времени:

- **pressed** (**нажатие** ЛК на ГЭ);
- **released** (**отжатие** ЛК на ГЭ);

Для каждого из событий может быть независимо задано несколько функций управления, выбираемых из контекстного меню (меню открывается при нажатии ПК мыши на названии события):

- **передать значение** (**Send Value**);
- **показать/скрыть элементы** (**Show/Hide Elements**);
- **перейти на экран** (**Jump to Screen**);

- **послать комментарий** (**Send Comment**);
- **послать подсказку** (**Send ToolTip**);
- **послать строку** (**Send String**);
- **выполнить** (**Execute**).

Функции управления отображаются в виде новых разделов списка свойств объекта (для каждой функции создается отдельный раздел). Для удаления функции управления или изменения ее позиции в списке используется контекстное меню, вызываемое нажатием ПК мыши на названии функции. Если для события задано несколько функций, в реальном времени они отрабатываются по порядку в соответствии с позицией в списке (функция перехода на экран всегда выполняется последней).

Для каждого события можно задать подтверждение и звуковой сигнал. Для этого используются атрибуты **Подтверждение** и **Сигнал.**

Чтобы подтвердить действие нужно нажать клавишу **ОК**. Для отмены – **Cancel.**

Если установлен **Сигнал**, то при совершении указанного действия система воспроизводит **Стандартный звук**, заданный в Windows.

Код доступа – код доступа к использованию функций управления (0-255). Права на доступ к функциям управления задаются для пользователя в виде маски в разделе **Доступ** / **Формы** канала **Пользователь**. При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к функциям управления разрешен, в противном случае – запрещен. Код доступа к использованию функций управления отображается в таблице графических элементов.

Если пользователи в системе не заданы, значение кода доступа не учитывается.

Если ни один бит маски канала **Пользователь** не выделен, доступ к функциям управления разрешен только при значении кода доступа 0.

При наведении на ГЭ с функцией управления курсор принимает вид  . Для ГЭ с функцией управления может быть также сконфигурировано выделение в реальном времени.

Встроенные графические элементы

'Линия'

ГЭ **Линия**  не имеет специфических свойств и размещается в графическом слое стандартным способом.

'Текст'

ГЭ **Текст**  не имеет специфических свойств и размещается в графическом слое стандартным способом.

При динамизации атрибуты автоматически привязываются к основной привязке, если она задана.

При задании формата вывода можно добавить поясняющий текст, который отображается в реальном времени.

'Плоские фигуры'

В эту группу входят следующие ГЭ:

Плоский клапан 

Треугольник 

Овал 

Стрелка 

Эллипс, сектор 

'Ресурсы'

В эту группу входят следующие ГЭ:

Текстовый ресурс 

Растровое изображение 

Векторное изображение 

Видеоклип 

Рисунок из файла 

Текст из файла  .

Эти ГЭ представляют собой невидимые окна, в которых по заданным правилам отображаются ресурсы из соответствующих или файлов.

'Видеоклип'

ГЭ **Видеоклип** размещается на графическом экране стандартным способом; ресурс из библиотеки выбирается в навигаторе, который появляется на экране при выборе элемента на панели инструментов **Графические элементы.**

Выбранный в ходе размещения ГЭ ресурс устанавливается в качестве статического значения атрибута **Видеоклип**.

При нажатии ЛК в поле значения этого атрибута в окне свойств на экране появляется следующий диалог, с помощью котрого можно задать другое статическое значение атрибута **Видеоклип**, в том числе пустое (по команде **Очистить**).

Помимо атрибута **Видеоклип**, на вкладке **Осн. свойства** конфигурируются следующие специфические атрибуты данного ГЭ:

- **Привязка** – предназначен для задания условия воспроизведения видеоклипа по значению привязанного аргумента. Если значение аргумента не равно 0, видеоклип воспроизводится, если 0 – не воспроизводится;
- **Пауза** – определяет паузу между кадрами при воспроизведении клипа. Этот параметр задается в тактах обновления экрана (значение статического параметра по умолчанию – 0);
- **Показывать при остановке** – если этот атрибут имеет значение **True** (значение по умолчанию), после остановки воспроизведения ГЭ отображает первый кадр клипа; если **False**, после остановки воспроизведения клип становится невидимым;
- **Непрерывное воспроизведение** – если этот атрибут имеет значение **True** (значение по умолчанию), клип воспроизводится циклически; если **False**, клип воспроизводится однократно.

'Объемные фигуры'

В эту группу входят следующие ГЭ:

Цилиндр 

Сфера 

Конус 

Тор 

Пирамида 

Емкость 

Клапан 

Труба 

Рельефный конус 

Криволинейный конус 

Градиент 

'Кнопки'

В эту группу входят следующие ГЭ

Кнопка 

Группа кнопок 

Картинка-кнопка 

'Кнопка'

ГЭ **Кнопка**  размещается в графическом слое стандартным способом.

Для создания в шаблоне экрана **N** кнопки с функцией перехода на экран **M** достаточно перетащить шаблон или канал вызова экрана **M** из навигатора проекта в шаблон экрана **N.**

ГЭ **Кнопка** имеет следующие специфические настройки, задаваемые в окне свойств:

- **два состояния** – если этот флаг установлен, устойчивыми являются оба состояния кнопки (нажатое и отжатое), в противном случае устойчиво только отжатое состояние;
- **привязка** – определяет аргумент для управления состоянием кнопки (нажатое и отжатое) при установленном атрибуте **Два состояния**. При значении аргумента 0 кнопка отжата, при любом другом значении – нажата.
- **плоская** – при установке этого флага кнопка становится плоской;
- **изображение** – кнопка выбора рисунка из ресурсной библиотеки изображений;
- **истинный размер изображения** – при установке этого флага картинка принимает свой истинный размер; если флаг не установлен, рисунок сжимается до размера 24*24 пикселя.

'Графики'

В эту группу входят следующие ГЭ:

Тренд

Тренд XY
‘Тренд’

ГЭ **Тренд** размещается в графическом слое стандартным способом.

ГЭ **Тренд** предназначен для отображения изменения значения аргументов экрана во времени, а также для отображения данных SIAD, данных индивидуальных архивов и исторических данных, полученных от серверов OPC HDA. Аргументы с типом данных с плавающей запятой отображаются на **аналоговой панели** в верхней части тренда. Аргументы с целочисленным типом данных на **дискретной панели,** которая располагается под аналоговой (рис. 32).

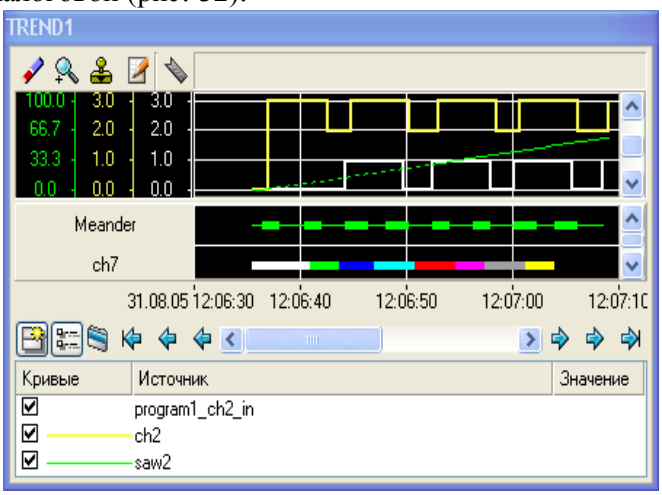


Рис. 32. ГЭ Тренд

Помимо типовых свойств тренд имеет значительное количество специфических атрибутов.

Окно свойств ГЭ содержит вкладки **Осн. свойства** и **Кривые**.

На вкладке **Осн. свойства** конфигурируются следующие специфические атрибуты:

Ориентация – расположение временной шкалы тренда (горизонтальное или вертикальное).

Масштабируемый – если TRUE, размеры тренда можно изменять в реальном времени.

Заголовок – текст, введенный в это поле, отображается в заголовке окна тренда.

Сетка – этот раздел содержит типовые инструменты задания параметров сетки тренда.

Легенда – этот раздел содержит типовые инструменты задания параметров легенды, отображаемой в нижней части тренда. Если на тренде 2 панели аналоговая и дискретная, легенда переключается между ними (для переключения нужно нажать ЛК на панели). В легенду выводится заголовок кривой (имя аргумента, если для кривой не задан атрибут **Заголовок**); цвет и значение в точке, указанной визиром. Чтобы в реальном времени показать/скрыть кривую, надо нажатием ЛК установить/снять соответствующий флаг в столбце **Кривые** легенды.

Ось времени – этот раздел содержит настройки временной оси:

Показывать – если TRUE, на оси отображаются значения времени.

Разбиение – количество разбиений видимой части оси.

Период подписи – период подписей на оси (в разбиениях). Линии сетки без подписи имеют цвет, заданный атрибутом **Доп. цвет** в разделе **Сетка.**

Диапазон – диапазон значений видимой части оси (0-100).

Единицы – единицы измерения диапазона. Выбираются из меню: **секунда, минута, час, день.**

Левая граница – аргумент, в который передается текущее значение левой временной границы тренда.

Правая граница – аргумент, в который передается текущее значение правой временной границы тренда.

Аргументы, хранящие текущие временные границы тренда, могут быть использованы, в частности, для вывода мгновенного снимка тренда в генерируемый документ. При этом на тренд в документе выводятся только те кривые, которые отображают данные индивидуального архива, выборку OPC HDA, или значения канала, архивируемого в SIAD.

Ось значений – этот раздел содержит настройки оси значений.

Разбиение – количество разбиений видимой части оси.

Период подписи – период подписей на оси (в разбиениях). Линии сетки без подписи имеют цвет, заданный атрибутом **Доп. цвет** в разделе **Сетка.**

Показывать – видимость осей значений.

Все оси – отображаются оси всех кривых.

Только активная – отображается ось кривой, выделенной в легенде.

Буфер – количество хранимых в памяти значений каждой кривой для вывода на тренд (24-10^6, по умолчанию – 500). Кроме того, этот параметр задает максимальное число значений, извлекаемых из архива и отображаемых на тренде при переходе к заданной временной метке.

Буфер можно сохранить в файл – для этого у канала, вызывающего шаблон экрана с трендом, нужно установить параметр **Дамп** как **Read/Write**.

Буфер сохраняется в файл с именем **<ID>.DRG**, где **ID** – идентификатор канала, вызывающего шаблон экрана с трендом. Период сохранения буфера определяется параметром **Период сохранения доп. информации** в настройках узла (вкладка **Отчет тревог / Дамп / Параметры).**

При запуске проекта на тренде будут показаны значения последнего сохраненного буфера.

Буфер тренда, входящего в состав объекта, нельзя сохранить в файл.

Масштаб дискрет (%) – этот атрибут задает высоту поля, отводимого на дискретной панели для отображения одной кривой. Высота задается в процентах (>=100%) относительно размера базового шрифта (100%, значение по умолчанию).

Цвета статусов – цвета для целочисленных аргументов при их равенстве соответственно 0...7, отображаемых на дискретной панели. Для использования этих цветов атрибуту **Интерпретировать как** кривой должно быть установлено значение **Статус.**

Чтобы получить на экране график, необходимо сконфигурировать хотя бы одну кривую на вкладке **Кривые**:

Для добавления кривой нужно выполнить команду **Добавить** из контекстного меню раздела **Кривые**, для удаления – команду **Удалить** из контекстного меню раздела **КриваяN.**

Набор свойств кривой зависит от атрибута **Интерпретировать как** (см. рисунок выше) – если для этого атрибута задано значение **Статус**, для кривой конфигурируются только привязка и заголовок.

Полный набор инструментов конфигурирования параметров кривой включает следующие инструменты:

Заголовок – название кривой, при задании выводится в легенду, в противном случае в легенде отображает имя канала (0, если отображаемый аргумент привязан к аргументу или не имеет привязки).

Привязка – привязка кривой к аргументу экрана. Для отображения значения канала (в том числе данных по этому каналу из SIAD) к аргументу должен быть привязан атрибут **0,R** этого канала; для отображения индивидуального архива – атрибут **1,A** соответствующего канала CALL .

Цвет – цвет кривой.

Стиль линии – стиль кривой.

Толщина линии – толщина кривой в пикселях.

Тип меток – выделение точки кривой с периодом изменения привязанного аргумента с помощью одной из меток.

Формат – формат чисел на оси значений;

Стиль при I<>0 и W=0 – стиль линии при аппаратной/программной недостоверности значения канала, привязанного к отображаемому аргументу. При нажатии ЛК в этом поле выводится стандартный список стилей.

Стиль при I=0 и W=1 – стиль для отображения значений канала при **(8) W** =1.

Стиль при I<>0 и W=1 – стиль при аппаратной/программной недостоверности значения канала и при **(8) W** =1.

Макс. значение – верхний предел оси значений.

Мин. значение – нижний предел оси значений.

Интерполирование – этот атрибут может принимать следующие значения:

- **нет** – нет интерполяции;
- **по периоду реального времени** – интерполирование методом проведения прямой между текущим значением канала и его значением на предыдущем такте пересчета;
- **простая линейная** – интерполирование методом проведения прямой между точками, соответствующими изменениям значения канала.

В реальном времени доступны следующие инструменты ГЭ **Тренд:**

– возврат к исходному масштабу тренда;

– увеличение. Кнопка имеет два положения – нажатое (увеличение включено) и отжатое (увеличение отключено). Для детального просмотра некоторого участка тренда нужно в режиме увеличения выделить его мышью, удерживая ЛК;

– включение/выключение режима перемещения визира по меткам активной (выбранной в легенде) кривой;

– вход в диалог редактирования кривых тренда, идентичный вкладке **Кривые** окна **Свойства объекта**. Окно предназначено для добавления/удаления кривых и изменения их настроек в реальном времени;

Если на экране непосредственно (не в окне) размещен графический объект, содержащий тренд, то данная функция тренда недоступна.

– переход к временной метке. При нажатии на эту кнопку появляется диалоговое окно ввода даты и времени. После нажатия кнопки ОК на тренд выводится информация, начиная с указанного времени;

– окно отображения даты и времени точки, указанной визиром. Визир (вертикальная линия) появляется в месте нажатия ЛК;

– показать панель инструментов. Если кнопка не нажата, панель инструментов не отображается;

– показать/скрыть легенду;

– индикатор синхронизации отображаемых данных с архивом;

–  и  – кнопки для перехода по временной оси графика. Соответственно **к началу**, **на день назад**, **на час назад** и **на начало следующего часа**, **на начало следующего дня**, **к концу**.

Для смещения по осям тренда можно также использовать стандартные инструменты прокрутки.

Чтобы в реальном времени изменить масштаб по любой из осей, надо нажать ЛК в области тренда, а затем нажатием сочетания клавиш **CTRL+** →/ ←/ ↓/ ↑установить требуемый масштаб.

'Диаграммы'

В эту группу входят следующие ГЭ:

Круговая диаграмма

Гистограмма

'Гистограмма'

ГЭ **Гистограмма** отображает значения привязанных аргументов экрана в виде столбцов.

ГЭ размещается в графическом слое стандартным способом.

Окно свойств гистограммы содержит вкладки **Осн. свойства** и **Столбцы**.

На вкладке **Осн. свойства** конфигурируются следующие специфические атрибуты гистограммы:

– в разделе **Заголовок** задаются типовые параметры верхнего колонтитула ГЭ;

– в разделе **Подпись** задаются типовые параметры нижнего колонтитула ГЭ;

– в разделе **Легенда** задаются типовые параметры легенды. В легенду выводятся названия и цвета столбцов и текущие значения соответствующих аргументов.

Число десятичных знаков значений, отображаемых в легенде, задается атрибутом **Дробная часть** этого раздела;

атрибут **Показывать описания** позволяет выводить в верхней части столбца текущее значение привязанного аргумента.

Контрольные вопросы

1. Приведите примеры графических интерфейсов
2. В чем преимущество графического интерфейса для систем управления
3. В чем недостатки графического интерфейса для систем управления

Лабораторная работа №6 Разработка автоматизированной системы управления

Процесс создания систем диспетчерского контроля и управления технологическим процессом можно разбить не следующие этапы:

а) детализация технических требований на создаваемую диспетчерскую систему контроля и управления;

б) разработка проектно – сметной документации в сокращенном или полном объеме;

в) сбор и изучение исходных данных;

г) составление полного перечня переменных;

д) комплектация системы;

е) разбиение объекта управления на технологические участки и последующая распределение переменных по участкам и группа;

ж) создание базы данных;

и) создание статических частей графических экранов интерфейса оператора;

к) заполнение графических экранов интерфейса оператора динамическими элементами;

л) составление схемы переходов между графическими экранами оператора;

м) составление алгоритмов управления (для всех возможных режимов работы объекта, в том числе аварийного);

н) генерация печатных документов;

п) верификация базы данных;

р) разработка эксплуатационной документации;

с) тестирование системы в автономном режиме (без УСО);

т) монтаж;

у) тестирование системы в рабочем режиме (с УСО);

ф) внедрение, в том числе пуск- наладка и обучение персонала.

Здесь рассмотрен полный процесс создания АСУ ТП вне зависимости от выбранного пакета SCADA для реализации системы. Пункты с **ж** по **м**, а также пункт **с** сильно зависят от выбранного для реализации SCADA пакета. В частности, некоторые из них могут отсутствовать ввиду выполнения некоторых из вышеперечисленных этапов возможностями конкретного пакета автоматически.

Особенно в данном списке следует отметить этап сбора и изучения исходных данных. Это очень ответственный этап, от качественного выполнения которого зависят как качество непосредственно проектируемой системы, так и срок выполнения работ по ее созданию. Эти два показателя в свою очередь весомо влияют на конкурентоспособность как самого проекта, так и организации – проектировщика.

Исходными данными здесь являются:

– функциональная схема КИПиА;

– разделы регламента (или рабочих инструкций) с максимально возможным по детализации описанием технологии;

– различного рода ведомости и спецификации средств КИПиА;

– перечень контролируемых и регулируемых параметров;

– документация и внешний вид существующих щитов КИП с вторичными приборами;

– разводка параметров по существующим вторичным приборам;

– фотографии, рисунки и чертежи основных технологических агрегатов, помогающие лучше нарисовать мнемосхемы;

– заполненные образцы отчетных документов различной периодичности.

Для SCADA-системы программного пакета TRACE MODE создание системы АСУ ТП при проектировании делится на следующие этапы:

- создание структуры проекта в навигаторе;

- конфигурирование или разработка структурных составляющих, например, разработка шаблонов графических экранов интерфейса оператора, разработка шаблонов программ, описание источников/приемников и т.д.;

- конфигурирование информационных потоков (каналов);
- выбор аппаратных средств АСУ (компьютеров, контроллеров и т.п.);
- создание узлов в слое **Система** и их конфигурирование;
- распределение каналов, созданных в различных слоях структуры, по узлам и конфигурирование интерфейсов взаимодействия компонентов в информационных потоках;
- сохранение проекта в единый файл для последующего редактирования (с помощью команды **Сохранить** или **Сохранить как**);
- экспорт узлов в наборы файлов для последующего запуска под управлением мониторов TRACE MODE (по команде **Сохранить для MPB**).

Все перечисленные этапы (исключая два заключительных) и входящие в их состав операции могут выполняться в произвольном порядке.

Проектирование систем автоматизации в TraceMode может быть осуществлено несколькими способами:

- проектирование «от шаблонов»;
- проектирование от технологии;
- проектирование от топологии;
- смешанное проектирование.

При проектировании автоматизированной системы управления (АСУ) «от шаблонов» сначала разрабатываются мнемосхемы HMI, а уже затем подбирает необходимое оборудование.

При этом способе создается информационная база проекта – создаются шаблоны экранов и программ, затем каналы по аргументам разрабатываемых шаблонов экранов и программ, дополняя основной подход методами автопостроения и связывания каналов в узлах проекта. Механизм автопостроения описан в приложении Г.

В данной работе представлен способ проектирования АСУ «от шаблонов».

Этапы разработки автоматизированной системы управления

Разработку автоматизированной системы управления (АСУ) будем проводить в следующем порядке:

а) создание проекта и сохранение его на диске;
б) создание пользовательской библиотеки компонентов;
в) разработка шаблонов экранов;
г) разработка шаблонов программ и их отладка;
д) создание узлов проекта и базы узлов;
е) создание архива и отчета тревог;
ж) разработка программ имитаторов и добавление их в проект;
и) отладка проекта.

Порядок разработки АСУ можно изменить. К примеру, можно сначала создать шаблоны программ, а потом - шаблоны экранов.

Также к любому из этапов можно вернуться. К примеру, сначала создать шаблоны экранов и программ для одного объекта, а затем экран и программу - для другого объекта.

Из вышеперечисленных этапов разработки АСУ необязательными являются этапы **б, ж.**

Разработку АСУ рассмотрим на примере проектирования АСУ ТП перекачки нефти в цехе подготовки и перекачки нефти ЦППН-8 Приобского месторождения ООО «РН-Юганскнефтегаз».

Настройка инструментальной среды

Перед созданием АСУ необходимо провести настройку инструментальной среды.

Для настройки инструментальной системы необходимо ЛК выбрать меню **Файл → Настройки ИС**, как показано на рис. 1.

В открывшемся окне **Настройки** выберем вкладку **Редактор аргументов**. В блоке **Функции** уберем галочку с параметра «Переименовывать аргументы по привязке». Данная настройка позволяет избавить от ошибок в названии каналов при автопостроении. Автопостроение подробно описано в приложении Г.

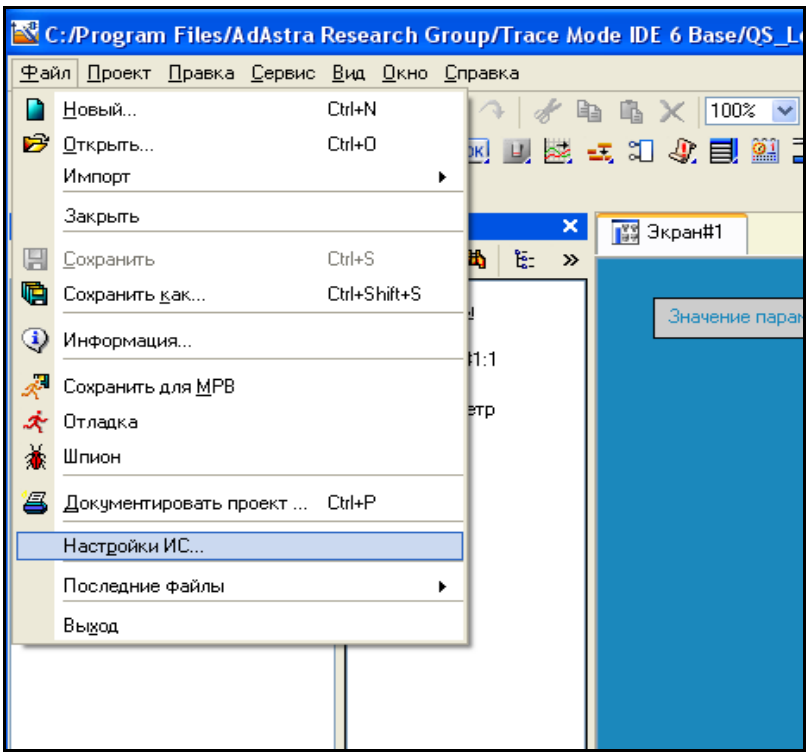


Рис. 1. Пункт **Файл** главного меню TraceMode

Также выберем пункт меню **РПД → Основные свойства**. В открывшемся окне установить флажок слева от элемента **Открывать свойства автоматически** и убрать флажок слева от пункта **Показать сетку**, как показано на рис. 2.

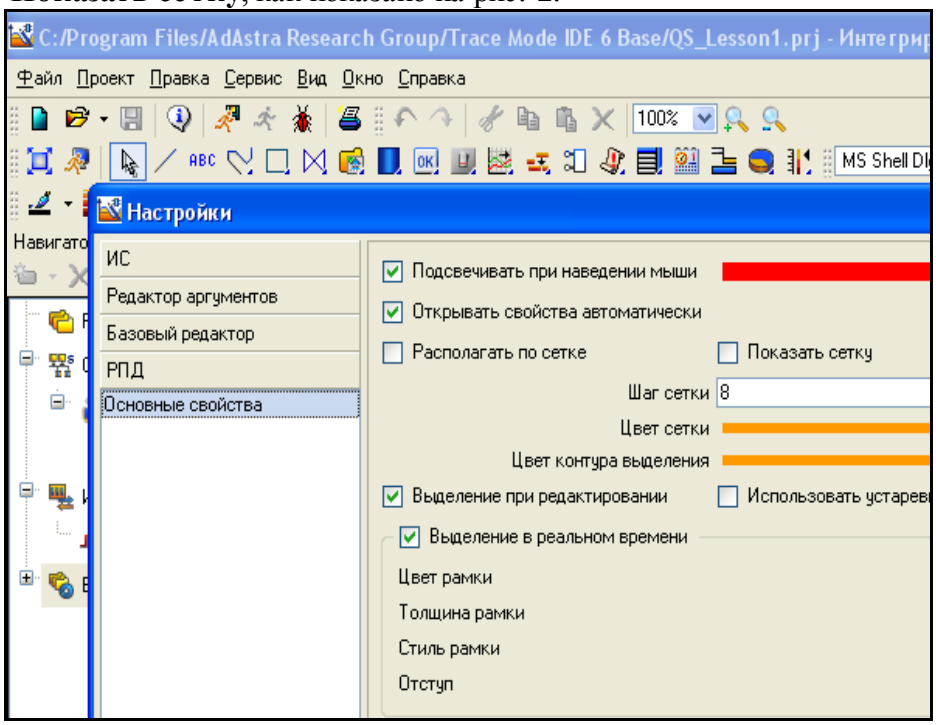


Рис. 2. Окно настройки ИС

Сетка упростит задачу более точного и согласованного размещения отдельных частей, из которых состоят графические элементы. Дальнейшее использование сетки при размещении графических элементов непосредственно на экранах оператора остается на усмотрение разработчика, так как в некоторых случаях ее использование может привести к смещению составных частей графических элементов относительно друг друга. Если вы решите применять её при создании отдельных графических элементов, то необходимо к строке меню выбрать ЛК **Файл**

→ **Настройки ИС**. В открывшемся окне Настройки ЛК выберем **РПД → Основные свойства**, в появившемся окне ЛК установить флажки слева от элемента **Показывать сетку** и **Располагать по сетке**.

Создание проекта

Откроем интегрированную систему разработки TRACE MODE. С помощью иконки  инструментальной панели создадим новый проект.

При этом в открывшемся на экране диалоге, показанном на рис. 3, необходимо выбрать уровень сложности.

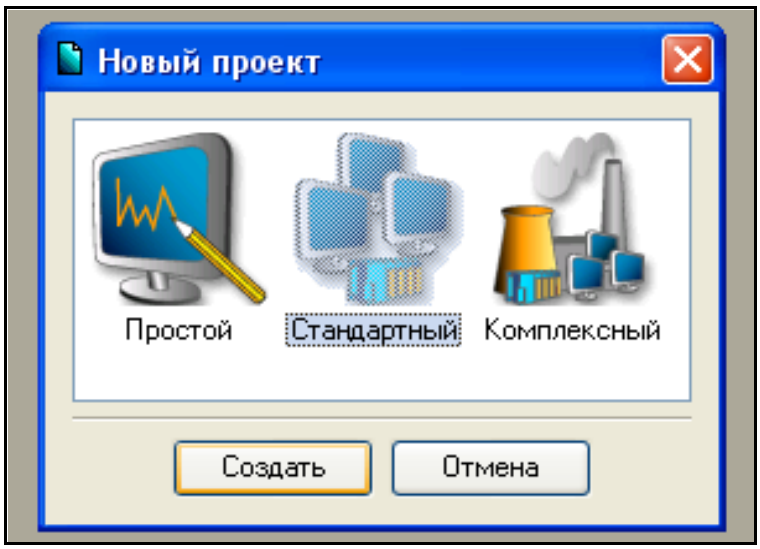


Рис. 3. Окно выбора типа проекта

В TRACE MODE существует 4 типа уровня сложности разработки проекта:

- простой;
- стандартный;
- комплексный;
- настраиваемый.

Уровни сложности отличаются слоями, доступными при разработке проекта.

При уровне сложности «Простой» доступны слои **Ресурсы**, **Система**, **Источники/приемники**, **Библиотека**. При уровне сложности «Стандартный», кроме слоев, доступных при уровне сложности «Простой», доступны также слои **Шаблоны программ**, **Шаблоны экранов**, **Шаблоны отчетов**, **Шаблоны связи с БД**. При уровне сложности «Комплексный», кроме слоев, доступных при уровне сложности «Стандартный», доступны также слои **Топология**, **Технология**, **КИПиА**. При уровне сложности «Настраиваемый» можно выбрать доступные слои. Также при данном уровне сложности можно сделать доступным слой **База каналов**.

При создании нового проекта предлагается выбрать один из 3 уровней сложности: простой, стандартный, комплексный. Настраиваемый уровень сложности можно включить в настройках инструментальной среды TRACE MODE через меню **Файл→Настройки ИС…** на вкладке **ИС→Уровень сложности**.

Уровень сложности разработки можно изменить в любой момент в пункте меню, указанном выше и показанном на рис. 4.

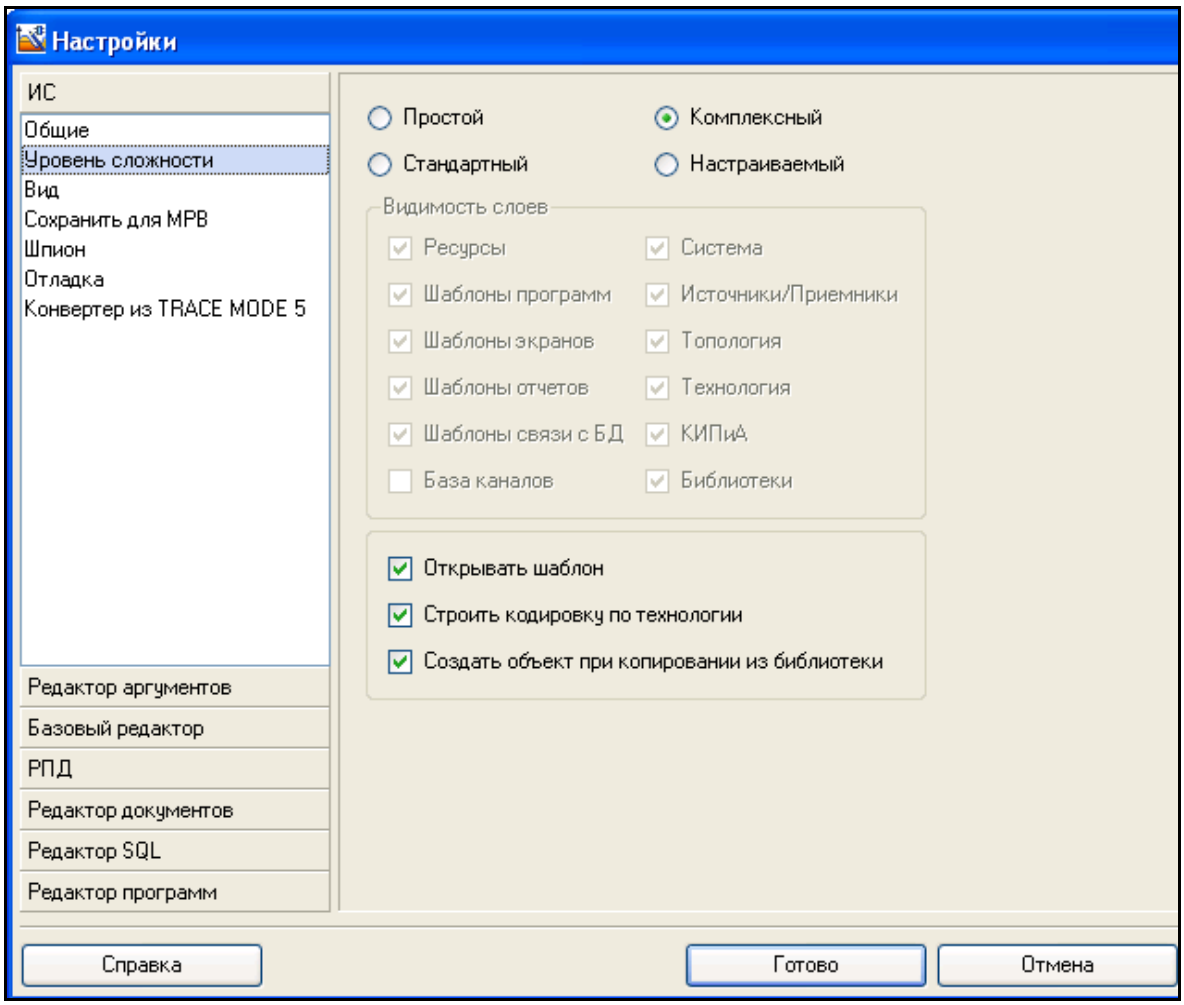


Рис. 4. Настройка уровня сложности из меню **Файл**

В данном случае выберем уровень сложности «**Стандартный**». После нажатия ЛК мыши на экранной кнопке **Создать** в левом окне **Навигатора проекта** появится дерево проекта.

После создания проекта сохраним его, нажав ЛК и указав необходимое имя (в данном примере «ASU.prj»).

Создание пользовательской библиотеки компонентов

В SCADA/HMI системе TRACE MODE 6 библиотеки подразделяются на два основных типа: ресурсные библиотеки и библиотеки компонентов [5].

Ресурсные библиотеки разбиты на коллекции графических примитивов различных форматов:

- анимация;
- векторная графика (метафайлы);
- растровая графика;
- графические элементы.

Последние представляют собой удобное средство для составления своих фирменных графических элементов (ГЭ) из совокупности стандартных ГЭ TRACE MODE 6. Их можно динамизировать, причем для каждого параметра задается индивидуальное имя. Конечно, группу ГЭ можно скопировать и не внося ее в ресурсную библиотеку, но эта возможность бывает очень полезна, если несколько ГЭ, образующих некий символ, привязаны к одному аргументу. В этом случае перепривязку одного элемента ресурсной библиотеки выполнить на порядок легче, чем несколько отдельных ГЭ, к тому же «спрятанных» в меню свойств этих элементов.

Библиотека компонентов представляет собой мощное средство для тиражирования готовых решений, и не только в области операторского интерфейса. Библиотеки компонентов

состоят из объектов, каждый из которых представляет собой проект TRACE MODE 6 в миниатюре, он может включать не только экраны, но и каналы, программы и прочие компоненты, между которыми сохраняются все связи.

Добавление базовых элементов

Воспользуемся пользовательской библиотекой компонентов. Для этого скопируем файл **tmdevenv.tmul** из поддиректории **%TRACE MODE%\Lib** в директорию **%TRACE MODE%** (под %TRACE MODE% понимается каталог, в который установлен TraceMode 6.05; по умолчанию это **C:\Program Files\AdAstra Research Group\Trace Mode IDE 6 Base**).

Перейдем в слой **Библиотеки_компонентов**, где в разделе **Пользовательская** откроем библиотеку **Library_1**. Сохраненный в данной библиотеке объект **Object_1** содержит в своем слое **Resources** необходимый для дальнейшей разработки набор графических объектов – изображения клапанов, емкостей, двигателей и т.д., показанный на рис. 5.

Перенесем нужные группы в слой **Ресурсы** текущего проекта с помощью механизма **drag-and-drop** и переименуем их, как показано на рис. 6. В данном случае были перенесены группы **Valves** и **Tanks**.

Здесь же в слое **Ресурсы** создадим группу **Картинки** для помещения в нее текстур, которые будут применены в оформлении создаваемых графических экранов, как показано на рис. 7.

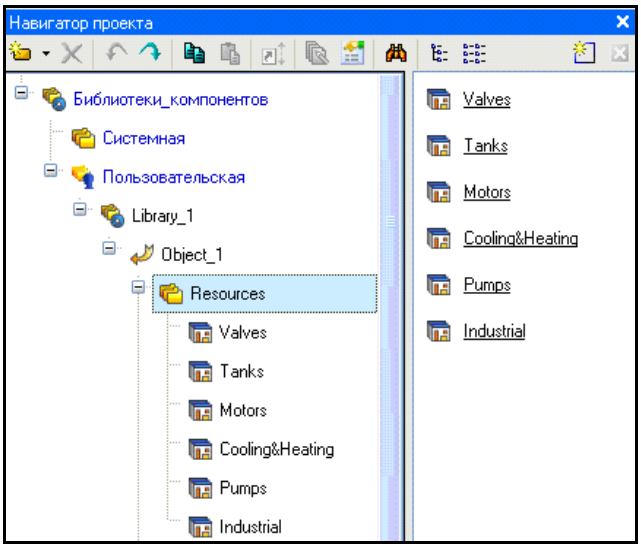


Рис. 5. Навигатор проекта

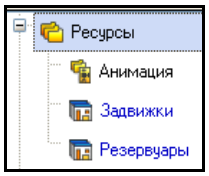


Рис. 6. Слой **Ресурсы** с добавленными группами

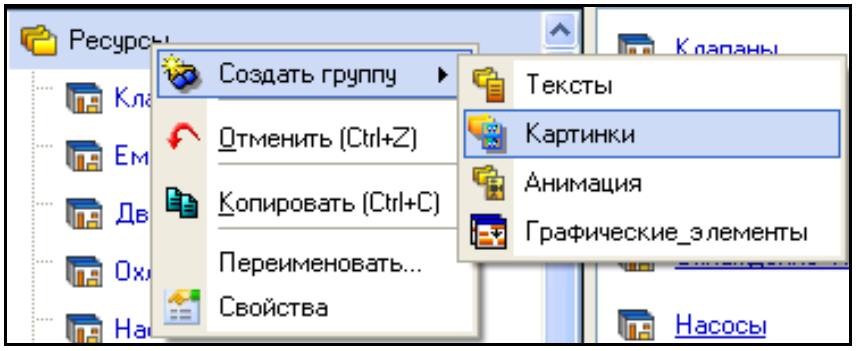


Рис. 7. Добавление группы **Картинки** в слой **Ресурсы**

Создадим в группе **Картинки** новый компонент – **Библиотека_Изображений#1,** как показано на рис. 8.

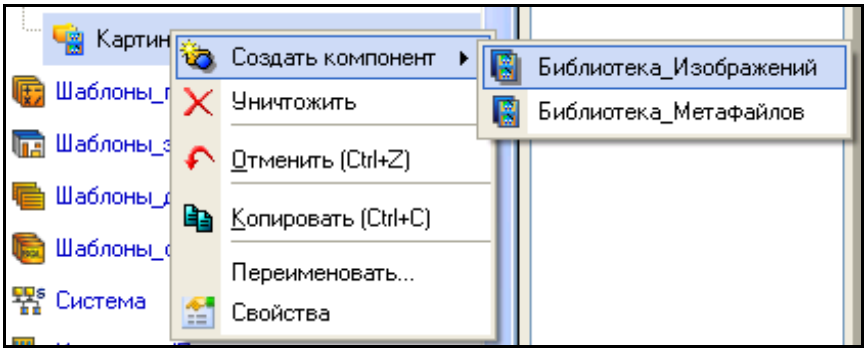


Рис. 8. Создание компонента **Библиотека_Изображений** в группе **Картинки**

Откроем двойным щелчком ЛК вновь созданную библиотеку для редактирования. Для ее наполнения воспользуемся иконкой на панели инструментов. В открывшемся диалоге выбора файлов для импорта укажем поддиректорию **…\Lib\Texture**. Выберем все файлы, как показано на рис. 9, и нажмем экранную кнопку **Открыть.**

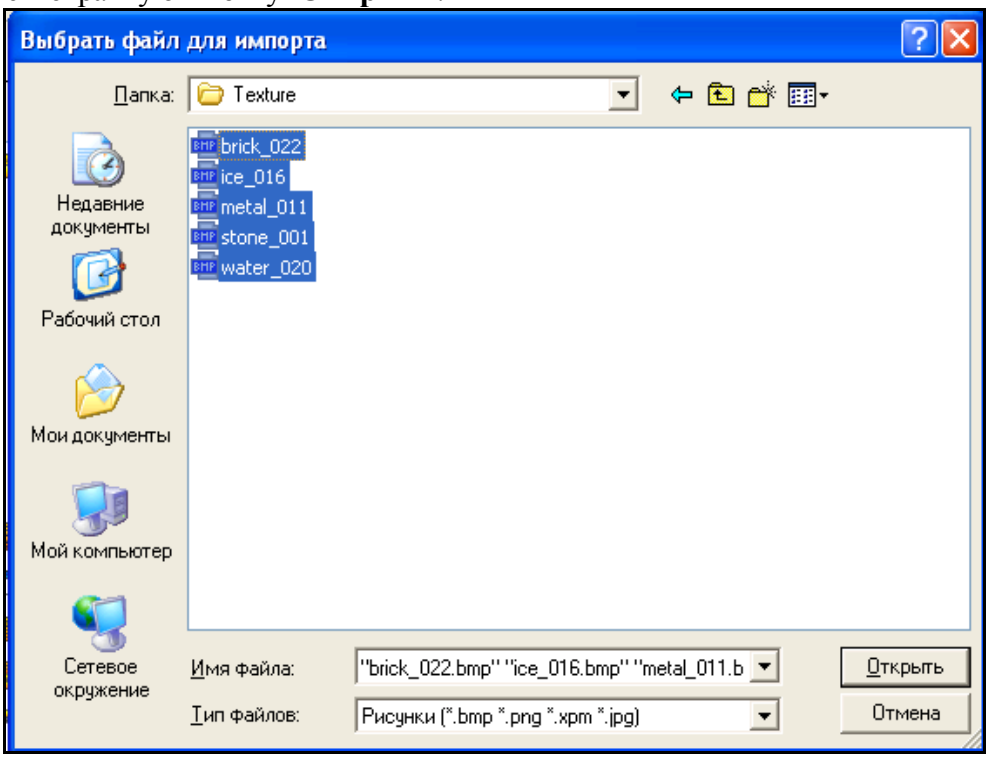


Рис. 9. Диалоговое окно выбора файлов для импорта

Содержимое библиотеки **Библиотека_Изображений#1** после указанных действий показано на рис. 10.

Подобным вышеописанному способом создадим в слое **Ресурсы** группу **Анимация**, в ней - библиотеку **Библиотека_Видеоклипов#1**. В открывшемся диалоге выбора файлов для импорта укажем поддиректорию **…\Lib\Animation**. Выберем файл **fluid_blue** и нажмем экранную кнопку **Открыть**. Содержимое библиотеки видеоклипов показано на рис. 11.

В зависимости от редакции используемой интегрированной среды разработки – базовой или профессиональной количество доступных текстур и видеоклипов в библиотеке различно.

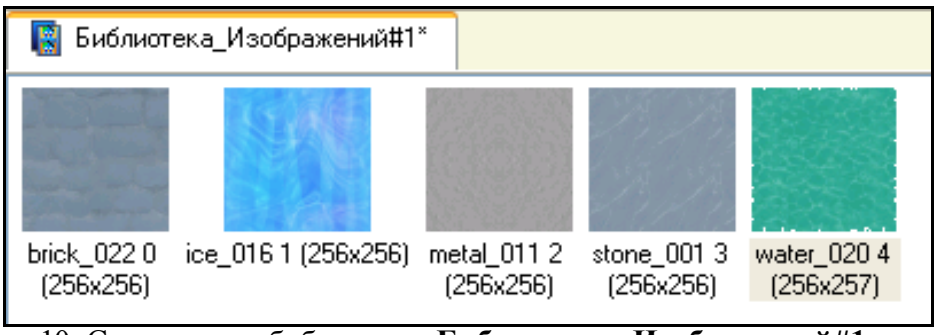


Рис. 10. Содержимое библиотеки **Библиотека_Изображений#1**

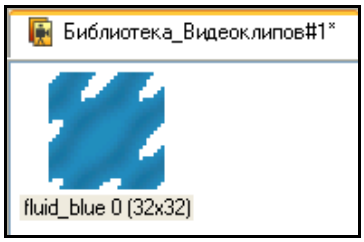


Рис. 11. Содержимое библиотеки **Библиотека_ Видеоклипов#1**

В качестве видеоклипов могут быть использованы практически любые имеющиеся файлы форматов avi или mng.

После проведения подготовительных мероприятий сохраним выполненную работу, нажав ЛК .

Добавление собственных компонентов в библиотеку

Также в библиотеке существует возможность добавления пользовательских графических элементов.

Для этого необходимо в слое **Ресурсы** выбрать существующую группу графических элементов (ГЭ) либо создать новую. Чтобы создать новую группу, необходимо нажать ПК мыши на слое **Ресурсы**. В появившемся контекстном меню выбрать **Создать группу**, затем **Графические_элементы**, как показано на рис. 12. В результате в слое **Ресурсы** будет создана группа **Графические_элементы_№** (под **№** понимается номер создаваемой группы), которую при необходимости можно переименовать.

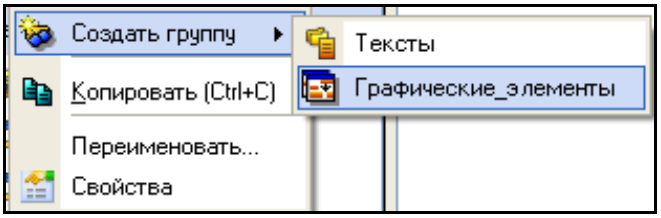


Рис. 12. Контекстное меню слоя **Ресурсы**

Чтобы создать графический объект, необходимо ПК мыши нажать на выбранную группу и в появившемся контекстном меню выбрать **Создать компонент**, затем **Графический_объект**. Данная операция показана на рис. 13.

Рис. 13. Контекстное меню группы графических элементов

В результате в слое Ресурсы в выбранной группе будет создан компонент **Графический_объект_1**, который затем можно переименовать.

Чтобы отредактировать созданный объект, необходимо дважды нажать ЛК мыши на выбранном компоненте.

Далее размещаем на появившемся экране графические элементы (ГЭ). Создание графических объектов описано в пункте 3.6.5.

Контрольные вопросы

1. Охарактеризуйте язык списка инструкций IL.
2. В чем назначение модификаторов в языке IL?
3. Назовите три способа для вызова функциональных блоков в языке IL.
4. Для чего служат модификаторы в языке IL?
5. Что такое оператор в языке IL?

С помощью какой команды производят прямое объявление адресов в языке IL?

Лабораторная работа №7 Разработка шаблонов графических экранов интегрированных систем управления

Этапы разработки шаблона графического экрана

Этапы разработки шаблона экрана:

а) добавление шаблона экрана;

б) настройка параметров экрана;

в) размещение графических элементов (ГЭ) экрана и задание статических атрибутов;

г) создание аргументов экрана;

д) настройка динамизируемых атрибутов ГЭ;

е) настройка динамических свойств ГЭ;

ж) размещение кнопок и настройка событий на их нажатие;

и) создание необходимых графических объектов (ГО) и размещение на экране.

Данные этапы могут выполняться в любой последовательности. При этом обязательными являются этапы **а** и **в.**

Примечание: Если у экрана не будет создано ни одного аргумента, то данный экран не будет загружен.

Рассмотрим проектирование шаблона экрана на примере экрана РВС.

Добавление шаблонов экранов

Перейдя в слой **Шаблоны_экранов**, создадим в нем компонент **Экран#1**. Для этого ПК мыши на слое **Шаблоны_экранов** вызовем контекстное меню, в котором выберем **Создать компонент→Экран**. Последовательность действий изображена на рис. 14.

Данный экран будет главным. На нем будет схематически изображен участок ЦППН. Создадим также экраны насосной, резервуарного парка, сепараторов, газового уравнителя и экран тревоги. Список экранов показан на рис. 15.

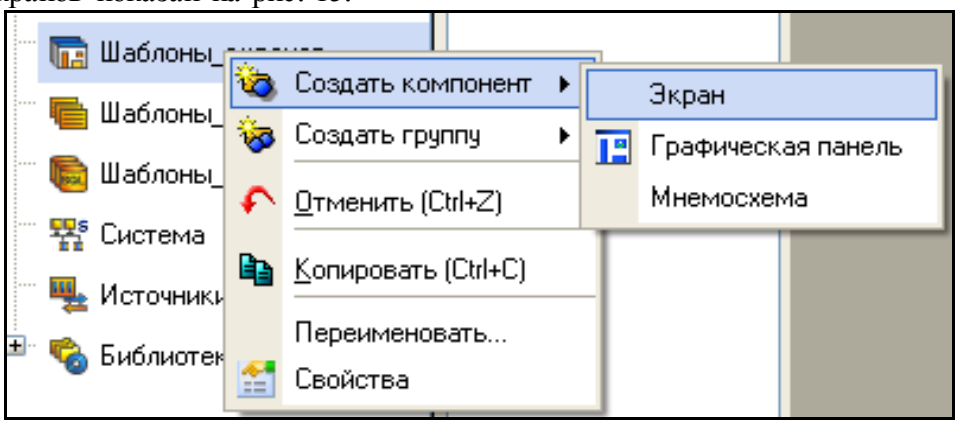


Рис. 14. Создание компонента **Экран** в слое **Шаблоны_экранов**

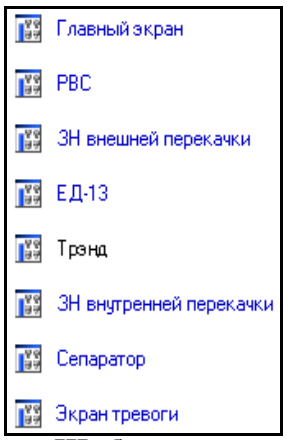


Рис. 15. Слой **Шаблоны_экранов**

Настройка параметров экрана

Откроем созданный экран двойным щелчком ЛК по пиктограмме **РВС**.

Зададим размер экрана и цвет фона созданного экрана ЛК по элементу строки меню **Сервис → Параметры экрана.**

Так как защита курсовых проектов по дисциплине «Синтез технических систем управления» происходит в мультимедийной аудитории с использованием программы **PowerPoint**, то для удобства создания презентации рекомендуется задавать разрешение экрана не более 1024 на 768, чтобы все элементы графического дисплея были четко видны и были достаточной величины. Используем разрешение экрана 800 на 600. После задания необходимых параметров закроем окно ЛК по кнопке **Готово**. Последовательность действий представлена на рис. 16.

Альтернативно можно выполнить двойной щелчок ЛК в любом свободном месте экрана и в открывшемся окне свойства объекта указать разрешение и цвет фона.

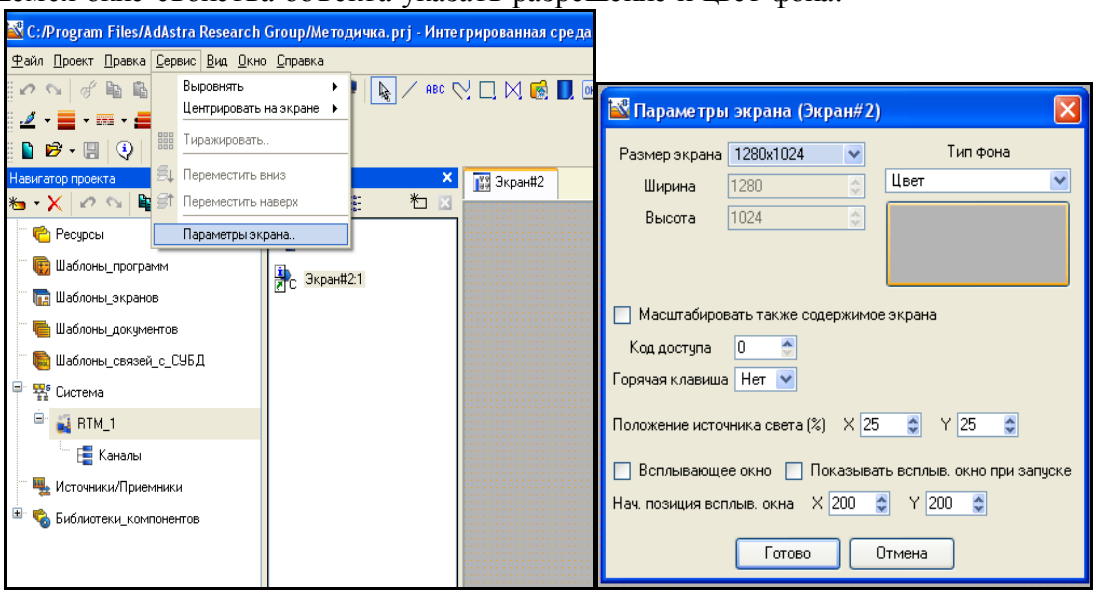


Рис. 16. Настройка параметров экрана

Размещение графических элементов (ГЭ) экрана и задание статических атрибутов

Чтобы выбрать ГЭ для размещения, нужно выполнить следующие действия:

- нажать ЛК на кнопке панели инструментов **Графические элементы**, показанной на рис. 17. При этом выбирается тот элемент, чья иконка выведена на кнопку (элемент, заданный по умолчанию для соответствующей группы, или элемент, выбранный ранее);

Рис. 17. Панель инструментов **Графические элементы**

- дважды нажать ЛК на кнопке и затем нажать ЛК на иконке требуемого ГЭ в появившемся меню (меню не открывается, если группа содержит только один графический элемент).

Например, на рис. 18 показано меню группы Прямоугольники.

Рис. 18. Меню группы **Прямоугольники**

После выбора элемента его иконка выводится на кнопку группы, как показано на рис. 19.

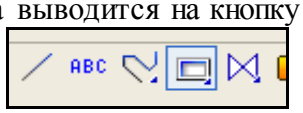


Рис. 19. Панель инструментов **Графические элементы** после выбора соответствующего ГЭ из группы

При выборе графического элемента редактор представления данных переходит в режим размещения, при этом курсор на графическом экране приобретает вид +. Далее в окне **Слои** необходимо нажатием ЛК указать слой, в котором должен быть размещен выбранный графический элемент. Если окно **Слои** закрыто, то ГЭ будет размещен в слое, в котором был размещен последний ГЭ. В случае если слой один, то необходимость в указании слоя отпадает. После размещения ГЭ его можно переместить в другой слой. Далее продолжить процедуру размещения ГЭ можно двумя способами:

- перетащить ГЭ с панели инструментов на экран (метод drag-and-drop); после размещения ГЭ имеет размеры, заданные по умолчанию, РПД переходит в режим редактирования, окно свойств ГЭ открывается автоматически. При перетаскивании с панели инструментов ГЭ групп **Ресурсы** и **Объекты** на экране размещается первый ресурс/объект первой библиотеки;

- переместить курсор в нужную точку экрана и нажатием ЛК установить точку привязки ГЭ. Далее действия по размещению ГЭ могут отличаться, однако для большинства графических элементов они стандартны – перемещение мыши после установки точки привязки выводит на экран образ ГЭ, при этом отрезок от точки привязки до текущего положения курсора является диагональю прямоугольника, ограничивающего ГЭ (если при перемещении мыши удерживать нажатой клавишу CTRL, ряд ГЭ окажется вписанным в квадрат). Повторное нажатие ЛК приводит к размещению графического элемента в выбранном графическом слое (в ряде случаев для размещения ГЭ на экране достаточно установить точку привязки).

Для графических элементов группы **Ломаные и кривые** каждое нажатие ЛК после установки точки привязки задает узловую точку (промежуточную вершину). Для установки последней вершины и выхода из режима размещения этих ГЭ нужно нажать ПК. Положение узловых точек, заданное при размещении, можно в дальнейшем изменить.

Для отмены размещения ГЭ в его процессе надо нажать ESC.

При пересечении ГЭ возникает необходимость разместить один ГЭ поверх другого или наоборот. Для этого нужно выделить ГЭ ЛК мыши. Затем вызвать контекстное меню ПК мыши и выбрать **Переместить наверх** для того, чтобы переместить на один уровень вверх или **Переместить вниз** для перемещения на один уровень вниз. Контекстное меню представлено на рис. 20. Пример выполнения команды показан на рис. 21.

В качестве задвижек без управления используем ГО **valve_1** из библиотеки из группы **Задвижки.** Резервуар разместим также из библиотеки из группы **Резервуары.**

После размещения ГЭ необходимо настроить их свойства.

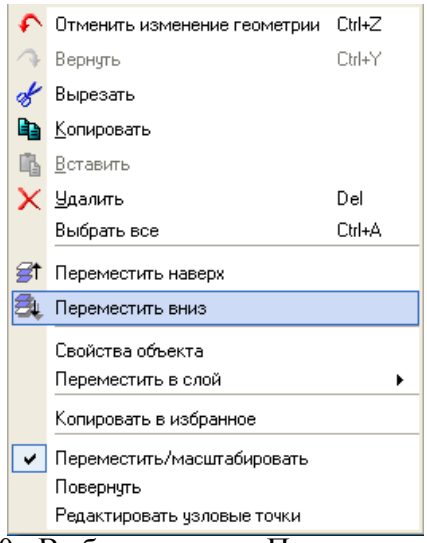
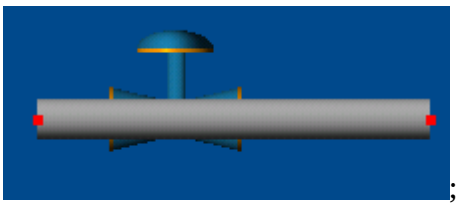
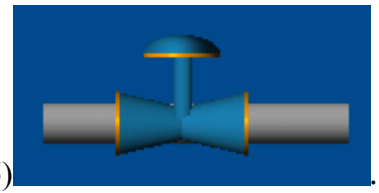


Рис. 20. Выбор команды Переместить вниз

а)  ; б)  .

а) до выполнения команды; б) после выполнения команды.

Рис. 21. Выполнение команды Переместить вниз с ГЭ Труба

Графические элементы имеют следующие настраиваемые свойства:

- атрибуты (статические и динамизируемые);
- динамические свойства;
- функции управления.

Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени.

В данном разделе описано задание статических атрибутов. Задание динамизируемых атрибутов описано в пункте 3.6.6, динамических свойств – в пункте 3.6.7, функций управления – в пункте 3.6.8.

У ГЭ **Текст** зададим свойство **Текст**. Для этого двойным щелчком ЛК на тексте вызовем окно свойств. Это окно продемонстрировано на рис. 22.

В поле **Значение** свойства **Текст** ввести отображаемый текст.

Аналогично задать необходимые параметры свойств **Контур**, **Заливка** и **Цвет текста**. Для задания дополнительных параметров данных свойств необходимо дважды ЛК мыши нажать в поле **Свойство** на требуемом свойстве.

Все свойства, выделенные коричневым цветом и подчеркиванием, имеют дополнительные параметры, которые можно вызвать двойным нажатием ЛК мыши на названии свойства.

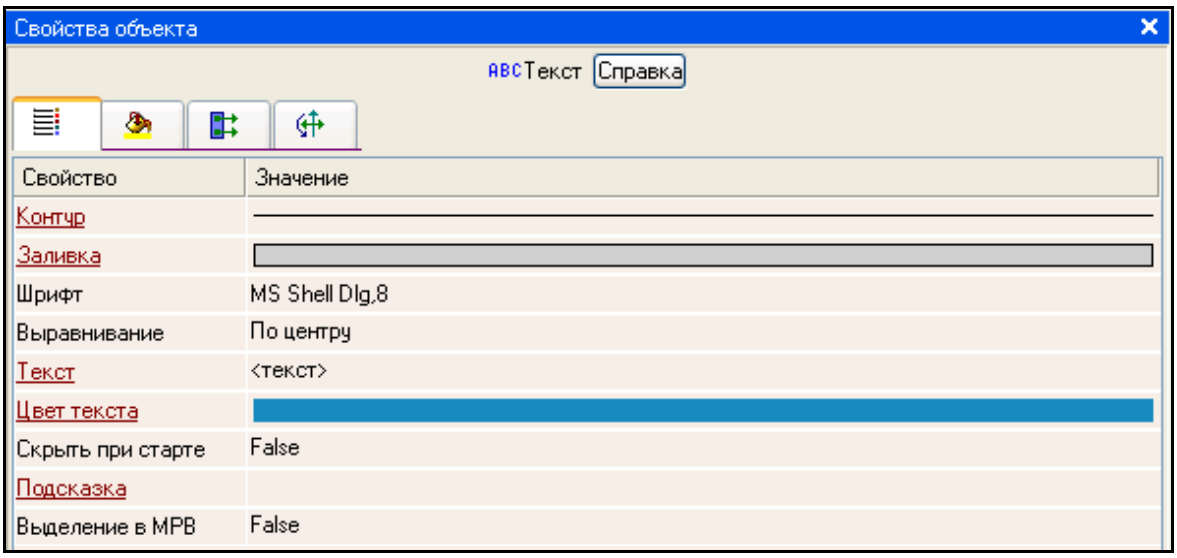


Рис. 22. Окно свойств ГЭ **Текст**

После размещения ГЭ и ГО получаем экран РВС, изображенный на рис. 23.

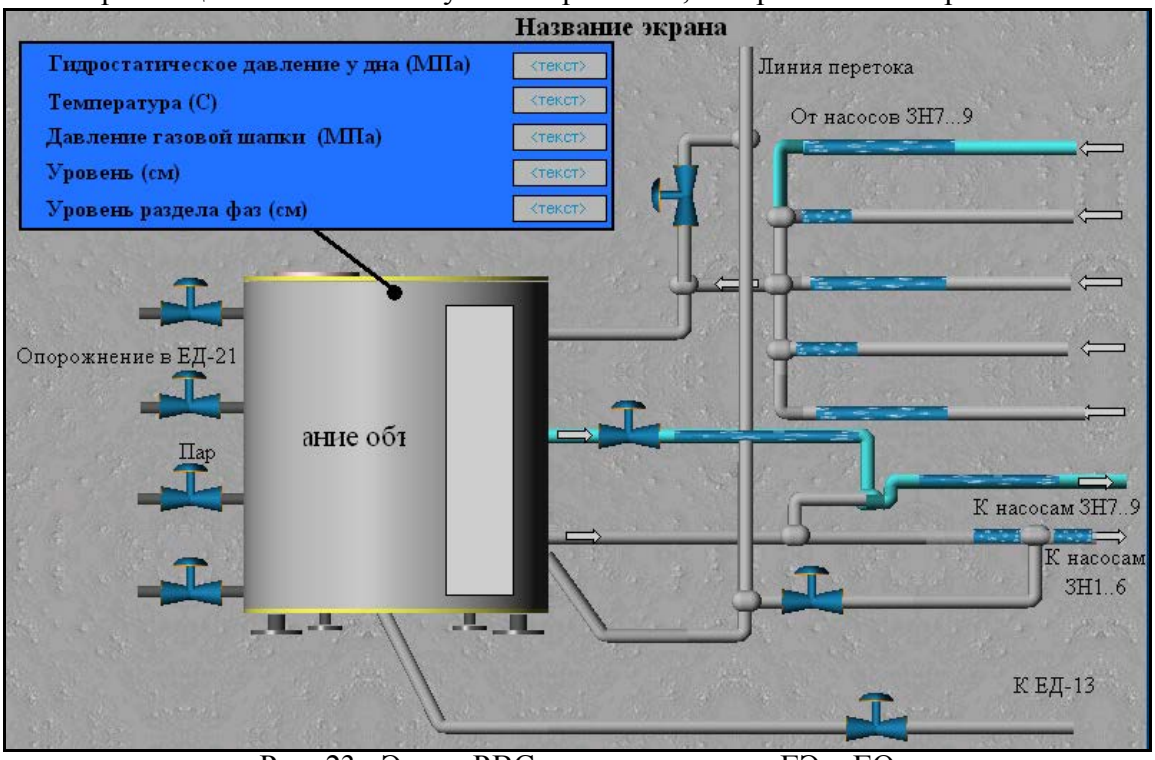


Рис. 23. Экран РВС с размещенными ГЭ и ГО

Создание аргументов экрана

В соответствии с техническим заданием на проектирование необходимо назначить аргументы шаблону экрана – щелчок ПК на созданном шаблоне экрана и выбор из выпадающего списка пункта **Свойства**, далее переход во вкладку **Аргументы**. Окно свойств с открытой вкладкой **Аргументы** показано на рис. 24.

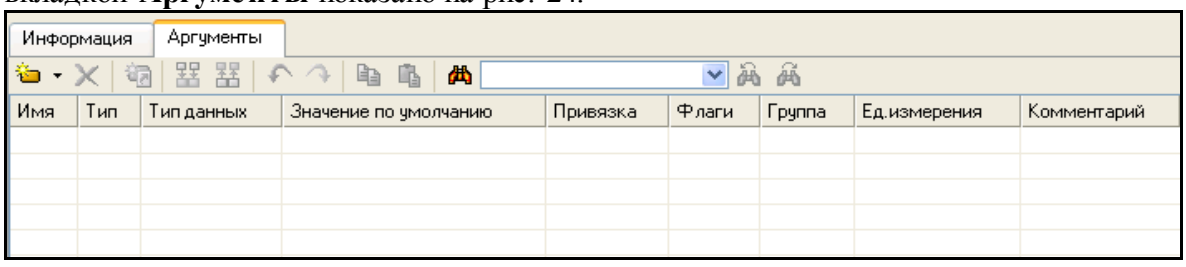


Рис. 24. Окно свойств с открытой вкладкой **Аргументы**

Чтобы создать аргумент, необходимо нажать на кнопку . После этого требуется настроить параметры аргумента.

Основные параметры аргументов и редактирование их в поле редактора аргументов:

- **Имя** – имя аргумента, задается по умолчанию и может быть изменено. Для перехода к редактированию имени аргумента нужно дважды нажать ЛК в данном поле. Имена аргументов одного и того же компонента должны быть уникальными.

- **Тип** – тип аргумента (INPUT, OUTPUT и INPUT/OUTPUT). Аргумент типа INPUT предназначен для приема значений, типа OUTPUT – для передачи (это следует учитывать при привязке аргумента). Тип INPUT/OUTPUT имеет самостоятельный смысл при вызове программы, в остальных случаях этот тип равнозначен типу OUTPUT. Тип аргумента выбирается в списке, который открывается при двойном нажатии ЛК в данном поле.

- **Тип данных** – тип данных аргумента. Тип данных выбирается в списке, который открывается при двойном нажатии ЛК в данном поле. Этот параметр должен учитываться при привязке аргументов.

- **Значение по умолчанию** – значение, указанное в этом поле, используется монитором в случае отсутствия привязки у аргумента. Для перехода к редактированию этого параметра нужно дважды нажать ЛК в данном поле.

- **Привязка** – привязка аргумента к атрибуту определенного канала или к аргументу экрана или программы. При двойном нажатии ЛК в данном поле на экране появляется диалог выбора компонента/аргумента для привязки.

- **Флаги** – флаги, установленные для аргумента. В списке, который открывается при двойном нажатии ЛК в данном поле, для аргумента могут быть установлены следующие флаги: HW, SL, NP, PO. От флагов зависит результат автопостроения/привязки каналов. Автопостроение описано в приложении Г.

- **Группа** – номер группы, к которой принадлежит аргумент. Для группирования нужно выделить несколько аргументов и нажать кнопку **Группировать выделенные аргументы** – по этой команде выделенной группе аргументов автоматически присваивается номер. От параметра **Группа** зависит результат автопостроения каналов из редактора аргументов. Автопостроение описано в приложении Г.

- **Единицы измерения** – единицы измерения (или унифицированный сигнал) выбираются в списке, который открывается при двойном нажатии ЛК в данном поле. От этого параметра зависит результат автопостроения/привязки каналов узла из редактора аргументов. Автопостроение описано в приложении Г.

- **Комментарий** – комментарий к аргументу. Для перехода к редактированию этого параметра нужно дважды нажать ЛК в данном поле [19].

Те аргументы, значения которых будут отображаться на экране, имеют тип **IN**, а те, что задаются с клавиатуры АРМ, отображаются на экране и пересылаются в конечном итоге в контроллер, имеют тип **OUT**. В процедуре автопостроения каналов от шаблонов автопривязка аргументов будет осуществляться соответственно к атрибутам **Реальное** и **Входное значение** каналов. Автопостроение описано в приложении Г.

Аргументы, для которых задан тип **IN**, не будут пересылаться в контроллер, даже если на экране для этого будут созданы необходимые инструменты. Поэтому для аргументов, значение которых будет задаваться на экране, необходимо задать тип **OUT.**

Флаг **NP**, выставленный для аргументов, не позволит создавать соответствующие каналы при операциях автопостроения.

Создадим аргументы для экрана РВС. Аргументы представлены на рис. 25.

В нашем примере для аргументов **Состояние_задв_№** и **Команда_упр_задв_№** (под № понимается номер задвижки) установлен флаг **NP**, так как для них не требуется создание каналов.

Настройка динамизируемых атрибутов ГЭ

Динамизацией атрибута называется задание условий его изменения в зависимости от

значения привязанного аргумента. При динамизации атрибута графический элемент становится индикатором выполнения заданных условий.

При размещении ГЭ на экране все его динамизируемые атрибуты по умолчанию статические.

Динамизируемые атрибуты выделены коричневым цветом и имеют подчеркивание. Чтобы изменить данные атрибуты, необходимо нажать двойным нажатием ЛК мыши на названии свойства и выбрать вид динамизации, нажав ЛК мыши в поле **Нет динамизации** и выбрав нужный пункт в раскрывающемся списке.

Имя	Тип	Тип данных	Значение по умолчанию	Привязка	Флаги
Гидростатическое_давление_дна	IN	REAL			
Температура	IN	REAL			
Давление_газовой_шапки	IN	REAL			
Уровень	IN	REAL			
Уровень_раздела_фаз	IN	REAL			
Номер_резервуара	IN	USINT			NP
Состояние_задв_1	IN	USINT			NP
Команда_упр_задв_1	OUT	USINT			NP
Состояние_задв_2	IN	USINT			NP
Команда_упр_задв_2	OUT	USINT			NP
Состояние_задв_3	IN	USINT			NP
Команда_упр_задв_3	OUT	USINT			NP
Состояние_задв_4	IN	USINT			NP
Команда_упр_задв_4	OUT	USINT			NP
Состояние_задв_5	IN	USINT			NP
Команда_упр_задв_5	OUT	USINT			NP
Состояние_задв_6	IN	USINT			NP
Команда_упр_задв_6	OUT	USINT			NP
Состояние_задв_7	IN	USINT			NP
Команда_упр_задв_7	OUT	USINT			NP
Состояние_задв_8	IN	USINT			NP
Команда_упр_задв_8	OUT	USINT			NP
Состояние_задв_9	IN	USINT			NP
Команда_упр_задв_9	OUT	USINT			NP

Рис. 25. Аргументы шаблона экрана РВС

Существует 6 видов динамизации:

- значение – индикация значения аргумента (только для атрибута Текст);
- Arg = Конст. – индикация равенства аргумента заданной константе;
- Arg >= Конст. – индикация превышения аргументом заданного порога;
- Arg & Конст. – индикация состояния битов значения аргумента, заданных маской Константа. Если хотя бы один такой бит установлен, индицируется ИСТИНА, иначе – ЛОЖЬ;
- Arg в диапазоне – индикация нахождения аргумента в заданных диапазонах;
- Arg в интервале – индикация нахождения аргумента в интервалах привязанного канала.

В зависимости от выбранного вида индикации меняются инструменты его конфигурирования. На рис. 26 показаны свойства динамизируемого атрибута **Текст** с видом динамизации **Значение**.

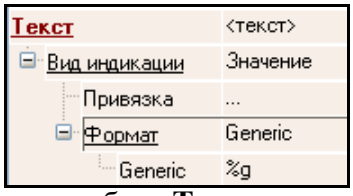


Рис. 26. Свойства динамизируемого атрибута **Текст** с видом динамизации **Значение**

После выбора вида динамизации необходимо указать привязку динамизируемого атрибута. Для этого необходимо нажать на поле **Значение** параметра **Привязка**. Появится диалоговое окно **Свойства привязки,** как показано на рис. 27.

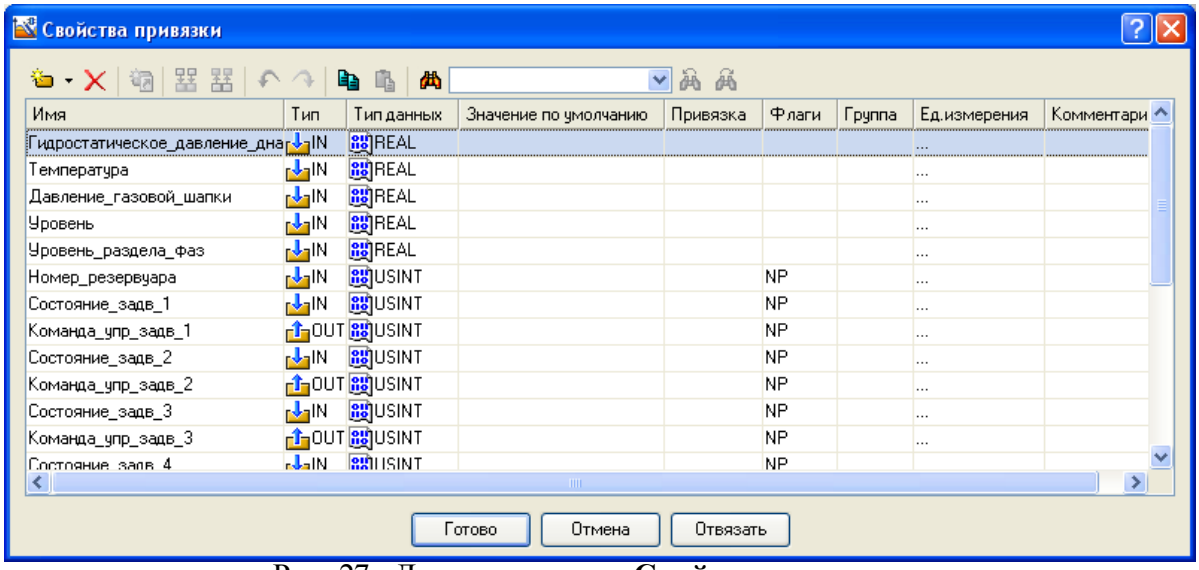


Рис. 27. Диалоговое окно **Свойства привязки**

В данном окне отображаются все аргументы шаблона экрана. В нем необходимо выбрать нужный аргумент, к которому предполагается привязать димамизируемый атрибут ГЭ. Затем следует нажать на кнопку **Готово.**

После этого настраиваются другие параметры динамизации.

Для вида динамизации «Значение» необходимо настроить параметр Формат. Точное описание данного параметра задается в нотации Си.

Обозначению формата предшествует знак процента (%):

- %d или %i – вывод значения как целого со знаком в формате DEC;
- %o – вывод значения в восьмеричном формате без знака;
- %u – вывод значения как целого без знака в формате DEC;
- %x – вывод значения в формате HEX без знака с использованием нижнего регистра для букв;
- %X – вывод значения в формате HEX без знака с использованием верхнего регистра для букв;
- %e – вывод значения со знаком в форме [–]D.mmmm e [sign]ddd, где D – один десятичный знак, mmmm – один или более десятичных знаков, ddd – три десятичных знака, sign – "+" или "-";
- %E – аналог %e с использованием E вместо e;
- %f – вывод значения со знаком в форме [–]DDDD.mmmm, где DDDD – один или более десятичных знаков. Число знаков перед десятичной точкой зависит от величины значения, число знаков после десятичной точки зависит от запрошенной точности. Число знаков после запятой (k) может быть задано при указании формата в виде %.<k>f;
- %g – вывод значения со знаком в f или e формате (в зависимости от того, в каком из этих двух форматов представление компактнее для данного числа, и точности). Формат e используется тогда, когда показатель степени меньше -4 или больше или равен точности числа. Замыкающие нули удаляются, десятичная точка появляется только тогда, когда число дробно;
- %G – аналог %g с использованием E вместо e.

Настройка динамических свойств ГЭ

К динамическим свойствам графических элементов относятся динамическая заливка, 3 вида динамической трансформации (перемещение, масштабирование и вращение) и динамический контур.

Динамические свойства ГЭ, как и динамизированные атрибуты, используются для графического отображения значений аргументов экрана при работе в реальном времени.

Динамические свойства настраиваются соответственно на вкладках **Динамическая заливка** (), **Динамический контур** () и **Динамическая трансформация** () окна **Свойства**

объекта.

Размещение кнопок и настройка событий на их нажатие

После создания экранов для каждого объекта создадим кнопки для навигации в **HMI**.

Для этого воспользуемся ЛК мыши, выберем иконку графического элемента (ГЭ) «Кнопки»  . Затем разместим кнопку на экране **HMI**. Далее в автоматически открывшихся свойствах ГЭ “Кнопки” → **Основные свойства** в области **Текст** укажем текст, располагаемый на кнопке. Затем откроем вкладку **Действие** и щелчком ПК в области **mousePressed** в открывшемся выпадающем меню выберем пункт **Перейти на экран**, в котором укажем, на какой экран переходить в случае нажатия кнопки, как показано на рис. 28. Кнопка будет работать в режиме MPB, если проект запустить в профайлере. Переход на экран возможен, если экран, на который переходят, содержит хотя бы один аргумент.

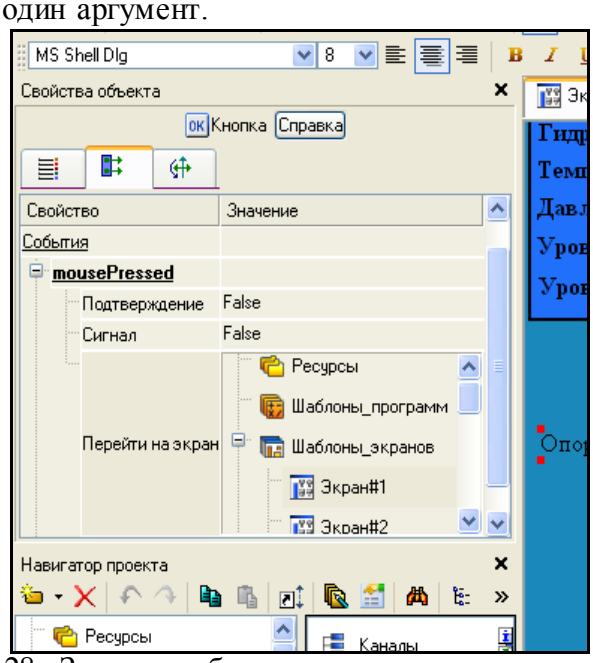


Рис. 28. Задание события при нажатии на кнопку

Получившийся экран показан на рис. 29.

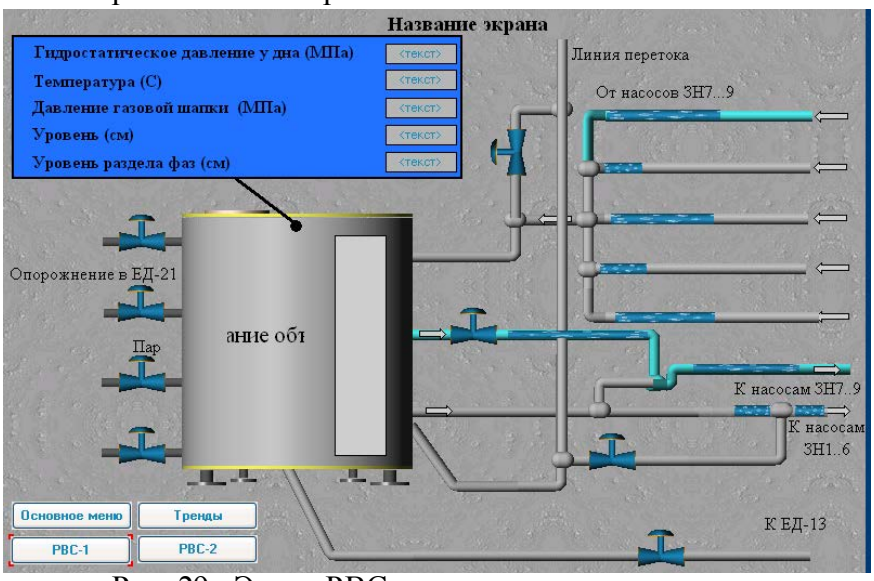


Рис. 29. Экран РВС с размещенными кнопками

Создание графических объектов

Добавим в слой **Ресурсы** в необходимую группу графический объект. Добавление

графических объектов (ГО) описано в пункте 3.5.2.

Рассмотрим создание ГО на примере задвижки с приводом.

В нашем случае добавим в слой **Ресурсы** в группу **Задвижки** ГО и переименуем его на **valve_2.**

Размещение ГЭ и ГО аналогично размещению на экране ГЭ и ГО и подробно описано в пункте 3.6.4.

Разместим ГЭ **Клапан** и **Кнопка.**

Для ГЭ **Клапан** зададим следующие свойства:

- Материал→Материалы=Золото;

- Форма клапана=  ;

- Форма привода=  .

Для атрибута **Цвет привода** зададим тип динамизации **Arg в диапазоне**. Зададим привязку атрибута. Для этого ЛК мыши нажмем на кнопку в поле **Привязка** [4]. В появившемся окне добавим аргумент **Состояние_задвижки**, как описано в пункте 3.6.5. Зададим данному аргументу свойства, как показано на рис. 30.

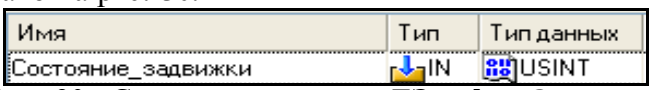


Рис. 30. Свойства аргумента ГЭ **valve_2**

После этого нажмем кнопку **Готово.**

Создадим 5 диапазонов. Настроим их, как показано на рис. 31.

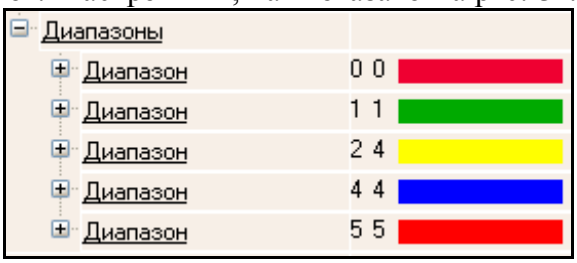


Рис. 31. Диапазоны динамизации для атрибута **Цвет привода**

Для ГЭ **Кнопка** зададим свойства, как показано на рис. 32.

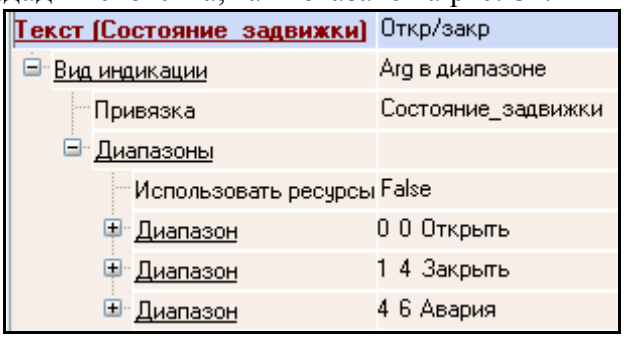


Рис. 32. Атрибут **Текст** ГЭ **Кнопка**

Для кнопки зададим событие **Передать значение**. Тип передачи **Прямой**. В качестве привязки выберем аргумент **Команда_управления_задвижкой,** для которого зададим свойства, представленные на рис. 33.

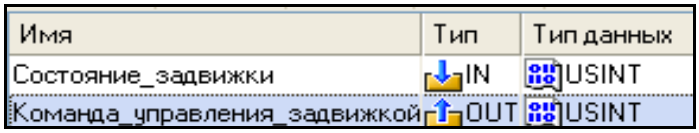


Рис. 33. Свойства аргумента ГЭ **valve_2**

Поле **Значение** зададим равным 1, что будет соответствовать смене состояния задвижки.

На рис. 34 изображен ГЭ **valve_2** после вышеописанных операций.

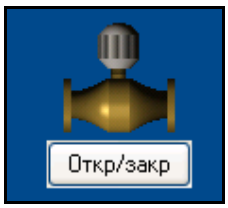


Рис. 34. ГЭ **valve_2**

Разместим созданный ГЭ на шаблоне экрана РВС, как описано в пункте 3.6.4. В нашем примере на экране РВС должно быть 9 управляемых задвижек.

После размещения двойным нажатием ЛК мыши вызовем окно свойств ГЭ **valve_2**. Привяжем для каждой задвижки атрибуты **Состояние_задвижки** и **Команда_управления_задвижкой** к соответствующим аргументам шаблона экрана РВС **Состояние_зад_№** и **Команда_упр_задв_№** (под № понимается номер задвижки).

Контрольные вопросы

1. Цели разработки интегрированных систем управления
2. Задачи разработки интегрированных систем управления
3. Возможности разработки интегрированных систем управления

Лабораторная работа №8 Разработка шаблонов программ интегрированных систем управления

Общие сведения о языках программирования

В прошлом большое количество языков программирования, связанных с конкретным производителем и в той или иной степени удовлетворяющих общим стандартам (DIN 19239, DIN 19237, DIN 40719 Part 6 and VDI 2880), порождало массу проблем. Все это привело к возникновению разнообразных решений. Языки программирования релейных диаграмм - ladder diagram (LD), функциональных блоковых диаграмм - function block diagram (FBD) и список инструкций - instruction list (IL) сформировались постепенно под влиянием многих участников рынка вместе с различными версиями языка последовательных функциональных схем - sequential function chart (SFC), таких как Grafcet, GRAPH 5 and CSF to DIN 40719 Part 6. Тем не менее элементы языка у разных производителей отличались, и всегда существовали машинозависимые особенности языков, которые приходилось учитывать.

Для того, чтобы решить эту проблему, международный электротехнический комитет (МЭК, IEC) разработал стандарт технологических языков программирования контроллеров IEC 61131, третья часть которого описывает языки программирования ПЛК.

Международный стандарт IEC 61131-3 описывает следующие языки программирования:

а)графические языки:

1) последовательных функциональных схем (SFC);
2) релейных диаграмм (LD);
3) функциональных блоковых диаграмм (FBD);

б) текстовые языки:

1) список инструкций (instruction list - IL);
2) структурированный текст (Structured Text - ST).

Синтаксис этих языков детально описан в стандарте IEC 61131-3, так что пользователь найдет один и тот же синтаксис во всех пакетах, поддерживающих этот стандарт. Графическое представление прикладных программ - это типичная особенность программирования PLC. В то же время текстовые языки широко используются для программирования компьютеров. Языки программирования определены в стандарте таким образом, что допускают разработку приложения на смеси этих языков, которая впоследствии собирается в единую исполняемую программу. Кроме того, стандарт открыт для использования других языков программирования.

В Trace Mode имеются следующие языки программирования:

а) ST-программа;
б) SFC-диаграмма;
в) FBD-диаграмма;
г) LD-диаграмма;
д) IL-программа.

Программа на языке ST представляет собой последовательность предложений или выражений. Язык ST похож на язык программирования C++, поэтому для программирования на ST необходимо знать синтаксис языка C++.

Язык SFC позволяет создавать программы в виде алгоритма, состоящего из SFC-шагов и SFC-переходов.

FBD-программа представляет собой цепочку (диаграмму) последовательно выполняемых функциональных блоков. Удобством создания FBD-программы является использование готовых блоков (выбор из списка) и наглядность отображения созданной программы.

LD-программа представляет собой диаграмму последовательно выполняемых функциональных блоков. Программа на языке IL представляет собой последовательность инструкций ассемблерного типа, но без ориентации на конкретную микропроцессорную архитектуру. Языки LD и IL относят к низкоуровневым языкам.

Этапы разработки шаблона программ

Основные этапы разработки шаблона программы:

- создание шаблона программы;
- создание аргументов шаблона программы;
- выбор языка программирования и создание программы на выбранном языке программирования;
- компиляция и отладка программы.

Как и в случае с экранами, данные этапы могут выполняться в любой последовательности. Также в любой момент можно вернуться к любому этапу.

Рассмотрим разработку шаблона программы на примере программы управления задвижкой.

Создание шаблона программы

Продолжая разработку проекта, создадим шаблоны программ. Чтобы добавить шаблон программы, в левом окне навигатора проекта ЛК выберем слой **Шаблоны_программ**, по щелчку ПК вызовем контекстное меню и выберем **Создать компонент→Программа**. Контекстное меню показано на рис. 35.

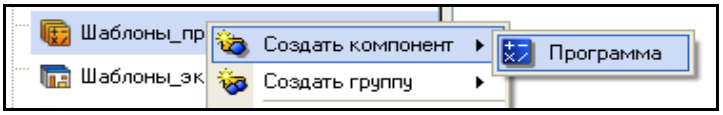


Рис. 35. Создание шаблона программы

Название программы можно переименовать. Для этого, выделив созданный компонент ЛК, изменим его имя на нужное.

Переименуем программу на **Управление_задвижкой.**

Для удобства созданные шаблоны программ можно группировать. Для этого необходимо ПК мыши на слое **Шаблоны_программ** вызвать контекстное меню и выбрать **Создать группу→Группа**. Группы можно также создавать внутри других групп.

Создание аргументов шаблона программы

В соответствии с техническим заданием на проектирование необходимо назначить аргументы шаблону программы. Для этого двойным щелчком ЛК на созданном шаблоне программы откроем окно редактора шаблонов программ и, выделив ЛК пункт **Аргументы**, перейдем в табличный редактор аргументов. Создадим аргументы для данного шаблона программы, как описано в пункте 3.6.5.

Аргументы, для которых задан тип **IN**, не будут меняться в результате выполнения программы, даже если в программе будут присутствовать операции присвоения. Поэтому для аргументов, значение которых будет задаваться в программе, необходимо задать тип **IN/OUT** или **OUT.**

В FBD-диаграммах аргументы, тип которых задан значением **IN**, можно привязать только ко входам FBD-блоков; значением **OUT** – к выходам FBD-блоков; **IN/OUT** – ко входам и выходам FBD-блоков.

На рис. 36 представлены необходимые аргументы, созданные для шаблона программы **Управление_задвижкой.**

Имя	Тип	Тип данных
Команда_управления	IN/OUT	USINT
Состояние_задвижки	OUT	USINT
Закрыть	OUT	USINT
Открыть	OUT	USINT
Авария	IN	USINT
Открыта	IN	USINT
Закрыта	IN	USINT

Рис. 36. Аргументы программы **Управление_задвижкой**

Разработка программы

После определения входных и выходных аргументов приступим непосредственно к разработке программы.

В данном примере рассмотрим написание программы на языке программирования ST.

Для этого выделим ЛК имя созданного шаблона программы и в появившемся диалоге выбора языка программирования укажем **ST программу**, как показано на рис. 37.

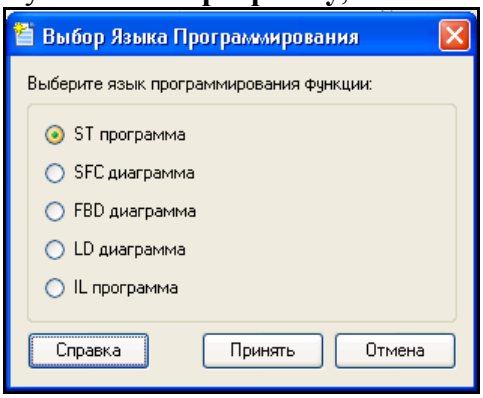


Рис. 37. Окно выбора языка программирования

После выбора языка программирования появляется заготовка программы, представленная на рис. 38. После ключевого слова **PROGRAM** идут объявления аргументов программы.

Текст, выделенный серым цветом, нельзя редактировать. Данный блок будет автоматически обновляться при добавлении, изменении или удалении аргументов.

```
PROGRAM
    VAR_INOUT Команда_управления : USINT; END_VAR
    VAR_OUTPUT Состояние_задвижки : USINT; END_VAR
    VAR_OUTPUT Закрыть : USINT; END_VAR
    VAR_OUTPUT Открыть : USINT; END_VAR
    VAR_INPUT Авария : USINT; END_VAR
    VAR_INPUT Открыта : USINT; END_VAR
    VAR_INPUT Закрыта : USINT; END_VAR


END_PROGRAM
```

Рис. 38. Заготовка программы

Текст программы заносится в белое поле.

В TRACE MODE 6 определены следующие операторы, образующие предложения ST-программы:

- return;
- if;
- case;
- while;
- repeat;
- for;
- break;
- exit;
- continue;
- goto.

Также при написании программы на языке ST можно использовать символьные операторы:

- арифметические операторы (+, -, /, *, %, mod, **);
- побитовые операторы (&, |, ^, xor, ~, <<, shl, >>, shr, rol, ror);

– операторы сравнения (==, !=, <>, <, >, <=, >=);
– логические операторы (||, &&, !, or, and, not);
– операторы присваивания (=, :=).

Подробное описание вышеперечисленных операторов можно найти в справке TRACE MODE в разделе **Программирование алгоритмов→Описание языка Техно ST→Операторы языка Техно ST→ Операторы Техно ST**.

Текст программы **Управление_задвижкой**, а также все остальные разработанные шаблоны программ представлены в приложении Е.

Отладка программы

Общие сведения

Средства отладки включают в себя несколько режимов непрерывного и пошагового выполнения программы с возможностью установки точек останова. Для запуска требуемого режима отладки можно использовать команды меню **Программа** главного меню или аналогичные команды панели инструментов отладчика. Для вывода служебных сообщений отладчика и компилятора предусмотрены специальные окна.

Отладка программы возможна только после ее успешной компиляции.

В листинге текстовых программ точка останова обозначается значком  . При пошаговой отладке текущий шаг обозначается значком  , как показано на рис. 39.

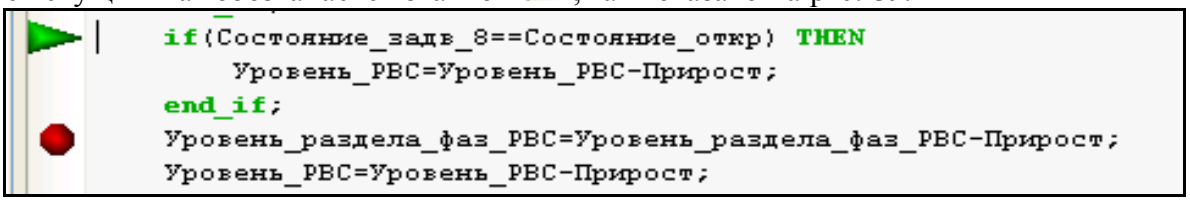


Рис. 39. Программа в режиме отладки

В программах, заданных в графическом виде, закладки, текущий шаг и точки останова обозначаются соответствующим цветом, как показано на рис. 40.

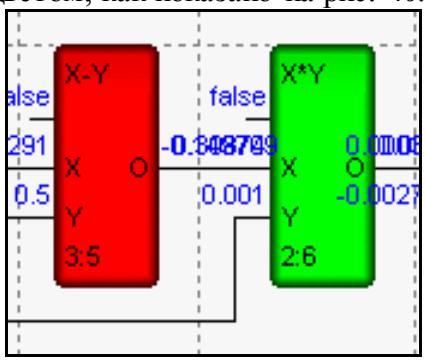


Рис. 40. FBD-диаграмма в режиме отладки

Панель инструментов отладчика

Меню **Программа** главного меню и панель инструментов отладчика содержат следующие команды для запуска требуемого режима отладки и настройки параметров редакторов программ:

 - компиляция (F7) – запустить компиляцию программы;

 - установить/удалить точку останова (F9) – установить/удалить точку останова программы;

 - удалить все точки останова (Ctrl+Shift+F9) – удалить все точки останова программы;

 - старт (F5) – запустить выполнение программы в непрерывном режиме;

 - выполнять до курсора (Ctrl+F10) – запустить выполнение программы в непрерывном режиме до текущей позиции курсора;

 - трассировка (F11) – запустить пошаговое выполнение программы с пошаговым выполнением вызываемых функций;

 - шаг (F10) – запустить пошаговое выполнение программы с выполнением вызываемых

функций в непрерывном режиме;

- выйти из функции (Shift+F11) – выполнить текущую функцию/программу в непрерывном режиме (доступно в режиме отладки);

- стоп (Shift+F5) – выйти из режима отладки;

- посмотреть значение переменной (Shift+F9) – открыть диалог Быстрый просмотр (доступно в режиме отладки);

- переменные – открыть/закрыть окно переменных;

- стек – открыть/закрыть окно стека вызовов функций;

- сообщения – открыть/закрыть окно сообщений компилятора и отладчика.

Окно просмотра переменных

Данное окно включает в себя в виде вкладок 5 окон просмотра текущих значений переменных и показано на рис. 41.

В окне **Локальные** отображаются переменные текущего программного компонента.

В окнах **Аргументы**, **Глобальные** и **Текущая структура** отображаются соответственно аргументы, глобальные переменные и переменные объекта.

Имя	Тип данных	Десятичное	Шестнадцатиричное	Двоичное
Состояние_задв_1	USINT	0	16#00	2#00000000
Состояние_задв_2	USINT	0	16#00	2#00000000
Состояние_задв_3	USINT	0	16#00	2#00000000
Состояние_задв_4	USINT	0	16#00	2#00000000
Состояние_задв_5	USINT	0	16#00	2#00000000
Уровень_РВС	REAL	0	16#00000000	2#00000000000000000000000000000000
Уровень_раздела_фаз_РВС	REAL	0	16#00000000	2#00000000000000000000000000000000
Состояние_задв_6	USINT	0	16#00	2#00000000
Прирост	REAL	1	16#00000001	2#00000000000000000000000000000001
Состояние_откр	USINT	1	16#01	2#00000001
Состояние_задв_7	USINT	0	16#00	2#00000000
Состояние_задв_8	USINT	0	16#00	2#00000000
Состояние_задв_9	USINT	0	16#00	2#00000000
Состояние_задв_10	USINT	0	16#00	2#00000000

Аргументы | Локальные | Глобальные | Текущая структура | Просмотр

Рис. 41. Окно переменных

При пошаговой отладке программы в окнах **Локальные** и **Глобальные** можно вручную задать значения переменных (для перехода к заданию значения нужно дважды нажать ЛК в поле значения или выполнить команду **Изменить значение** из контекстного меню). Для восстановления значений аргументов по умолчанию контекстное меню окна **Аргументы** содержит команды **Восстановить значение по умолчанию** и **Восстановить все значения по умолчанию.**

В окне **Просмотр** отображаются переменные и выражения, заданные пользователем. По умолчанию окно пустое. Список переменных и выражений для просмотра задается с помощью диалога **Быстрый просмотр**. Корректировать список можно в самом окне с помощью команд контекстного меню, которое выводится на экран при нажатии ПК в области окна. Меню содержит следующие команды:

Вставить – добавить строку в список;

Удалить – удалить выделенную строку из списка (аналог нажатия клавиши **Del** на

клавиатуре);

Переименовать – редактировать выделенную строку списка.

Компиляция программы

Перед отладкой и включением в состав проекта разработанный шаблон программы необходимо скомпилировать. Для этого используем иконку на панели инструментов или нажимаем функциональную клавишу F7. Результат компиляции показывается в окне **Вывод**, которое может быть открыто с помощью иконки на панели инструментов либо из основного меню интегрированной среды разработки, как показано на рис. 42.

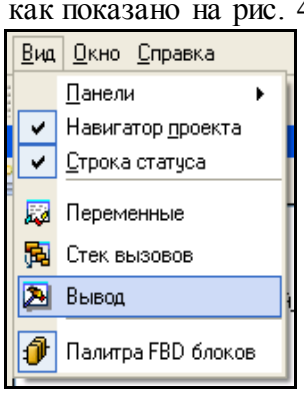


Рис. 42. Включение окна Вывод из основного меню интегрированной среды

В нашем случае данное окно содержит сообщение об успешном окончании процесса компиляции, что показано на рис. 43.

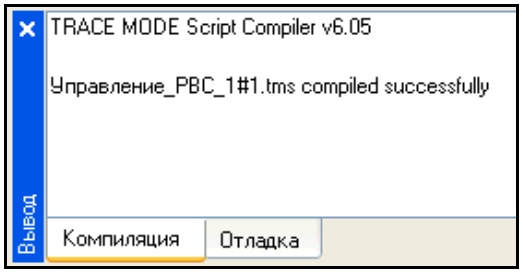


Рис. 43. Результат компиляции в окне **Вывод**

Узлы проекта и базы каналов

После разработки шаблонов экранов и программ необходимо создать узлы проекта АРМ, для которого в дальнейшем будем формировать базы каналов, используя механизм автопостроения. Автопостроение подробно описано в приложении Г. Проведем выбор слоя **Система** щелчком ЛК мыши. Далее с помощью ПК создадим узел MPB для АРМ как показано на рис. 44. Описание узлов проекта в TRACE MODE описано в приложении В.

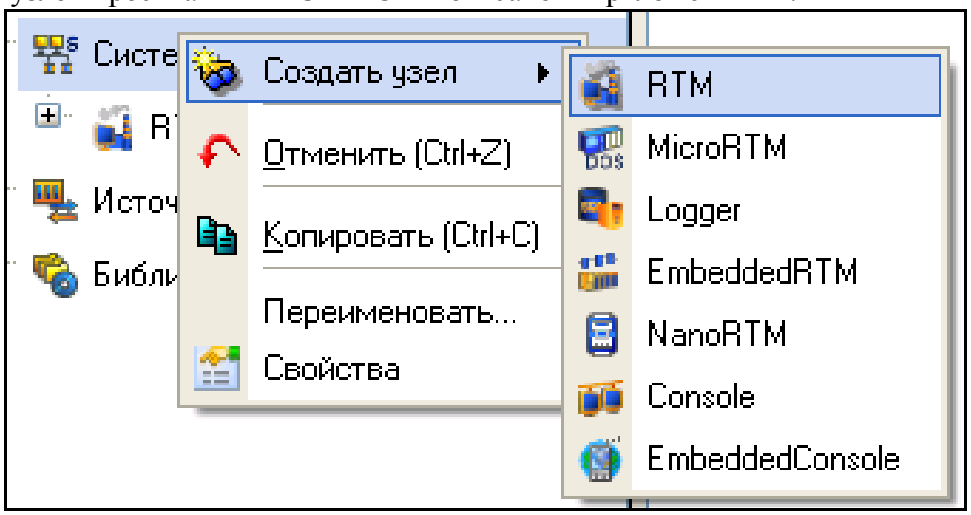


Рис. 44. Создание узла MPB для АРМ

В результате выполненных действий в слое **Система** будет создан узел проекта, как показано на рис. 45.

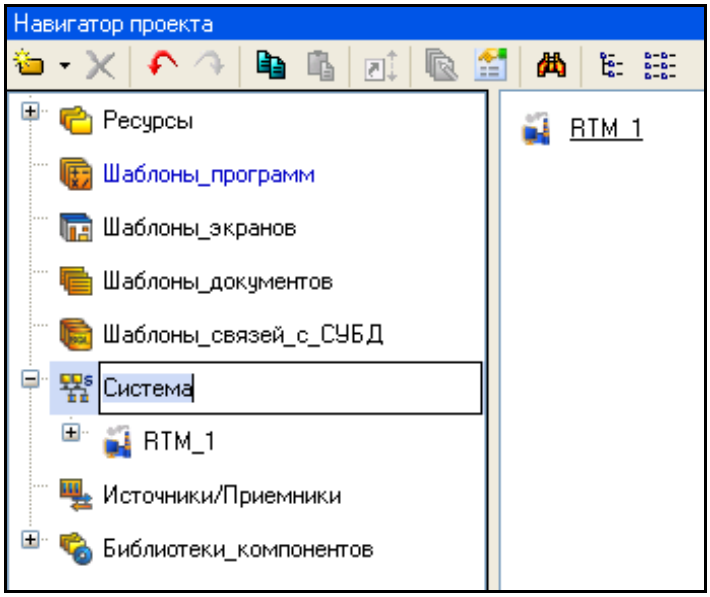


Рис. 45. Узлы проекта в навигаторе проекта

Переименуем его в **ЦППН-8.**

Внутри узла проекта **ЦППН-8** присутствует группа каналов **Каналы**, что показано на рис. 46.

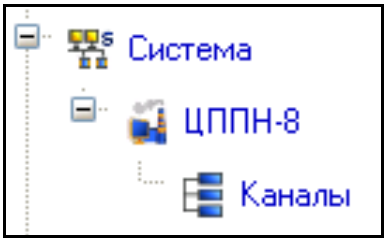


Рис. 46. Слой **Система**

Переименуем эту группу в **ЦППН-8**. В этой группе каналов создадим еще несколько групп для каждого структурного элемента ЦППН-8, как показано на рис. 47.

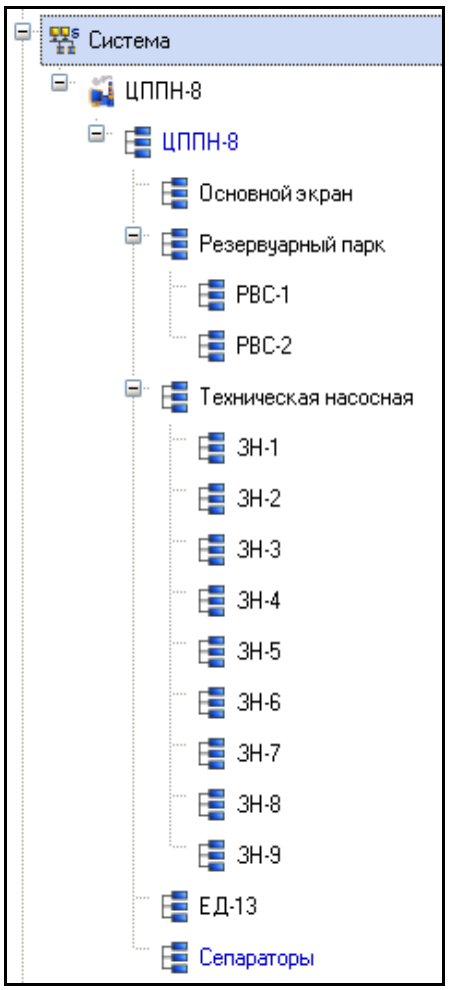


Рис. 47. Группы каналов для узла ЦППН-8

С помощью иконки создадим дополнительное окно навигатора проекта и откроем в верхнем окне слой **Шаблоны_экранов**, а в нижнем – группу компонентов **ЦППН-8** вновь созданного узла АРМ **ЦППН-8**, как показано на рис. 48.

Затем, выделяя ЛК шаблоны экранов и удерживая ЛК, перетащим их в группы узла **ЦППН-8** методом **drag-and-drop**. Результат выполнения операции показан на рис. 49.

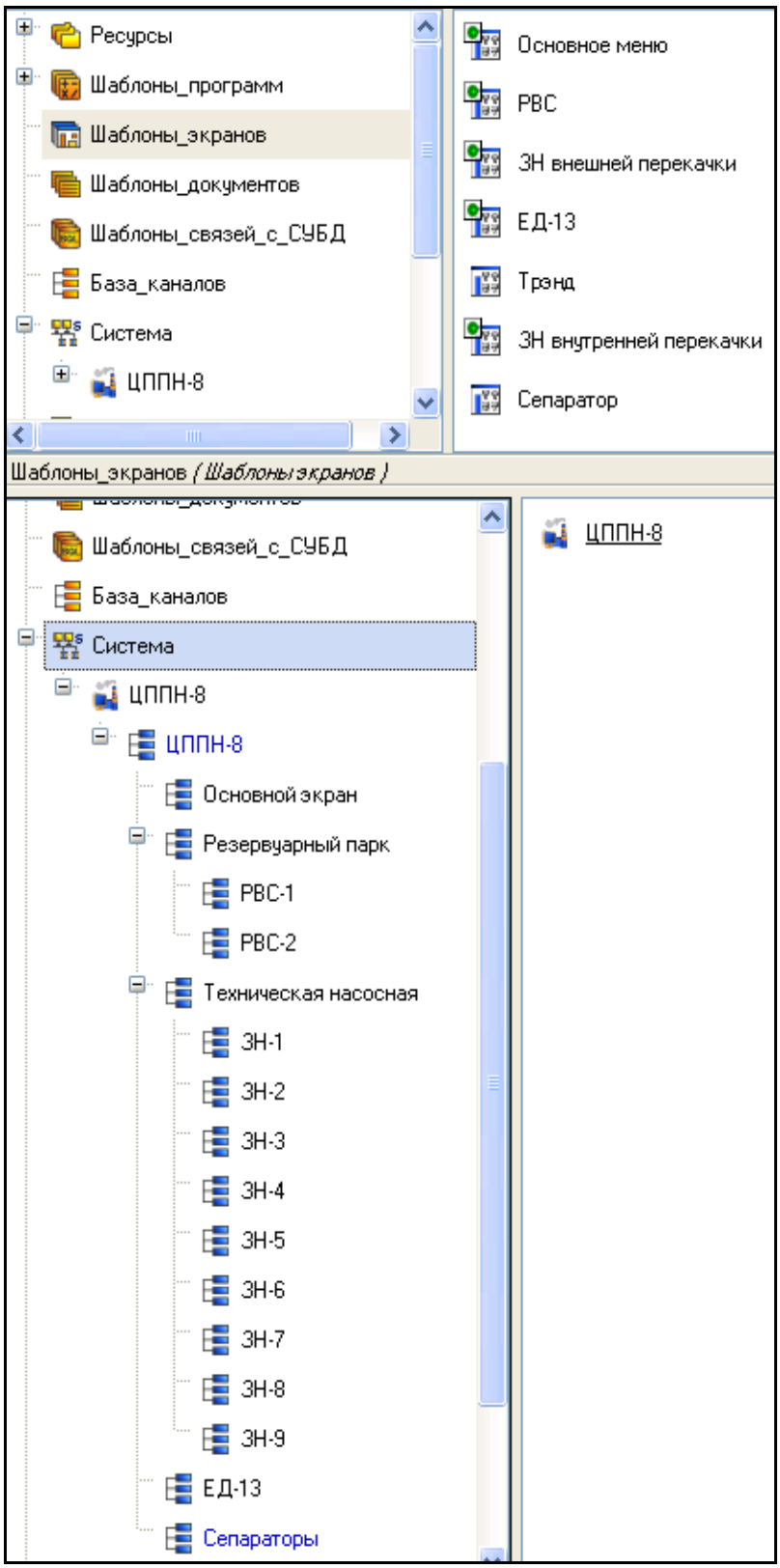


Рис. 48. Узел **RTM_1** с созданными группами

Следующим шагом разработки проекта создадим каналы по аргументам разработанных шаблонов экранов. Для этого войдем в группу каналов АРМ – узла ЦПП-8 РВС-1 и вызовем свойства канала класса Вызов РВС-1:1, как показано на рис. 50.

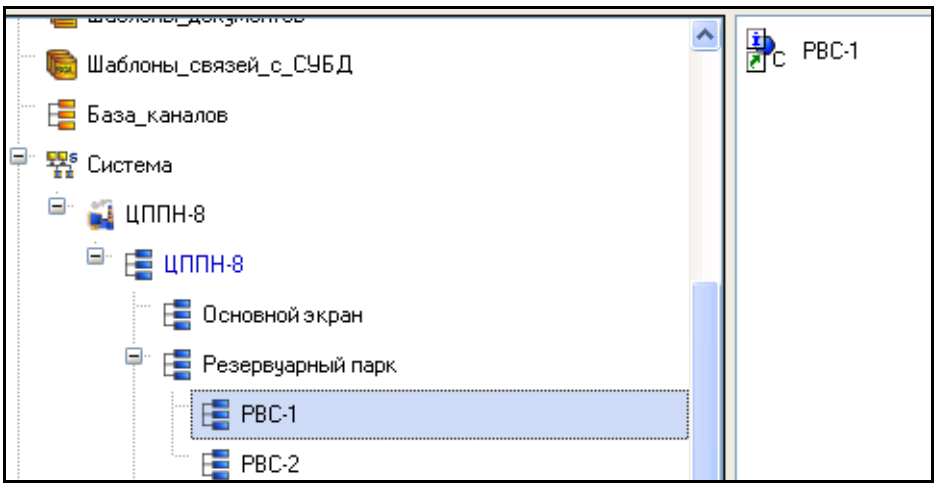


Рис. 49. Созданный экран в группе **РВС-1**

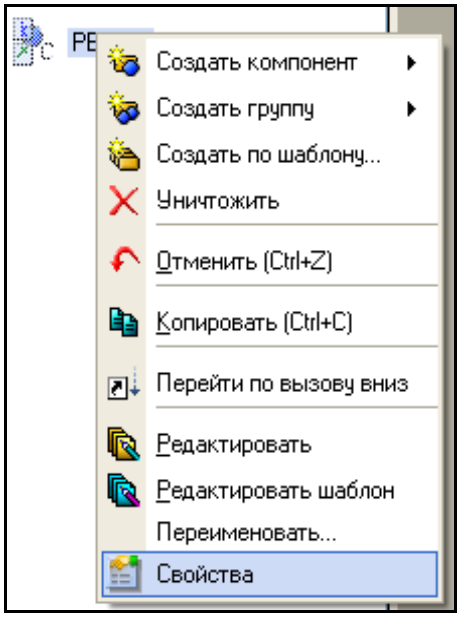
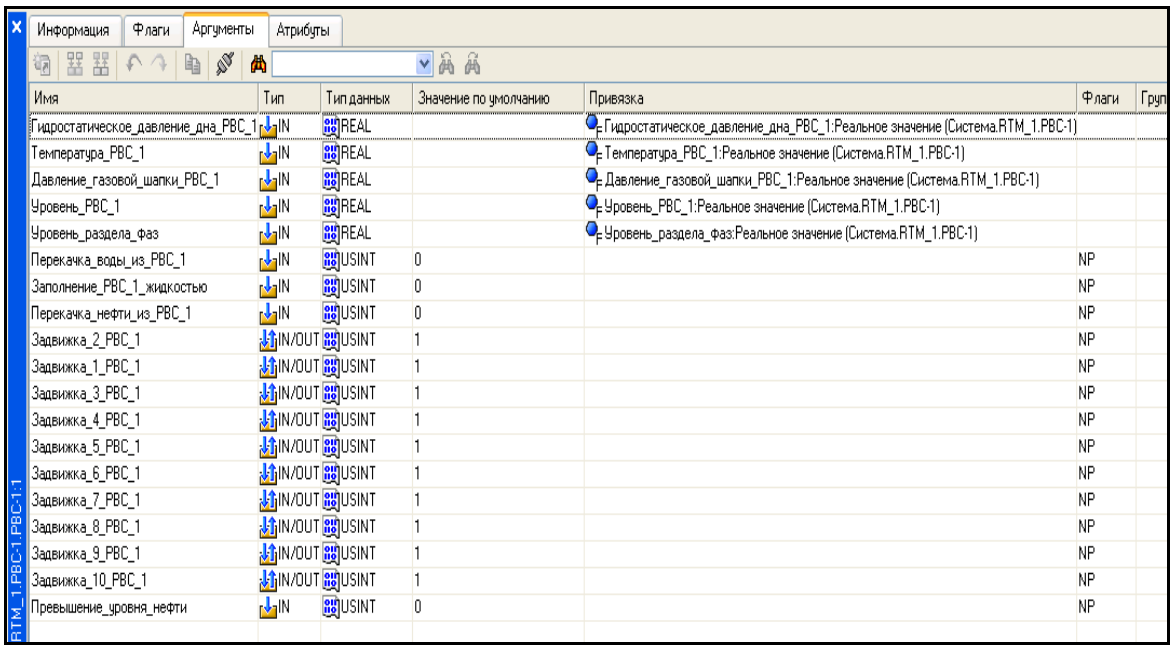


Рис. 50. Вызов контекстного меню экрана **РВС-1:1**

Перейдем во вкладку **Аргументы**, выделим ЛК первый аргумент и с помощью щелчка ЛК мыши на иконке создадим каналы в выбранной группе и автоматически свяжем их атрибуты с аргументами шаблона экрана. Данная операция называется автопостроением каналов. Автопостроение подробно описано в приложении Г. Результат его выполнения представлен на рис. 51.

До задания информационных потоков между узлами проекта проведем настройку архива и отчета тревог в АРМ.

Следующим шагом будет создание программ по шаблонам. Для этого аналогично созданию экранов по шаблону необходимо открыть два окна навигатора. В одном из них надо выбрать слой **Шаблоны_программы**, в другом - необходимую группу каналов, как показано на рис. 52.



Имя	Тип	Тип данных	Значение по умолчанию	Привязка	Флаги	Груп
Гидростатическое_давление_дна_PBC_1	IN	REAL		Гидростатическое_давление_дна_PBC_1:Реальное значение (Система.RTM_1.PBC-1)		
Температура_PBC_1	IN	REAL		Температура_PBC_1:Реальное значение (Система.RTM_1.PBC-1)		
Давление_газовой_шапки_PBC_1	IN	REAL		Давление_газовой_шапки_PBC_1:Реальное значение (Система.RTM_1.PBC-1)		
Уровень_PBC_1	IN	REAL		Уровень_PBC_1:Реальное значение (Система.RTM_1.PBC-1)		
Уровень_раздела_фаз	IN	REAL		Уровень_раздела_фаз:Реальное значение (Система.RTM_1.PBC-1)		
Перекачка_воды_из_PBC_1	IN	USINT	0		NP	
Заполнение_PBC_1_жидкостью	IN	USINT	0		NP	
Перекачка_нефти_из_PBC_1	IN	USINT	0		NP	
Задвижка_2_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_1_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_3_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_4_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_5_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_6_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_7_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_8_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_9_PBC_1	IN/OUT	USINT	1		NP	
Задвижка_10_PBC_1	IN/OUT	USINT	1		NP	
Превышение_уровня_нефти	IN	USINT	0		NP	

Рис. 51. Аргументы экрана **PBC-1**

Затем, выделяя ЛК шаблоны программ и удерживая ЛК, перетащим их в необходимые группы узла **ЦППН-8**. Результат выполнения операции представлен на рис. 53.

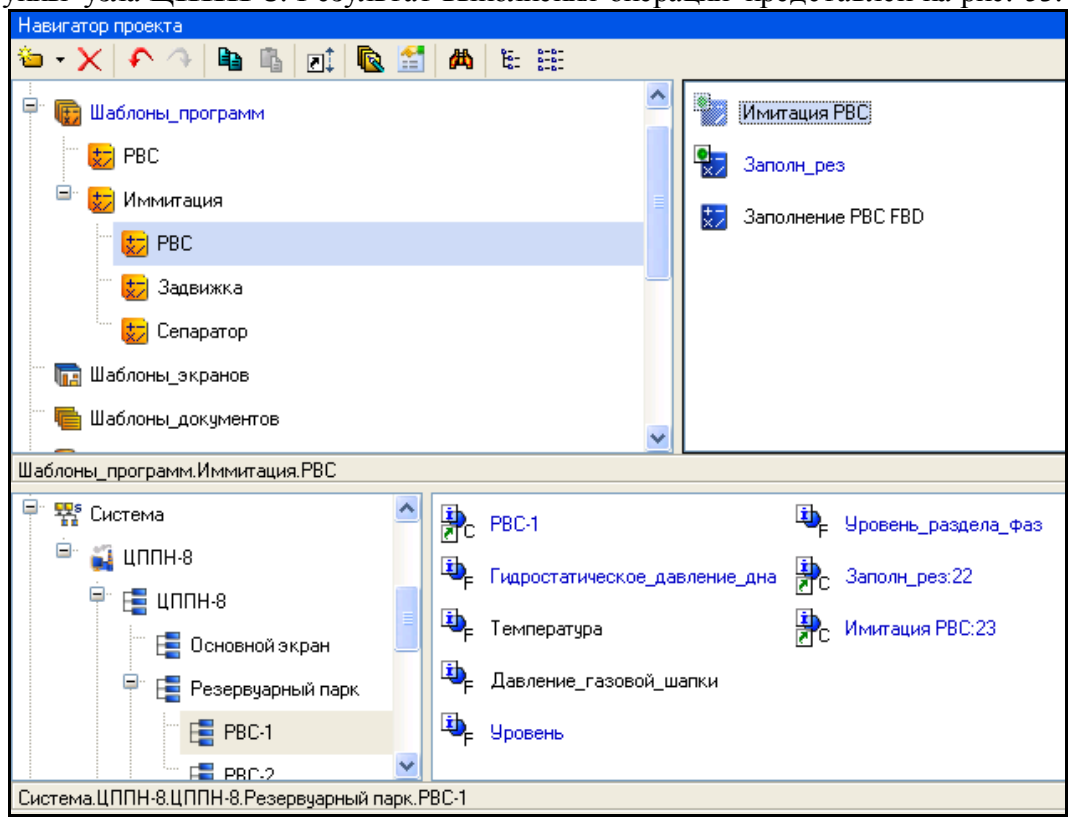


Рис. 52. Окно навигатора до выполнения операции

Затем для каждой программы выполним привязку атрибутов каналов к аргументам.

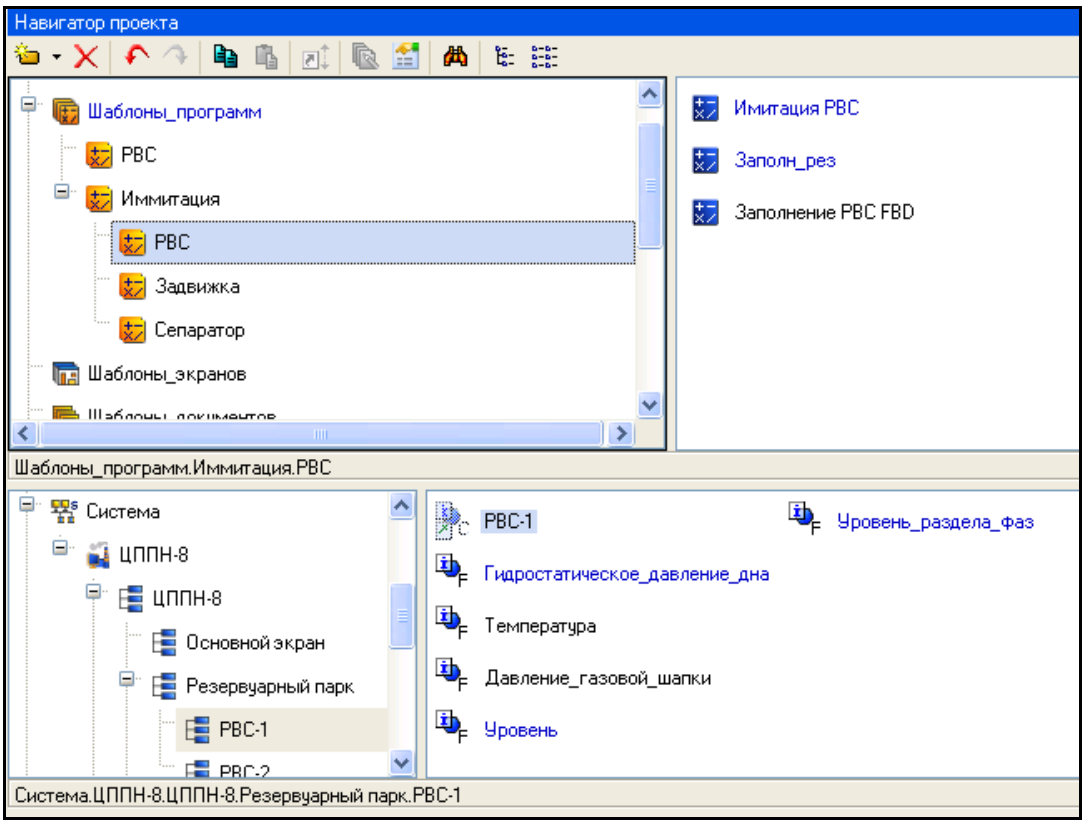


Рис. 53. Окно навигатора после выполнения операции

Для этого войдем в группу каналов АРМ – узла **ЦППН-8 РВС-1** и вызовем свойства канала класса **Вызов «Автоматизация_РВС:15»**, как показано на рис. 54.

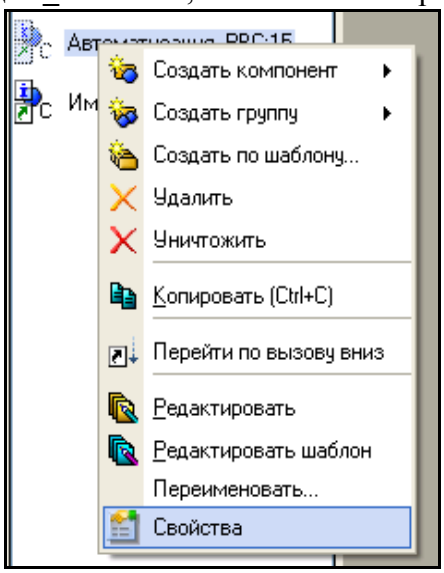


Рис. 54. Вызов контекстного меню программы «Автоматизация_РВС:15»

Перейдем во вкладку **Аргументы**. Затем двойным щелчком ЛК мыши в столбце «Привязка» в поле аргумента вызовем диалоговое окно конфигурирования связи. Диалоговое окно показано на рис. 55.

В качестве привязки может быть выбран атрибут канала либо аргумент канала. Аргументы присутствуют только у каналов класса **CALL** (к примеру у канала, ссылающегося на шаблон экрана или программы). В качестве атрибута канала по умолчанию устанавливается атрибут **Реальное значение.**

В данном диалоговом окне выбираем необходимый канал. В данном случае для аргумента **Уровень_РВС** выбираем канал **Уровень.**

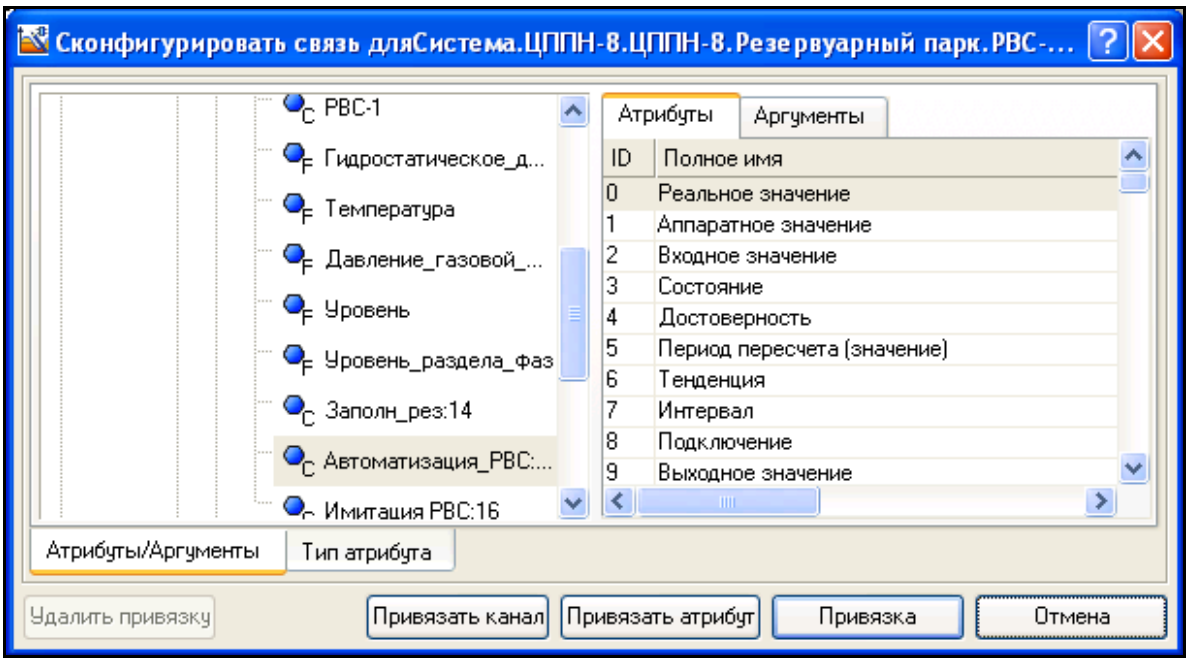


Рис. 55. Диалоговое окно конфигурирования привязки

Контрольные вопросы

1. Общие сведения о языках программирования
2. Международный стандарт IEC 61131-3
3. Этапы разработки шаблона програм

Лабораторная работа №9 Разработка АСУТП в среде SCADA системы TRACE MODE 6

Цель работы

1. Изучить основные понятия, структуру и назначение отдельных элементов SCADA системы TRACE MODE 6

2. Изучить порядок работы по созданию АСУ ТП с помощью редакторов SCADA системы TRACE MODE 6.

3. Разработать проект учебной АСУ ТП в SCADA системы TRACE MODE 6.

Методика выполнения работы

1. Изучить теоретические сведения о SCADA системе TRACE MODE 6, изложенные ниже.

2. Изучить основные действия по созданию и редактированию проектов автоматизации в соответствии с выданным преподавателем вариантом и указаниями, содержащимися в файле Учебник по TRACE MODE 6 Вариант X-Y. doc, где X-Y – номер выданного варианта.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. СТРУКТУРА СИСТЕМЫ

TRACE MODE 6 – это программный комплекс, предназначенный для разработки и запуска в реальном времени распределенных автоматизированных систем управления технологическими процессами (АСУТП) и решения ряда задач управления предприятием (АСУП).

Для решения задач АСУП в TRACE MODE 6 интегрирован пакет **T-FACTORY**.

Комплекс программ TRACE MODE 6 делится на 3 части:

Интегрированная среда разработки проекта (ИС) – единая программная оболочка, содержащая все необходимые средства для разработки проекта.

Под **проектом** в TRACE MODE 6 понимается вся совокупность данных и алгоритмов функционирования распределенной АСУ (АСУТП и/или T-FACTORY), заданных средствами TRACE MODE.

Итогом разработки проекта в ИС является создание файлов, содержащих необходимую информацию об алгоритмах работы АСУ. Эти файлы затем размещаются на аппаратных средствах (компьютерах и контроллерах) и выполняются под управлением исполнительных модулей TRACE MODE.

В интегрированную среду разработки TRACE MODE 6 встроены более десяти редакторов, автоматически открывающихся при вызове того или иного компонента проекта. Среди них:

- редактор графических экранных форм;
- редактор программ на визуальном языке Techno FBD;
- редактор программ на визуальном языке Techno SFC;
- редактор программ на визуальном языке Techno LD;
- редактор программ на процедурном языке Techno ST;
- редактор программ на процедурном языке Techno IL;
- редактор шаблонов документов;
- редактор SQL-запросов;
- редактор паспортов оборудования (EAM);
- редактор персонала (HRM);
- редактор материальных ресурсов (MES).

Исполнительные модули (мониторы, MPB) – программные модули различного назначения, под управлением которых в реальном времени выполняются составные части проекта, размещаемые на отдельных компьютерах или в контроллерах.

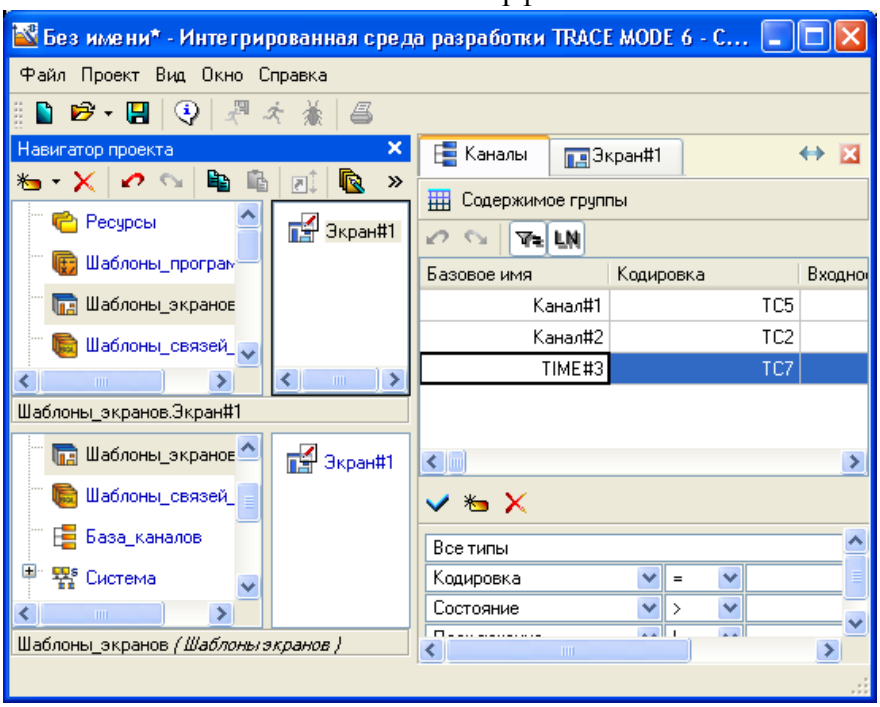
Составная часть проекта, размещаемая на отдельном компьютере или в контроллере и выполняемая под управлением одного или нескольких исполнительных модулей TRACE MODE, называется **узлом проекта**.

В общем случае размещение узла на том же аппаратном средстве, на котором он должен исполняться под управлением монитора, не является обязательным – мониторы могут загружать узлы с удаленных аппаратных средств.

Драйверы обмена – драйверы, используемые мониторами TRACE MODE для взаимодействия с устройствами, протоколы обмена с которыми не встроены в мониторы.

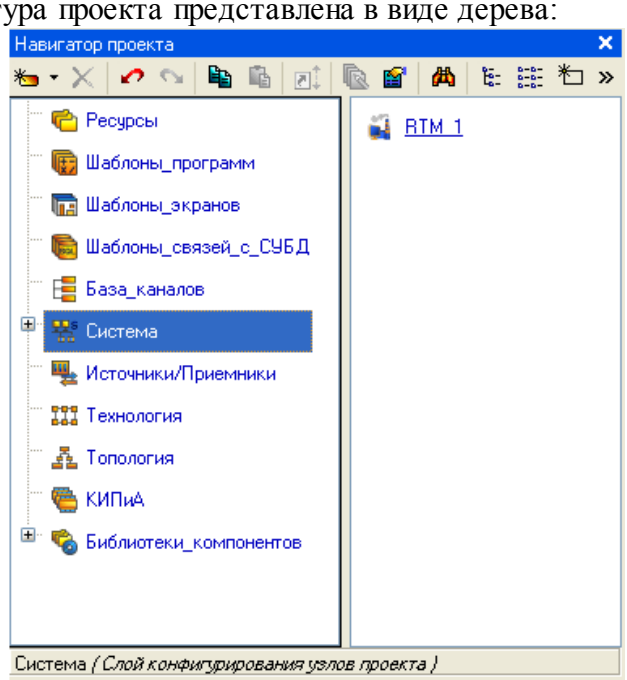
Принципы разработки проекта в ИС

ИС объединяет в единой оболочке **навигатор** и набор редакторов для создания всех составляющих проекта. ИС имеет многооконный интерфейс:



В ИС поддерживаются стандартные операции изменения размеров и перемещения окон.

В навигаторе структура проекта представлена в виде дерева:



Корневые группы этого дерева (**слои**) предопределены и создаются автоматически при создании нового проекта (слои отображаются в левом окне навигатора). Элементарные

структурные составляющие (листья структурного дерева) называются **компонентами** проекта. Например, компонентами проекта являются: канал; канал, вызывающий шаблон; шаблон; источник данных и т.д.

Группы компонентов, которые могут быть созданы в структуре проекта, предназначены для структурирования проекта.

В ряде случаев группы имеют и другой, вполне определенный, смысл – например, узлы проекта создаются как корневые группы слоя **Система.**

В правом окне навигатора отображается содержимое слоя (группы), выделенной в левом окне, – таким образом, компоненты проекта могут быть отображены только в правом окне.

Структура проекта редактируется в навигаторе с помощью команд меню **Проект**, контекстного меню и панелей инструментов, а также с помощью метода drag-and-drop.

Технология разработки проекта в ИС

Разработка проекта в ИС включает следующие процедуры:

- создание структуры проекта в навигаторе;
- конфигурирование или разработка структурных составляющих – например, разработка шаблонов графических экранов оператора, разработка шаблонов программ, описание источников/приемников и т.д.;
- конфигурирование информационных потоков;
- выбор аппаратных средств АСУ (компьютеров, контроллеров и т.п.);
- создание узлов в слое **Система** и их конфигурирование;
- распределение каналов, созданных в различных слоях структуры, по узлам и конфигурирование интерфейсов взаимодействия компонентов в информационных потоках;
- сохранение проекта в единый файл для последующего редактирования (с помощью команды **Сохранить** или **Сохранить как**);
- экспорт узлов в наборы файлов для последующего запуска под управлением мониторов TRACE MODE (по команде  **Сохранить для MPB**).

Перечисленные процедуры (за исключением двух заключительных) и входящие в их состав операции могут выполняться в произвольном порядке. Например, можно начинать разработку проекта с разработки шаблонов графических экранов оператора, с создания узлов и их каналов в слое **Система** (если аппаратные средства АСУ известны заранее), можно конфигурировать каналы и информационные потоки после распределения каналов по узлам и т.п.

Классификация компонентов

По функциональному назначению компоненты проекта относятся к одному из следующих видов:

- **каналы** – компоненты, определяющие алгоритм работы. Каналы могут создаваться в различных слоях, однако их окончательное распределение по узлам в слое **Система** обязательно – в противном случае они не будут экспортированы для MPB;
- **шаблоны** – компоненты, которые при работе в реальном времени могут вызываться каналами с передачей параметров. Передача параметров настраивается при разработке проекта в ИС посредством привязки **аргументов** шаблона к каналам или источникам/приемникам;
- **источники/приемники** – компоненты, тем или иным способом описывающие внутренние переменные различных устройств или приложений, с которыми требуется обмениваться данными. Под устройствами здесь понимаются контроллеры, а также внешние и внутренние модули/платы различного назначения, обмен с которыми поддерживается мониторами TRACE MODE (в том числе через драйверы). Системные переменные TRACE MODE также создаются в ИС как источники/приемники. Источники/приемники являются шаблонами каналов;
- **наборы ресурсов** – наборы текстов, изображений и видеоклипов, которые могут быть использованы при разработке шаблонов графических экранов;
- **графические объекты** – компоненты, представляющие собой в общем случае несколько графических элементов (из имеющихся в редакторе представления данных), сгруппированных в

один. Графические объекты могут быть использованы при разработке шаблонов графических экранов;

- **последовательные порты** – параметры COM-портов;
- **словари сообщений** – наборы сообщений, генерируемых при возникновении различных событий;
- **клеммы** – эти компоненты, описывающие электрические контакты (например, монтажных шкафов), являются элементами схемы электрических соединений АСУ.

Классификация слоев

Предопределенные слои структуры проекта имеют следующее назначение:

- **Ресурсы** – для создания пользовательских наборов текстов, изображений и видеоклипов, а также графических объектов;
- **Шаблоны программ** – для создания шаблонов программ;
- **Шаблоны экранов** – для создания шаблонов графических экранов;
- **Шаблоны связей с БД** – для создания шаблонов связей с базами данных;
- **Шаблоны документов** – для создания шаблонов документов (отчетов);
- **База каналов** – этот слой является хранилищем всех каналов проекта. Выполнять операции с каналами (в том числе создавать их) можно в различных слоях, однако во всех случаях эти операции на самом деле реализуются в слое **База каналов**. В любом другом слое, где выполняется команда для совершения операции с каналом, ее результат только отображается – поэтому существуют команды удаления и уничтожения каналов. В слое **База каналов** можно начинать разработку проекта;
- **Система** – для конфигурирования узлов и их составляющих (узел создается как корневая группа этого слоя);
- **Источники/приемники** – для создания описаний источников/приемников в различных устройствах и программных приложениях, обмен с которыми поддерживается мониторами, а также для конфигурирования системных переменных TRACE MODE 6,
- **Технология** – для разработки проекта от технологии (т.е. с группировкой компонентов по признаку их принадлежности к технологическому объекту). При отладке проекта слой **Технология** может играть роль узла – для него определена команда **Сохранить узел для MPB**. Кроме того, для этого слоя определены команды взаимодействия с технологической базой данных;
- **Топология** – для разработки проекта от топологии (т.е. с группировкой компонентов по месту расположения);
- **КИПиА** – для описания электрических соединений АСУ;
- **Библиотеки компонентов** – для создания библиотек **объектов** – проектных решений отдельных задач. Этот слой содержит предопределенные группы **Системные** и **Пользовательские**. В группе **Системные** содержатся библиотеки, подключенные к ИС по умолчанию.

Классификация узлов

Узлы проекта создаются как корневые группы слоя **Система**. Предопределенное название узла указывает на семейство мониторов, для которых данный узел предназначен. Узел может содержать только те компоненты, которые поддерживаются мониторами соответствующего семейства.

В общем случае, узлы могут выполняться под управлением различных мониторов.

Как правило, узел выполняется на отдельном аппаратном средстве. В случае запуска двух и более узлов на одном аппаратном средстве оно должно быть оборудовано соответствующим количеством сетевых карт.

Параметры узлов задаются в соответствующем редакторе.

RTM

Узел **RTM** предназначен для запуска на компьютере под управлением исполнительных модулей семейства **RTM** (MPB) – мониторов с поддержкой отображения графических экранов

оператора, поддержкой обмена по последовательному интерфейсу и сети с различным оборудованием и выполняющего пересчет каналов всех классов, кроме каналов T-FACTORY.

T-FACTORY

Узел **T-FACTORY** предназначен для запуска на компьютере под управлением исполнительных модулей семейства **T-FACTORY** – мониторов для решения задач АСУП.

MicroRTM

Узел **MicroRTM** предназначен для запуска на компьютере или в контроллере под управлением исполнительных модулей семейства **Micro RTM.** Основное отличие этих мониторов от МРВ – отсутствие поддержки отображения графических экранов.

Logger

Узел **Logger** предназначен для запуска на компьютере под управлением исполнительного модуля **Logger** (регистратор) – монитора, способного вести архивы по каналам всех узлов проекта.

EmbeddedRTM

Узел **EmbeddedRTM** предназначен для запуска на компьютере или в контроллере под управлением исполнительных модулей семейства **Embedded RTM** – мониторов с поддержкой мнемосхем, поддержкой обмена с оборудованием по различным протоколам и выполняющего пересчет каналов.

NanoRTM

Узел **NanoRTM** предназначен для запуска в контроллере под управлением исполнительного модуля **Nano RTM** – монитора, аналогичного **Micro RTM**, но предназначенного для работы с малым числом каналов.

Console

Узел **Console** предназначен для запуска на компьютере под управлением исполнительных модулей, которые, в отличие от МРВ, не выполняют пересчет каналов, предназначенных для работы с данными. Консоли позволяют получать данные от других узлов проекта по сети, отображать их на графических экранах и управлять технологическим процессом из графики. Консоли не могут взаимодействовать с узлами T-FACTORY.

TFactory_Console

Узел **TFactory_Console** предназначен для запуска на компьютере под управлением исполнительных модулей, аналогичных консолям, но, кроме того, способных взаимодействовать с узлами T-FACTORY.

TM_OPC_Server

Узел **TM_OPC_Server** предназначен для запуска на компьютере под управлением OPC-сервера TRACE MODE 6.

2. Меню и панель инструментов ИС

Оболочка ИС имеет главное меню, включающее меню **Файл**, **Вид**, **Окна** и **Справка**, и панель инструментов.

Редакторы, встроенные в ИС, имеют свои меню и панели инструментов, которые при открытии этих редакторов частично или полностью добавляются к имеющимся в ИС. При открытии редактора возможно также модифицирование списка команд меню ИС.

В случае открытия нескольких редакторов, панели инструментов и меню ИС соответствуют редактору, окно которого в текущий момент является активным.

Меню и панель инструментов оболочки ИС доступны во всех случаях.

Меню ‘Файл’ и главная панель инструментов ИС

Главная панель инструментов ИС включают следующие команды:


Новый (**Ctrl-N**) – создать новый проект;


– открыть проект (файл с расширением **prj**):


Открыть (**Ctrl-O**) – выбрать файл в стандартном диалоге операционной системы;

– выбрать файл из списка последних открытых;

Импорт – по этой команде открывается меню, содержащее следующие команды:

Импорт из версии 5 – открыть проект, разработанный в TRACE MODE 5. По этой команде открывается окно выбора файла ***.ctm**, и выбранный проект конвертируется в TRACE MODE 6. Операция конвертирования зависит от флага **Отключить конвертирование графической базы**;

Сохранить (Ctrl-S) – сохранить проект в файл **prj** с тем же именем (пользовательские библиотеки компонентов сохраняются по этой команде в файл **tmdevenv.tmul**);

Сохранить как (Ctrl-Shift-S) – сохранить проект в файл **prj** с заданием его имени (пользовательские библиотеки компонентов сохраняются по этой команде в файл **tmdevenv.tmul**);

Информация о проекте – открыть одноименный диалог (эта команда доступна также из меню **Проект** и контекстного меню навигатора проекта).

В этом диалоге можно указать автора проекта, организацию и комментарий к проекту. В диалоге индицируется время создания и время последнего изменения проекта. В нижней части диалога выводится информация об основных видах компонентов проекта (максимально возможное число компонентов данного вида в проекте, использованное число компонентов данного вида и число компонентов данного вида, оставшееся в распоряжении разработчика);

Сохранить для MPB – экспортировать узлы для последующего запуска под управлением исполнительных модулей TRACE MODE. Экспорт одного узла возможен из меню **Проект** и контекстного меню навигатора проекта;

Отладка – загрузить выделенный узел (слой **Технология**) в профайлер. Эта команда доступна после выполнения команды **Сохранить для MPB** или **Сохранить узел для MPB**;

Шпион – получить в ИС реальные данные с работающих узлов;

Печать – документировать проект в файл;

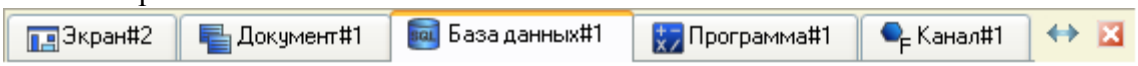
Настройки ИС – отрыть диалог задания общих настроек ИС и редакторов шаблонов;

Последние файлы – показать список последних файлов, открытых в ИС. Выбранный в списке файл загружается в ИС;

Выход – выйти из интегрированной среды.

Панель переходов между окнами ИС

Для переходов между открытыми окнами редакторов используется панель, показанная на рисунке ниже. Оранжевая полоска обозначает активное окно.



Для перехода в нужное окно нужно нажать ЛК на соответствующей вкладке этой панели, при этом выбранное окно становится активным.

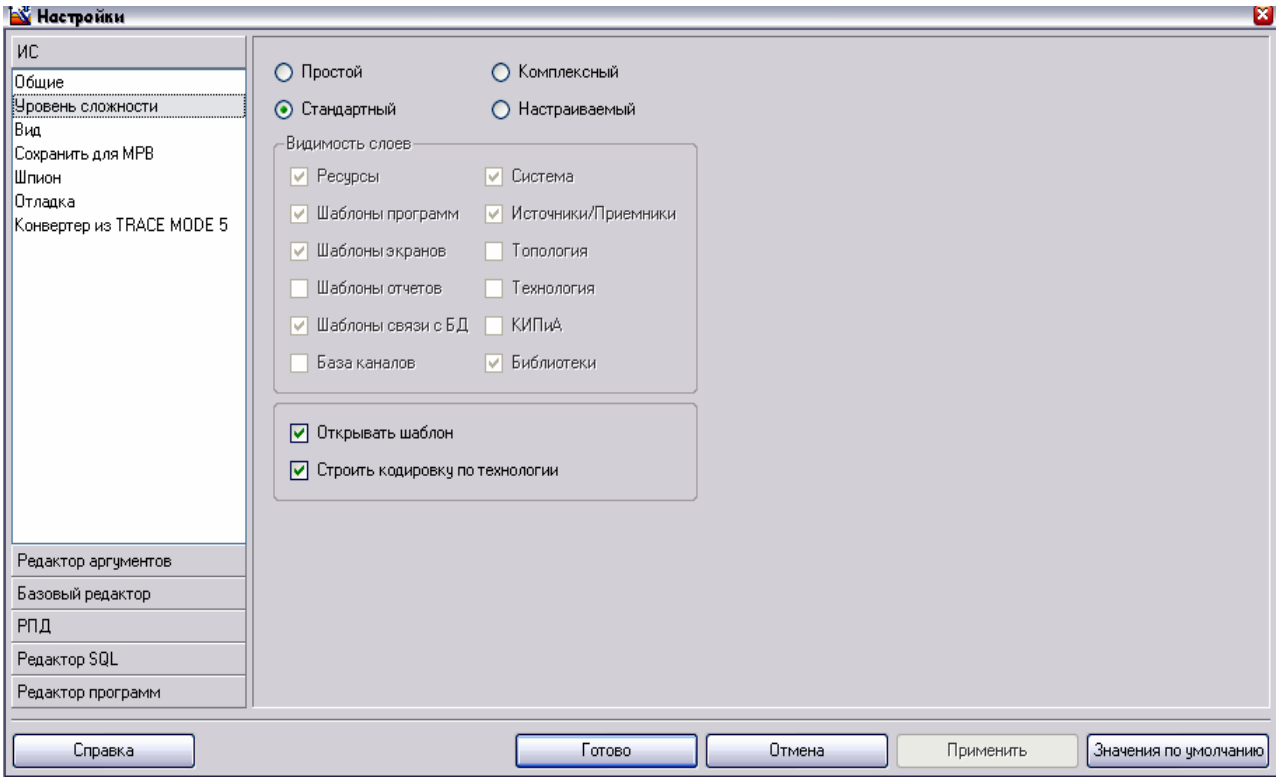
Панель содержит также следующие инструменты:

– синхронизировать с деревом проекта (по этой команде в навигаторе проекта будет выделен компонент, окно редактора которого активно, а также слой (группа), содержащая этот компонент);

– закрыть активное окно.

Задание общих настроек ИС

Для задания общих настроек ИС и редакторов шаблонов предназначен диалог, который открывается по команде **Настройки ИС** меню **Файл**:



Настройки ИС задаются на одноименной вкладке этого диалога, набор инструментов которой изменяется в зависимости от выбранного в левом списке раздела, и вкладке **Базовый редактор**.

Вкладка ‘Интегрированная среда разработки’

Раздел ’Общие’

При выборе этого раздела вкладка содержит следующие инструменты:

- переключатели режима отображения редакторов компонентов – **Открывать редакторы как вкладки**, **MDI** и **SDI**. В первом режиме каждый открытый редактор занимает всю рабочую область ИС и изменение размеров окна редактора недоступно.

В режиме **MDI** размеры окна каждого редактора могут быть изменены, при этом доступны стандартные инструменты окон (– открыть меню, содержащее стандартные команды для работы с окном, – минимизировать окно, – восстановить окно, – закрыть окно).

В режиме **SDI** при выделении объекта структуры в навигаторе соответствующий редактор открывается автоматически, занимая всю рабочую область ИС.

Раздел ’Уровень сложности’

При выборе этого раздела вкладка содержит переключатели уровня сложности проекта (задают отображаемый набор слоев):

- **простой** – отображаются слои **Ресурсы**, **Система**, **Источники/Приемники** и **Библиотеки компонентов**;
- **стандартный** (значение по умолчанию) – отображаются те же слои, что и для простого уровня, и все слои шаблонов (экранов, программ, связей с БД и документов);
- **комплексный** – отображаются все слои, кроме слоя **База каналов**;
- **настраиваемый** – при выборе этого уровня в диалоге доступны переключатели отображения всех слоев, включая слой **База каналов**.

Раздел ’Сохранить для MPB’

При выборе этого раздела вкладка содержит следующие инструменты:

флаги, определяющие степень детализации информации, которая выводится конвертером **FileCnv32.dll** в текстовые файлы **<имя файла prj>_<ordinal>.cnv** при выполнении команды **Сохранить для MPB**:

- **Каналы**;

- **Шаблоны**;
- **Группы**;
- **Ресурсы**;
- **Создать стандартные объекты** – если этот флаг установлен, каналы узла отображаются в профайлере сгруппированными в стандартные объекты (группы) TRACE MODE. Эта группировка имеет значение только для отладки;
- **Подробная информация**.
- Список **Глубина отслеживания источников**, с помощью которого задается порядковый номер источника/приемника в цепочках связей **канал – источник/приемник1 – источник/приемник2**, который будет использован при экспорте узла

Раздел 'Отладка'

При выборе этого раздела вкладка содержит переключатели типа профайлера, который будет запущен по команде **Отладка** (только для отладки узлов **RTM**, **Console**, **T-Factory** и **T-Factory_Console**):

- **Профайлер с поддержкой графических экранов** ;
- **Профайлер без поддержки графических экранов** .

Раздел 'Конвертер из предыдущей версии'

При выборе этого раздела вкладка содержит флаг **Отключить конвертирование графической базы**. Установка этого флага отключает конвертирование графической базы проекта при его импорте в TRACE MODE 6.

Сохранение проекта для редактирования

По команде **Сохранить** (**Ctrl-S**) или **Сохранить как** (**Ctrl-Shift-S**) из меню **Файл** проект сохраняется в бинарный файл с расширением **prj** (в директории сервера проекта) для последующего редактирования в ИС.

В ИС предусмотрено резервирование предыдущей версии файлов **prj** и **tmul** – при повторном выполнении команды **Сохранить** расширение файлов, сохраненных ранее, изменяются соответственно на **~prj** и **~tmul**.

В ИС можно загрузить один проект.

На одном компьютере можно запустить несколько ИС и загрузить в них разные проекты.

Сохранение проекта для запуска

По команде **Сохранить для MPB** меню **Файл** или панели инструментов ИС все узлы экспортируются в наборы файлов для их последующего копирования на аппаратные средства, на которых они должны исполняться под управлением мониторов TRACE MODE.

При выполнении команды **Сохранить для MPB** в директории сервера проекта создается поддиректория **<имя файла prj без расширения>**, в которой для каждого узла создается папка с набором файлов. Папка узла имеет имя **<name>_<ordinal + 1>**, где **name** – имя, заданное для узла при его конфигурировании в ИС, **ordinal** – порядковый номер узла в слое **Система**, начиная с 0 (порядковый номер узла в слое не следует путать с его индивидуальным номером).

Необходимым условием экспорта узла является наличие в нем хотя бы одного канала.

По команде **Сохранить узел для MPB** из меню **Проект** или контекстного меню навигатора выделенный узел (или слой **Технология**) экспортируется в произвольную папку, при этом при повторном экспорте резервные копии узла не создаются.

3. Редактирование структуры проекта

Навигатор имеет следующие средства для редактирования структуры проекта:

- меню **Проект**;
- панели инструментов;
- контекстное меню.

Кроме того, в навигаторе поддерживается метод перетаскивания объектов мышью (метод drag-and-drop), а также его модификации (drag-and-drop с удержанием служебных клавиш).

Для конфигурации/разработки объектов структуры в навигаторе предусмотрены команды **Свойства** и **Редактировать**, с помощью которых для каждого объекта структурного дерева могут быть открыты соответствующие **окно свойств** и **редактор**.

Меню и главная панель инструментов навигатора проекта

Меню **Проект**, главная панель инструментов и контекстное меню навигатора проекта содержат набор команд, который соответствует выделенному объекту структурного дерева. Для выделения объекта нужно нажать на нем ЛК. Групповое выделение объектов в навигаторе не поддерживается.

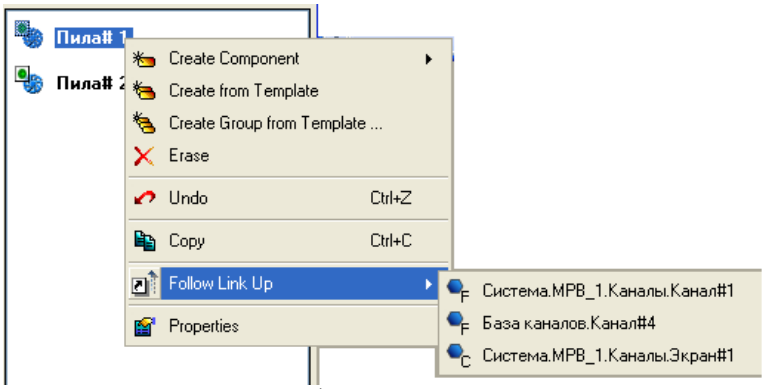
Меню **Проект**, главная панель инструментов и контекстное меню навигатора содержат как типовые команды для создания компонентов (групп компонентов), работы с буфером обмена и поиска, так и специфические:

 – такой вид приобретает типовой инструмент удаления  при выделении канала. При нажатии стрелки открывается меню, содержащее команды **Удалить** (удалить без удаления из слоя **База каналов**) и **Уничтожить** (удалить с удалением из слоя **База каналов**);

 – такой вид приобретает в некоторых случаях типовой инструмент вставки  . При нажатии стрелки открывается дополнительное меню, содержащее команды  **Вставить** и  **Вставить с привязкой**;

 **Перейти по ссылке вниз/вверх** – по этой команде открывается дополнительное окно навигатора, в котором:

- выделяется компонент, с которым связан данный компонент или который вызывается данным компонентом (в случае перехода по ссылке вниз);
- выделяется компонент, связанный с данным компонентом или вызывающий данный компонент (в случае перехода по ссылке вверх). Если подобных компонентов несколько, один из них выбирается в списке:



Если у компонента имеются связи/вызовы как вниз, так и вверх, данный инструмент принимает вид  , и при нажатии стрелки открывается дополнительное меню, содержащее команды **Перейти по ссылке вниз** и **Перейти по ссылке вверх**;

 **Резервирование** – создать резервные узлы для выделенного узла:

- **Нет** – не создавать (значение по умолчанию);
- **Дублированный** – создать один резервный узел для выделенного узла;
- **Троированный** – создать два резервных узла для выделенного узла;

 **Редактировать** – открыть выделенный объект структурного дерева в соответствующем **редакторе**;

Редактировать шаблон – открыть шаблон, вызываемый данным компонентом, в соответствующем редакторе;

Переименовать – перейти к редактированию имени выделенного объекта структуры. Для перехода к редактированию имени выделенного объекта можно также нажать на нем ЛК;

Свойства – открыть **окно свойств** объекта структурного дерева;

Сохранить узел для МРВ – экспортировать выделенный узел (слой **Технология**). По этой команде на экране отображается диалог выбора места расположения файлов узла (с помощью кнопки **Создать папку** и команд контекстного меню этого диалога возможно управление файловой структурой);

Загрузить дамп узла – восстановить содержимое и конфигурацию компонентов узла из его файла восстановления. Этот файл не следует путать с файлом **<имя_узла>.dump**, который создается по умолчанию при экспорте узлов;

Загрузить в контроллер – копирование узла в контроллер;

Информация о проекте – открыть одноименный диалог. Эта команда может быть выполнена также из контекстного меню навигатора проекта и с помощью панели инструментов ИС;

Импорт из БД и **Экспорт в БД** – эти команды, предназначенные для взаимодействия с технологической базой данных, доступны при выделении слоя **Технология**.

Создание объектов структуры

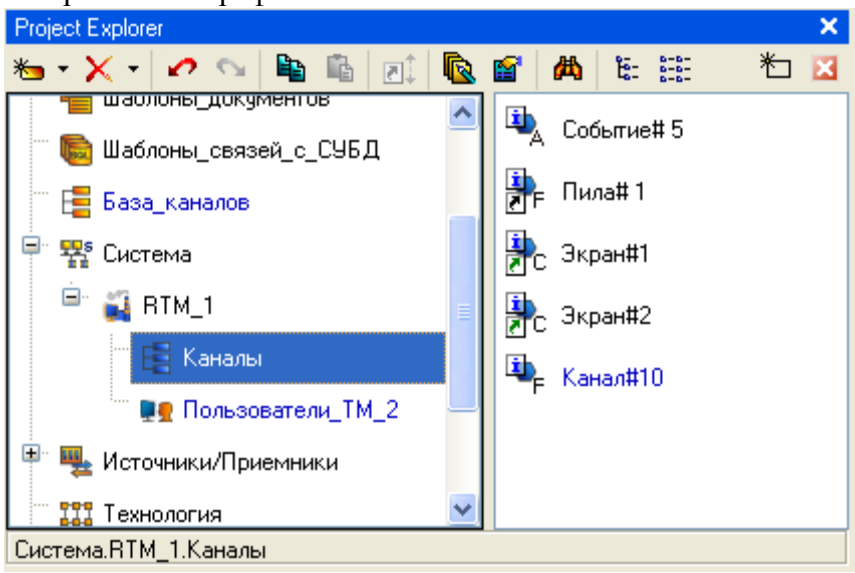
Для создания объектов структуры (компонентов и групп компонентов) используются типовые команды меню **Проект**, контекстного меню и панели инструментов навигатора .

При создании канала класса CALL с предустановленным свойством **вызов** в соответствующем слое шаблонов создается шаблон, вызываемый каналом. В навигаторе могут быть созданы следующие каналы класса CALL с предустановленным свойством **вызов**:

- **Экран** – канал с вызовом шаблона экрана;
- **Программа** – канал с вызовом шаблона программы;
- **Документ** – канал с вызовом шаблона документа;
- **Связь с БД** – канал с вызовом связи с базой данных.

Меню **Проект**, контекстное меню и панель инструментов навигатора содержат команды создания только тех объектов, которые может содержать выделенный слой/группа.

При редактировании сохраненного проекта (в том числе после выполнения команды **Сохранить/Сохранить как**) вновь созданные структурные объекты и объекты, их содержащие, выделяются в навигаторе синим шрифтом:



Удаление объекта структуры

Для удаления выделенного объекта структуры (компонента или группы компонентов) используется типовая команда **Удалить.**

При удалении канала (группы каналов) из любого слоя, кроме слоя **База каналов,** доступны две команды – **Удалить** (удалить без удаления из слоя **База каналов**) и **Уничтожить** (удалить с удалением из слоя **База каналов**). Если слой **База каналов** скрыт, команда удаления канала недоступна – в этом случае канал можно только уничтожить.

ИС автоматически удаляет привязки аргументов к компоненту, ссылки на компонент и вызовы компонента при удалении этого компонента (в случае канала – при его уничтожении).

Перемещение объектов структуры

Операция перемещения включает два действия – удаление объекта из места его начального расположения и вставку в указанную группу (слой).

Чтобы переместить объект, нужно перетащить его мышью в нужную группу (слой), удерживая клавишу **SHIFT.**

Если группа (слой), на которую указывает курсор в процессе перетаскивания, может содержать перемещаемый объект, курсор принимает вид , в противном случае – .

Копирование и вставка объекта структуры

Копирование объекта структуры в буфер обмена

Чтобы поместить копию выделенного объекта структуры в буфер обмена, нужно выполнить команду **Копировать** (**Ctrl+C**).

Если копируется слой, в буфер обмена помещается копия его содержимого (всех дочерних групп и компонентов).

Обычная вставка объекта структуры из буфера обмена

При обычной вставке создается новый объект, который является копией объекта, помещенного в буфер обмена. Операция вставки не очищает буфер обмена, поэтому после однократного копирования объекта возможна его множественная вставка.

Для обычной вставки нужно выделить группу (слой), в которой может быть создан объект того же вида, что и объект, помещенный в буфер обмена, и выполнить команду **Вставить** (**Ctrl+V**).

Копирование и вставка объекта структуры с помощью мыши

Для копирования и вставки объекта нужно перетащить его мышью в нужную группу (слой) с удержанием клавиши **CTRL**. Если группа (слой), на которую указывает курсор в процессе перетаскивания, может содержать копируемый объект, курсор принимает вид , в противном случае – . При выполнении данной операции объект не помещается в буфер обмена.

Специальная вставка объекта структуры

В навигаторе проекта поддерживается перетаскивание объектов мышью с одновременным удержанием клавиш **CTRL** и **SHIFT**. Если объект, на который указывает курсор в процессе перетаскивания, допускает завершение операции, курсор принимает вид , в противном случае – . Эквивалентом этой операции является копирование и специальная вставка объекта по команде **Вставить с привязкой.**

С помощью данной операции выполняются следующие действия:

- при перетаскивании канала (группы каналов) в слой (группу) – автопостроение канала (группы каналов) с настроенным свойством **связь** (каждый канал, созданный таким образом, связан с соответствующим исходным каналом и имеет тот же класс);
- при перетаскивании источника/приемника (группы источников/приемников) в слой (группу) – автопостроение канала (группы каналов) соответствующего класса с настроенным

свойством **связь** (каждый канал, созданный таким образом, связан с соответствующим источником/приемником);

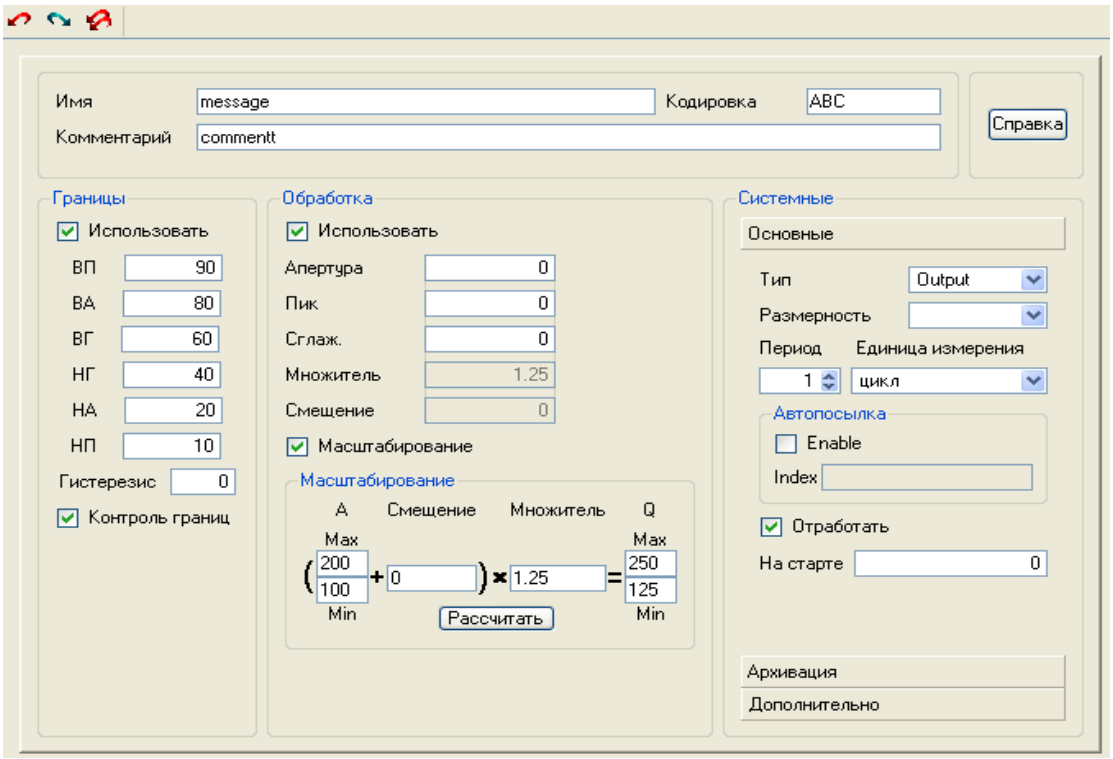
- при перетаскивании шаблона (группы шаблонов) в слой (группу) – автопостроение канала (группы каналов) класса CALL (каждый канал, созданный таким образом, настроен на вызов соответствующего шаблона и имеет такое же имя, что и шаблон);
- при перетаскивании источника/приемника или шаблона на компонент – настройка компонента на связь с источником/приемником или на вызов шаблона. В последнем случае канал класса CALL принимает имя шаблона.

Автоматический выбор вида операции вставки

При перемещении объекта структуры обычным методом drag-and-drop (без удерживания служебных клавиш) вид операции вставки (обычная или специальная) выбирается автоматически (идентифицируется по форме курсора).

Редакторы каналов

Для каждого класса канала в ИС встроен редактор (ниже показан редактор канала класса FLOAT):



Два раздела являются общими для всех редакторов каналов – верхний, содержащий кнопку вызова контекстной справки (контекстная справка вызывается также по нажатию функциональной клавиши **F1**) и поля для задания имени, комментария и кодировки канала, и раздел **Системные**, содержащий вкладки **Основные**, **Архивация** и **Дополнительно**.

Другие разделы редакторов содержит инструменты задания атрибутов, специфичных для каналов соответствующих классов.

Редакторы каналов содержат ту же панель инструментов для работы с буфером обмена, что и редактор узла.

4. Отладка проекта в ИС

Для отладки проекта ИС снабжена следующими механизмами:

- автономная отладка шаблонов;
- запуск выделенного узла (слоя **Технология**) под управлением одного из отладочных мониторов (профайлеров) из интегрированной среды по команде **Отладка** с протоколированием работы в текстовый файл;

- использование функции **шпион** – эта функция обеспечивает получение в режиме редактирования реальных данных с работающих узлов проекта. Для ее использования нужно выполнить команду  **Шпион**

Профайлеры

Для отладки узла (слоя **Технология**) его можно запустить (в том числе из ИС) под управлением одного из следующих отладочных мониторов:

- профайлера с поддержкой графических экранов (**rtc.exe**);
- профайлера без поддержки графических экранов (**rtmg32.exe**).

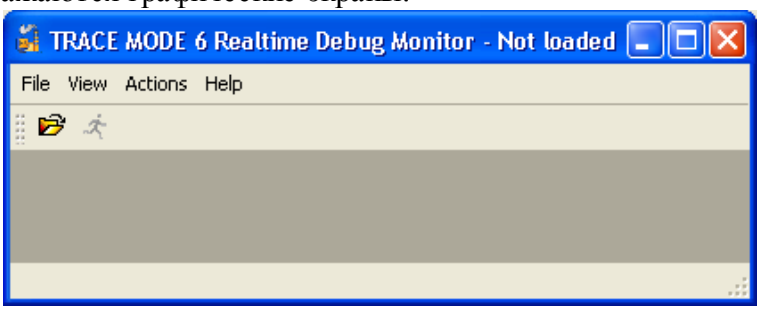
При конфигурировании ИС можно указать профайлер, который должен запускаться из ИС.

Для профайлера без поддержки графики можно задать режим отображения каналов распределенными по внутренним стандартным группам (объектам) TRACE MODE.

Профайлеры записывают протокол своей работы в файл **<имя файла prj>_<порядковый номер узла>.txt**, который сохраняется в папке узла. Степень детализации отладочной информации, выводимой в файл, может быть задана.

Профайлер с поддержкой графических экранов

Графическая оболочка этого профайлера содержит меню, панель инструментов и рабочее поле, в котором отображаются графические экраны:

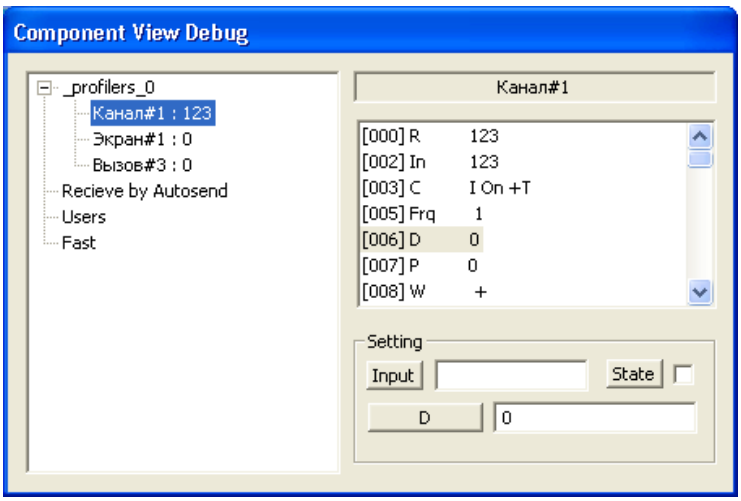


Меню **Файл** и панель инструментов содержат команды открытия ( , CTRL+O), перезагрузки и запуска ( , CTRL+R) узла, а также команду выхода из программы.

Меню **Операции** содержит команды авторизации/окончания сеанса, а также команду посылки сообщения в отчет тревог.

Меню **Вид** содержит следующие команды:

- **Полный экран** (CTRL+F) – переключение вида отображения графический экранов (в окне/полноэкранный);
- **Компоненты** (CTRL+0) – по этой команде на экране появляется диалог, в левой части которого отображаются каналы узла, а правая содержит инструменты задания атрибутов канала, выделенного в левой части:



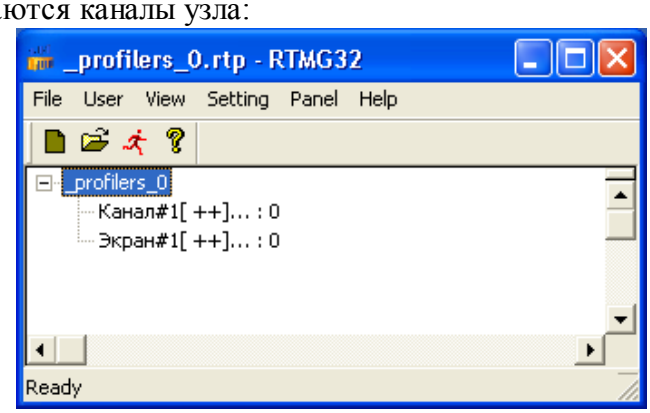
Для задания входного значения канала (атрибута 0, **In**) нужно ввести требуемое значение в поле справа от кнопки **Input** и нажать ЛК на этой кнопке.

Для задания значения произвольного атрибута нужно выделить атрибут в списке, ввести требуемое значение в поле справа от кнопки, на которую выводится короткое имя выбранного атрибута, и нажать ЛК на этой кнопке.

Для выключения канала нужно установить флаг в поле справа от кнопки **State** и нажать ЛК на этой кнопке.

Профайлер без поддержки графических экранов

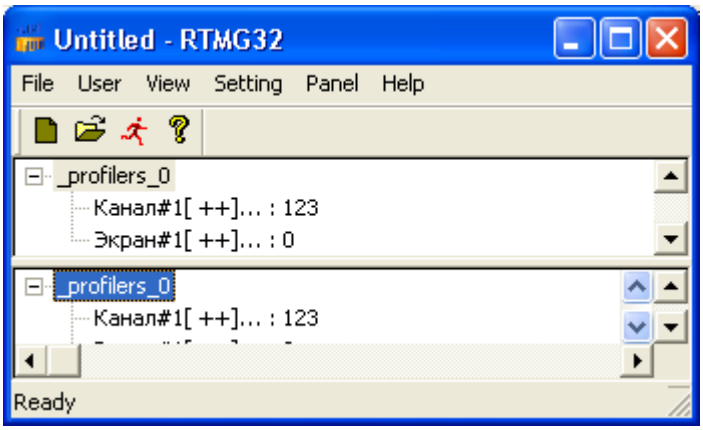
Графическая оболочка этого профайлера содержит меню, панель инструментов и рабочее поле, в котором отображаются каналы узла:



Меню **Файл** и панель инструментов содержат команды открытия (, CTRL+O), перезагрузки (, CTRL+R) и запуска ( ) узла, а также команду выхода из программы.

Меню **Пользователь** содержит команды авторизации/окончания сеанса, а также команду **Пользователи**.

Меню **Вид** содержит флаги управления видимостью строки статуса, панели инструментов и разделения окна:



5. Программирование алгоритмов в TRACE MODE 6

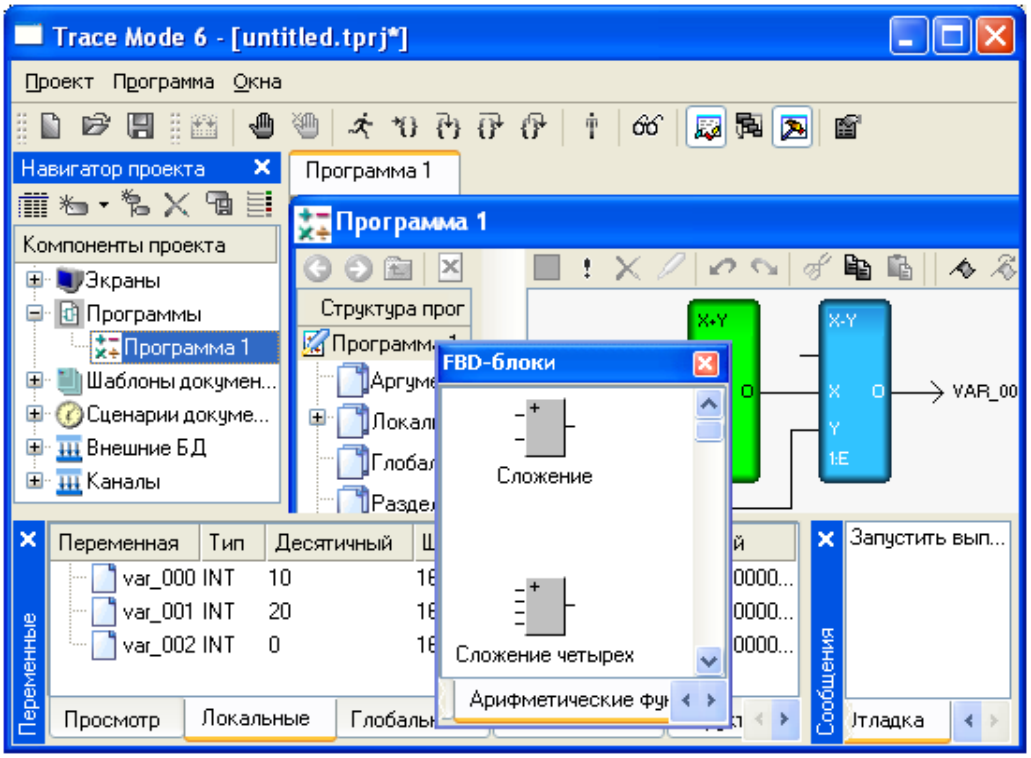
Для программирования алгоритмов функционирования разрабатываемого проекта АСУ в TRACE MODE 6 включены языки **Техно ST, Техно SFC, Техно FBD, Техно LD** и **Техно IL**. Данные языки являются модификациями языков **ST** (Structured Text), **SFC** (Sequential Function Chart), **FBD** (Function Block Diagram), **LD** (Ladder Diagram) и **IL** (Instruction List) стандарта IEC61131-3.

Программы и некоторые их компоненты (функции, шаги и переходы SFC и т.п.) могут быть разработаны на любом из встроенных языков в соответствующем редакторе, при этом языки для программы и ее компонентов выбираются независимо.

Для создания и редактирования свойств аргументов, переменных, функций и структурных типов программы, а также для использования в программе функций из внешних библиотек в интегрированную среду разработки проекта встроены специальные табличные редакторы.

TRACE MODE 6 имеет также средства для отладки программ.

Примерный вид интегрированной среды при редактировании программ показан на следующем рисунке:



Основным языком программирования TRACE MODE 6 является **Техно ST.** Программы, разработанные на языках **Техно LD**, **Техно SFC** и **Техно FBD**, перед компиляцией транслируются в **Техно ST**. **IL**-программы перед компиляцией частично транслируются в **ST**, частично – в ассемблер. Отсюда следует, например, что ключевые слова **Техно ST** являются таковыми и для всех других языков.

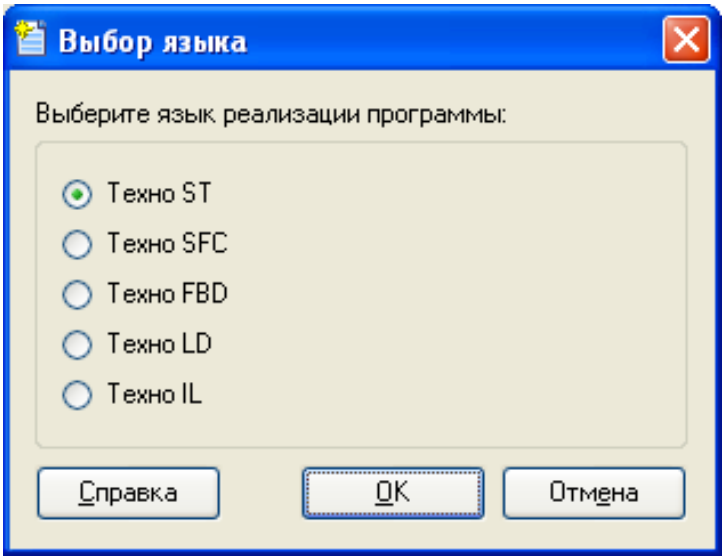
Подключение программы к проекту

Для подключения программы к проекту ее нужно вначале скомпилировать, а затем сохранить проект. Чтобы скомпилировать программу, нужно выполнить команду **Компилировать** из меню **Программа**, или нажать клавишу **F7** или нажать ЛК на иконке  панели инструментов отладчика. Перед сохранением проекта нужно убедиться, что компиляция прошла успешно (в окне сообщений компилятора в этом случае выводится соответствующее сообщение).

Отладка программы также возможна только после ее успешной компиляции.

Выбор языка программирования

Язык программирования может быть независимо задан для основной программы, функции-блока, функции и шага SFC. Язык выбирается в следующем диалоге:



Этот диалог автоматически появляется на экране при нажатии ЛК на имени вновь созданной программы или ее компонента (для которого язык может быть задан независимо) в окне структуры программы. После выбора языка программа (компонент) открывается в соответствующем редакторе.

Изменить язык можно только после удаления тела программы (компонента). Для этого нужно нажать ЛК на иконке  панели инструментов в окне структуры программы, после чего диалог выбора языка автоматически появляется на экране.

Создание элементов программ с помощью табличных редакторов

Табличные редакторы используются для создания следующих компонентов и элементов программ:

- аргументы;
- локальные переменные;
- глобальные переменные;
- функции-блоки (подпрограммы) и функции;
- структурные типы.

Кроме того, с помощью табличных редакторов конфигурируются обращения к функциям из внешних библиотек.

Перечисленные компоненты и элементы, наряду с листингами **ST** и **IL** и диаграммами **LD**, **SFC** и **FBD**, образуют ветви дерева в окне структуры программы.

Для входа в соответствующий табличный редактор нужно в окне структуры программы нажать ЛК на любом из перечисленных выше элементов.

Особенности редактирования

Для создания/удаления строк и поиска в табличных редакторах используется типовая панель инструментов.

Для перехода к редактированию отдельной ячейки таблицы нужно дважды нажать ЛК на этой ячейке. Редактирование ячейки производится либо путем непосредственного ввода с клавиатуры, либо путем выбора нужного значения из списка.

При задании числа в качестве разделителя целой и дробной части используется точка.

Если в ячейку столбца **[] Массив** ввести число, равное количеству элементов массива, то в этой ячейке отобразится диапазон индексов элементов (начиная с 0). Например, для двумерного массива при вводе **9, 8** отобразится **0 .. 8, 0 .. 7**.

Некоторые элементы (например, переменные), заданные в табличных редакторах, автоматически добавляются в листинги текстовых программ в виде соответствующих конструкций языка. Эти конструкции выделяются серым цветом; они недоступны для непосредственного редактирования с помощью клавиатуры:

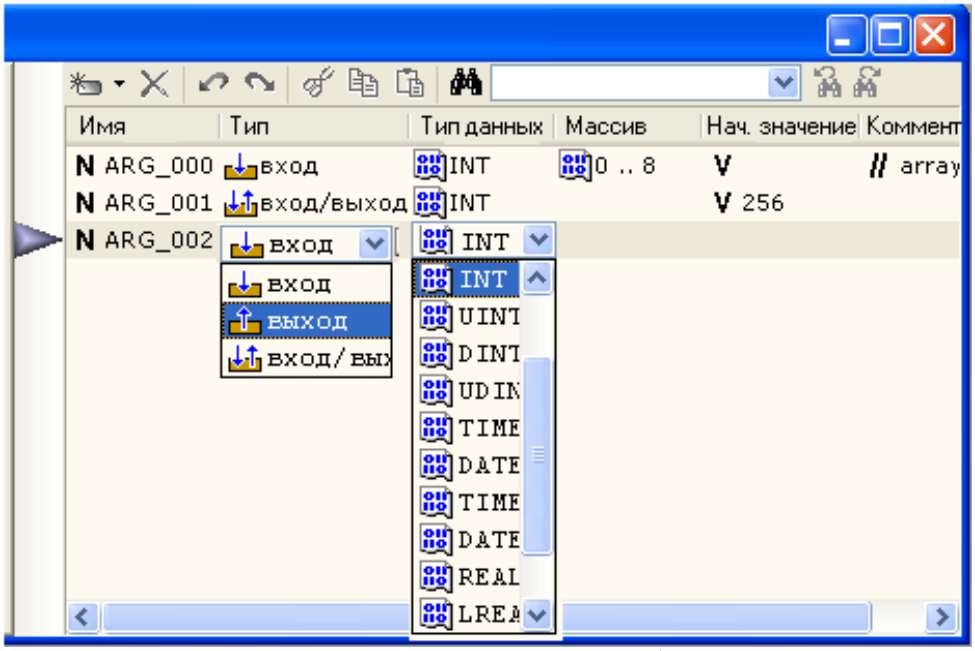
```
PROGRAM
    VAR VAR_000 : INT := 2; END_VAR
    VAR VAR_001 : INT := 3; END_VAR
    VAR VAR_002 : INT; END_VAR
    VAR s : STRUCT_000; END_VAR

    VAR_002 = fff(VAR_000, VAR_001);

    IF VAR_002 > 10 THEN
        VAR_002 = 10;
    ELSE
        VAR_002 = 1;
    END_IF;

    VAR_002 = s.f_struct(VAR_000, VAR_001);

END PROGRAM
```

Доступные типы данных (столбец **Тип данных**) для программ на всех языках одинаковы.

Начальное значение (столбец **Начальное значение**) может быть задано в любой из форм, определенных для **Техно ST**.

Табличный редактор аргументов программного компонента

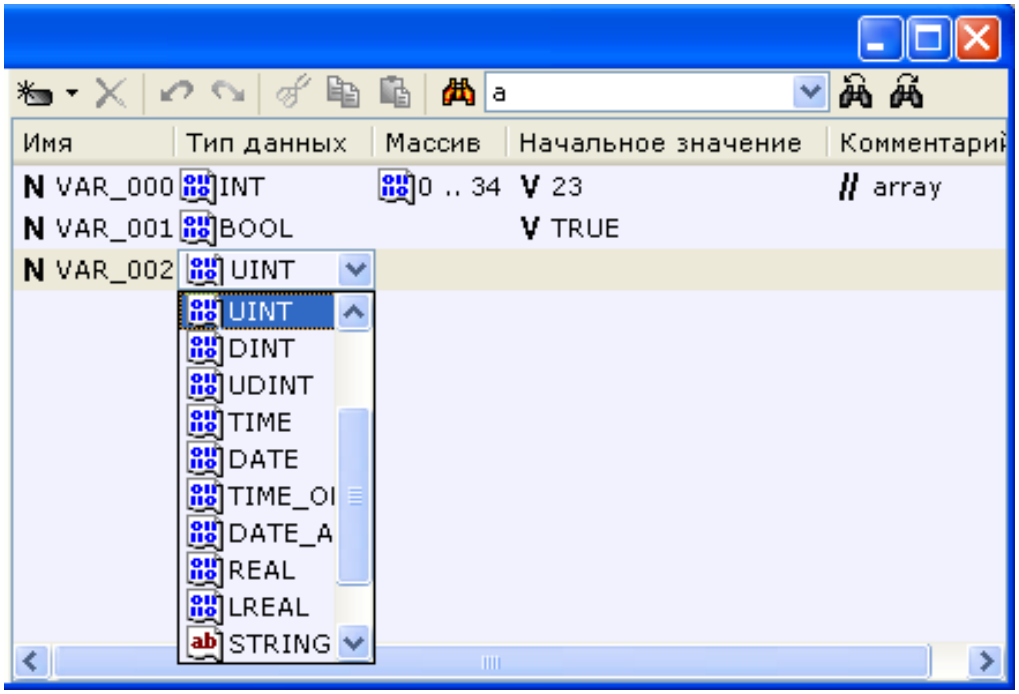
Вид табличного редактора аргументов программного компонента (функции или функции-блока) показан на следующем рисунке.



В этом редакторе задается имя аргумента, его тип (**вход**, **выход** или **вход/выход**), тип данных, начальное значение и комментарий. Если в поле **Массив** строки аргумента задать число, аргумент интерпретируется как массив.

Табличный редактор переменных

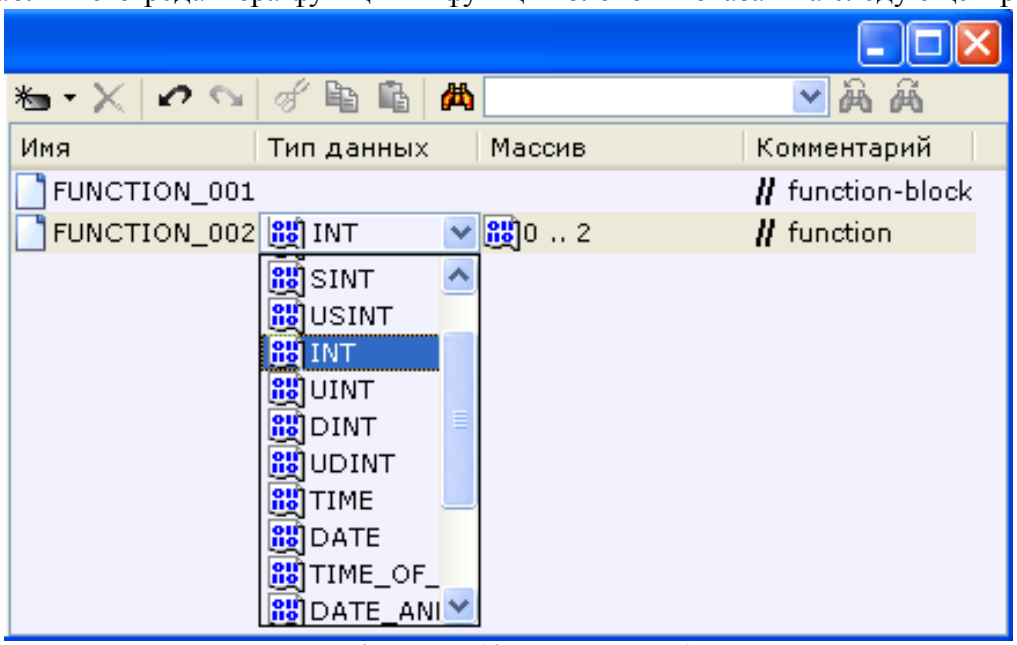
Вид табличного редактора переменных показан на следующем рисунке.



В этом редакторе задается имя переменной, ее тип данных, начальное значение и комментарий. Если в поле **Массив** строки переменной задать число, переменная интерпретируется как массив.

Табличный редактор функций и функций-блоков

Вид табличного редактора функций и функций-блоков показан на следующем рисунке.



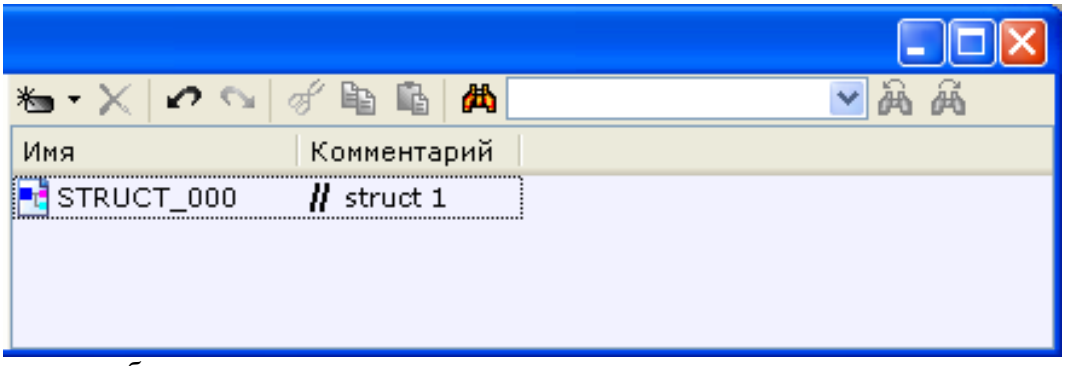
В этом редакторе задается имя функции (функции-блока) и комментарий.

Если указан тип возвращаемого значения, определяется функция, если тип возвращаемого значения не указан, определяется функция-блок.

Если в поле **Массив** строки функции задать число, функция возвращает массив. Для функции-блока поле **Массив** недоступно.

Табличный редактор структурных типов

В этом редакторе задается имя создаваемого структурного типа и комментарий.

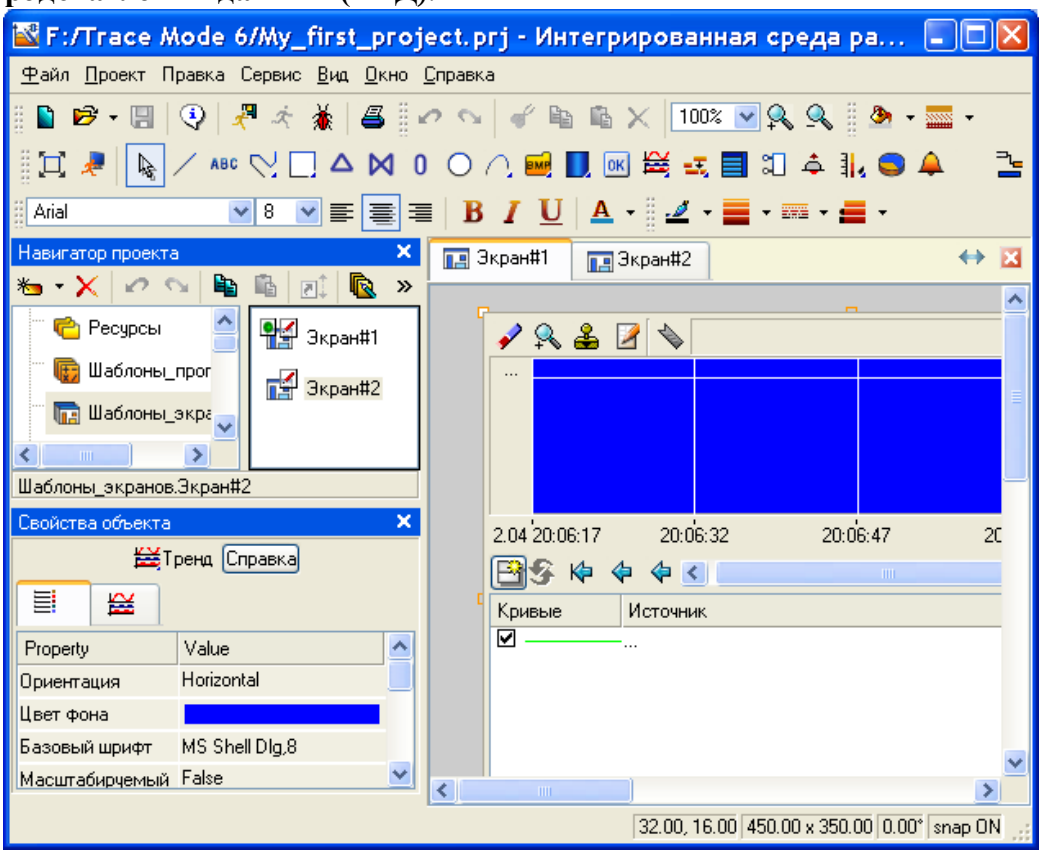


Более подробное описание алгоритмов программирования, а также описание языков программирования можно найти в справочной системе TRACE MODE в разделе **Программирование алгоритмов**.

6. Разработка графического интерфейса

Редактор представления данных

Графическое представление хода выполнения техпроцесса, а также управление техпроцессом с помощью графических средств являются одними из главных задач, решаемых ТРЕЙС МОУД 6. Для разработки интерфейса оператора в интегрированную среду встроен **редактор представления данных (РПД)**:



Интерфейс оператора разрабатывается в виде набора **графических экранов**, являющихся компонентами проекта.

С целью взаимодействия с другими компонентами проекта для графического экрана могут быть заданы аргументы.

Совокупность графических экранов узла образует его **графическую базу**. Совокупность графических баз всех узлов разрабатываемого проекта АСУТП образует **графическую часть** проекта.

Графический экран может содержать один или несколько графических **слоев**, каждый из которых, в свою очередь, может содержать один или несколько **подслоев**.

В графических слоях размещаются **графические элементы (ГЭ)**. Графические элементы имеют наборы настраиваемых **атрибутов, динамических свойств** и **функций управления**. Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени. Редактор представления данных содержит большое количество встроенных графических элементов, позволяющих изобразить практически любой техпроцесс, вывести на дисплей всю необходимую информацию о ходе его выполнения, а также управлять техпроцессом.

Режимы работы РПД

Редактор представления данных может находиться в одном из следующих режимов:

- режим **размещения** предназначен для заполнения графических слоев экранов графическими элементами. Для перехода в этот режим нужно нажать одну из кнопок выбора ГЭ на панели инструментов **Графические элементы**;
- режим **редактирования** предназначен для внесения изменений в созданные ранее графические экраны (например, для удаления/добавления графических элементов или изменения их свойств). Для перехода в этот режим надо нажать кнопку  главной панели инструментов;
- режим **эмуляции** используется для проверки работы графических элементов в реальном времени. Для перехода в режим эмуляции надо нажать кнопку  панели инструментов **Графические элементы**; для выхода из режима надо нажать кнопку  повторно.

Кроме того, в РПД предусмотрено два режима отображения графических экранов – обычный (в окне) и полноэкранный. Для переключения режима отображения используется кнопка  панели инструментов **Графические элементы**.

В режимах размещения и редактирования текущие координаты курсора отображаются в строке статуса (внизу справа). Там же отображается состояние флага **Располагать по сетке**.

Главное меню и панели инструментов РПД

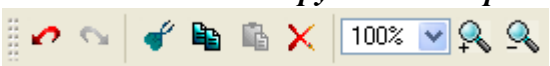
Панель инструментов 'Графические элементы'

С помощью инструментов этой панели выбираются графические элементы для размещения их в графических слоях экранов. При выборе ГЭ редактор переходит в режим размещения.

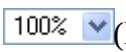
С помощью кнопки  данной панели можно перейти в режим редактирования, с помощью кнопки  – в режим эмуляции.

Кнопка  предназначена для переключения режима отображения графических экранов (обычный/полноэкранный).

Меню и панель инструментов 'Правка'



Меню и панель инструментов **Правка** содержат ряд типовых инструментов для редактирования графических экранов. Данные инструменты доступны также из контекстного меню ГЭ.

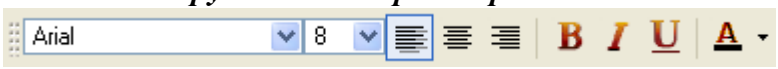
В списке  (**Масштаб**), а так же при помощи кнопок  и  панели инструментов **Правка** выбирается масштаб отображения.

Меню 'Сервис' и панель инструментов 'Топология экрана'

Данные панель инструментов и меню содержат команды для позиционирования и тиражирования выделенного графического элемента.

Меню **Сервис** содержит дополнительно команду **Параметры экрана.**

Панель инструментов 'Параметры текста'



В режиме редактирования с помощью типовых инструментов данной панели задаются параметры текста в выделенном графическом элементе (выделенной группе ГЭ). Данные команды применимы только к такому тексту, который может быть введен/отредактирован с помощью клавиатуры.

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры линии'

В режиме редактирования с помощью инструментов этой панели задаются параметры линии (линии контура) выделенного графического элемента (выделенной группы ГЭ):

 – выбор **цвета линии**. По этой команде на экран выводится стандартный диалог выбора цвета;

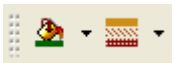
 – выбор **толщины линии.**

 – выбор **стиля линии**. По этой команде открывается список стилей, содержащий в том числе опцию **Без линии**

 – выбор **края линии (плоский, квадратный, круглый).**

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры заливки'



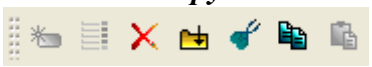
В режиме редактирования с помощью инструментов этой панели задаются параметры заливки выделенного графического элемента (выделенной группы ГЭ):

 – выбор **цвета заливки**. По этой команде на экран выводится стандартный диалог выбора цвета;

 – выбор **стиля заливки.**

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Ресурсные библиотеки'



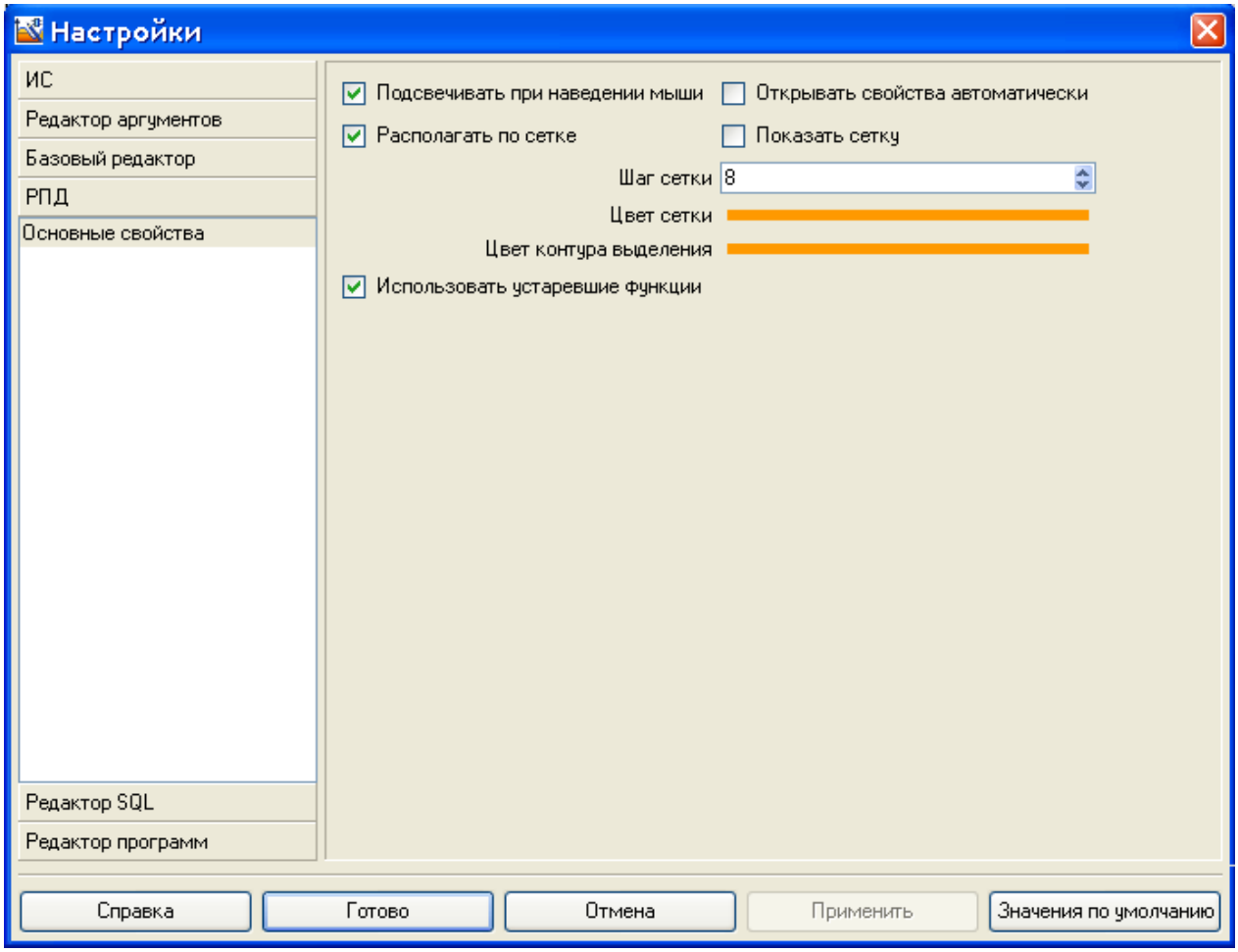
Инструменты данной панели предназначены для операций с библиотеками строк, рисунков и других ресурсов, которые могут быть использованы при разработке графических экранов.

Меню 'Вид'

Команды этого меню управляют видимостью табличного редактора аргументов экрана, окна **Слои** и таблицы графических элементов, а также панелей инструментов **Топология экрана** и **Параметры текста.**

Задание параметров РПД

Окно **Настройки**, вызываемое из меню **Файл**, в разделе **РПД** содержит диалог задания параметров РПД.

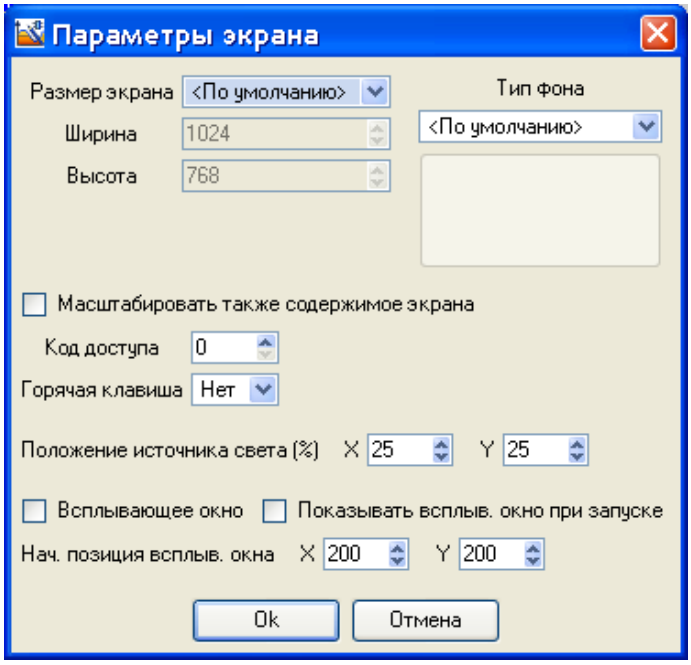


Этот диалог содержит следующие инструменты:

- **Подсвечивать при наведении мыши** – если этот флаг установлен, при наведении курсора мыши на ГЭ его вершины (узловые точки) выделяются красным цветом. Не следует путать эту функцию с функцией выделения ГЭ (.
- **Открывать свойства автоматически** – от этого флага зависит режим РПД после размещения графического элемента на экране;
- **Располагать по сетке** – если этот флаг установлен, при размещении, перемещении и масштабировании вершины прямоугольника, ограничивающего ГЭ, располагаются в узлах сетки. При размещении в узлах сетки располагаются также узловые точки ГЭ.
- **Показать сетку** – если этот флаг установлен, сетка отображается на графических экранах;
- **Шаг сетки** – задание шага сетки в пикселях (1-100);
- **Цвет сетки** – выбор цвета сетки;
- **Цвет контура выделения** – выбор цвета прямоугольника, ограничивающего ГЭ при выделении.
- **Использовать устаревшие функции** – при установке этого флага доступны некоторые опции предыдущих версий ТРЕЙС МОУД.

Задание параметров графического экрана

Параметры редактируемого графического экрана задаются в диалоге, который открывается при выполнении команды **Параметры экрана** меню **Сервис**:



Этот диалог содержит следующие инструменты:

- **Размер экрана** – задание размера экрана в пикселях. Размер можно выбрать из нескольких стандартных или задать свой с помощью опции **Произвольный** и полей **Ширина** и **Высота.**
- **Тип фона** – выбор типа фона. Содержит следующие варианты:
- **<По умолчанию>** – фон экрана по умолчанию;
- **Цвет** – при выборе этой опции для экрана можно задать цвет фона – для этого нужно нажать кнопку под списком и выбрать цвет в стандартном диалоге;
- **Изображение** – при выборе этой опции в качестве фона экрана можно использовать рисунок;
- **Масштабировать также содержимое экрана** – флаг, при установке которого размещенные на экране ГЭ масштабируются пропорционально изменению размеров экрана;
- **Код доступа** – код доступа к экрану (0-255). Права на доступ к экранам задаются для пользователя в виде маски в разделе **Доступ** / **Экраны** канала **Пользователь**. При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к экрану разрешен, в противном случае – запрещен;
- **Горячая клавиша** – меню выбора функциональной горячей клавиши (**F2** – **F12**). При запуске проекта в реальном времени нажатие заданной клавиши будет сопровождаться переходом на данный экран;
- **Положение источника света (%)** – положение источника света относительно экрана (угол с осями X и Y в процентах). Значение (50, 50) соответствует расположению источника света на нормали к экрану;
- **Всплывающее окно** – показывать экран в списке всплывающих экранов MPB;
- **Показывать всплыв. окно при запуске** – показывать экран при загрузке узла в MPB;
- **Нач. позиция всплыв. окна , X** и **Y** – положение всплывающего экрана по осям X и Y при загрузке узла в MPB. Данная опция не влияет на отображение обычного экрана.

Часть параметров для создаваемых графических экранов можно задать в редакторе группы шаблонов экранов.

Задание аргументов графического экрана

Для графического экрана могут быть заданы аргументы с целью взаимодействия с другими компонентами проекта. Чтобы открыть табличный редактор аргументов экрана, нужно выбрать опцию **Аргументы экрана** в меню **Вид.**

Операции с графическими элементами

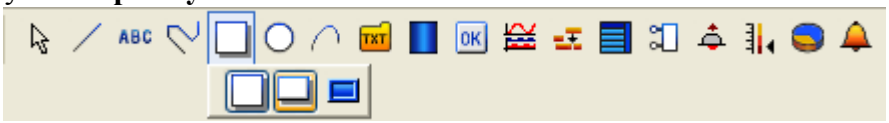
Размещение ГЭ

Встроенные графические элементы разбиты на группы. Каждой группе соответствует кнопка на панели инструментов **Графические элементы**. На кнопку выводится иконка одного из элементов данной группы.

Чтобы выбрать ГЭ для размещения, нужно выполнить следующие действия:

- нажать ЛК на кнопке панели инструментов **Графические элементы**. При этом выбирается тот элемент, чья иконка выведена на кнопку (элемент, заданный по умолчанию для соответствующей группы, или элемент, выбранный ранее);
- дважды нажать ЛК на кнопке и затем нажать ЛК на иконке требуемого ГЭ в появившемся меню (меню не открывается, если группа содержит только один графический элемент).

После выбора элемента его иконка выводится на кнопку группы. Например, на рисунке показано меню группы **Прямоугольники**:



После выбора элемента его иконка выводится на кнопку группы:

При выборе графического элемента редактор представления данных переходит в режим размещения, при этом курсор на графическом экране приобретает вид +.

Далее в окне **Слои** необходимо нажатием ЛК указать слой, в котором должен быть размещен выбранный графический элемент.

Далее продолжить процедуру размещения ГЭ можно двумя способами:

- перетащить ГЭ с панели инструментов на экран (метод **drag-and-drop**); после размещения ГЭ имеет размеры, заданные по умолчанию, РПД переходит в режим редактирования, окно свойств ГЭ открывается автоматически;
- переместить курсор в нужную точку экрана и нажатием ЛК установить **точку привязки** ГЭ. Далее действия по размещению ГЭ могут отличаться, однако для большинства графических элементов они стандартны – перемещение мыши после установки точки привязки выводит на экран образ ГЭ, при этом отрезок от точки привязки до текущего положения курсора является диагональю прямоугольника, ограничивающего ГЭ. (Если при перемещении мыши удерживать нажатой клавишу **CTRL**, ряд ГЭ окажется вписанным в квадрат). Повторное нажатие ЛК приводит к размещению графического элемента в выбранном графическом слое.

Для графических элементов групп **Ломаные** и **Кривые** каждое нажатие ЛК после установки точки привязки задает **узловую точку** (промежуточную вершину). Для установки последней вершины и выхода из режима размещения этих ГЭ нужно нажать ПК. Положение узловых точек, заданное при размещении, можно в дальнейшем изменить.

Режим РПД после размещения ГЭ данным способом зависит от флага **Открывать свойства автоматически**:

- если флаг установлен, то после размещения ГЭ автоматически открывается окно его свойств, а РПД переходит в режим редактирования;
- если флаг не установлен, то после размещения ГЭ РПД остается в режиме размещения. Этот способ удобен для многократного размещения на экране одного и того же графического элемента.

Перемещение и масштабирование ГЭ

Для перемещения или изменения размеров выделенного ГЭ (группы ГЭ) нужно выбрать в контекстном меню графического элемента режим **Перемещать/масштабировать** и далее использовать стандартные операции редактирования.

Удаление ГЭ

Для удаления выделенного ГЭ (группы ГЭ) нужно нажать клавишу **Del** на клавиатуре или выполнить команду **Удалить** с помощью меню или панели инструментов **Правка** или с помощью контекстного меню ГЭ.

В полноэкранном режиме редактирования для удаления выделенного ГЭ (группы ГЭ) с помощью клавиатуры используется комбинация клавиш **Ctrl+Del**.

Копирование и вставка ГЭ

Для копирования выделенного ГЭ (группы ГЭ) в буфер обмена нужно выполнить команду **Копировать.**

Для вставки содержимого буфера обмена в слой нужно выполнить команду **Вставить.**

Скопировать в буфер обмена можно в том числе группу графических элементов, лежащих в разных слоях экрана, однако при вставке такой группы все ее ГЭ будут размещены в одном и том же слое.

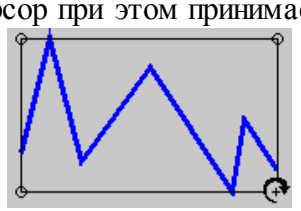
Выполнить команды **Копировать** и **Вставить** можно с помощью меню или панели инструментов **Правка** или с помощью контекстного меню ГЭ.

Поворот ГЭ

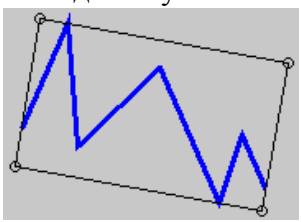
Существует 2 способа поворота ГЭ на экране в режиме редактирования.

Режим 'Повернуть'

Для перехода в этот режим нужно выбрать в контекстном меню выделенного ГЭ (группы ГЭ) опцию **Повернуть**. Далее надо установить курсор в одну из вершин прямоугольника, ограничивающего ГЭ (группу ГЭ) (курсор при этом принимает вид):



Затем нужно нажать ЛК и, удерживая кнопку нажатой, перемещением мыши задать нужный угол поворота ГЭ. Для выхода из режима надо отпустить ЛК:



При использовании данного метода ГЭ (группа ГЭ) вращается относительно центра ограничивающего прямоугольника (центром прямоугольника является точка пересечения его диагоналей).

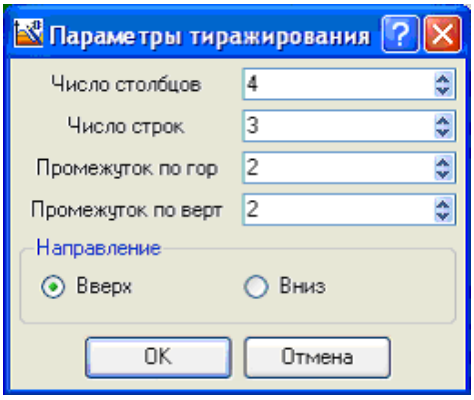
Поворот из таблицы графических элементов

Угол поворота ГЭ можно задать в поле **Угол (градусы)** диалога **Геометрия**. При использовании этого метода ГЭ поворачивается относительно точки привязки (при задании положительного значения – по часовой стрелке).

Тиражирование ГЭ

Тиражирование – это копирование выделенного графического элемента и его множественная вставка с табличным упорядочением. Копии вставляются вправо и вверх/вниз относительно выделенного ГЭ. Тиражирование выделенной группы ГЭ не поддерживается.

Операция тиражирования конфигурируется в диалоге, который открывается при выполнении команды **Тиражировать** из меню **Сервис** или нажатии кнопки на панели инструментов **Топология экрана**:



В этом диалоге задаются следующие параметры:

- **Число столбцов** – число элементов в строке (включая первоначально выделенный ГЭ);
- **Число строк** – число строк;
- **Промежуток по горизонтали** – промежуток между копиями по горизонтали в пикселях;
- **Промежуток по вертикали** – промежуток между копиями по вертикали в пикселях;
- **Вверх** – размножение по строкам вверх относительно выделенного ГЭ;
- **Вниз** – размножение по строкам вниз относительно выделенного ГЭ.

Ниже показан результат тиражирования кнопки (выделена на рисунке) при следующих параметрах: число столбцов – 4, число строк – 3, промежуток по горизонтали и вертикали – 5, направление тиражирования – вниз:

Задание типовых свойств ГЭ

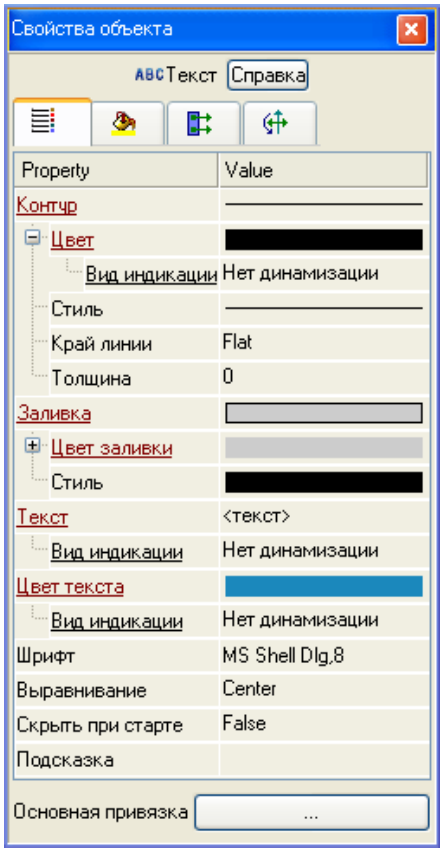
В этом разделе описаны типовые свойства графических элементов и инструменты их задания.

Графические элементы имеют следующие настраиваемые свойства:

- **Атрибуты**
- **Динамические свойства**
- **Функции управления**

Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени:

Для задания свойств ГЭ (группы ГЭ) используется окно **Свойства объекта**, содержащее различное число вкладок для разных элементов:



Чтобы открыть это окно, нужно выполнить команду **Свойства** из контекстного меню выделенного ГЭ (выделенной группы ГЭ) или дважды нажать ЛК мыши на ГЭ. При снятии выделения ГЭ окно его свойств автоматически закрывается. Окно **Свойства объекта** недоступно, если графический элемент расположен в слое, редактирование которого запрещено.

Атрибуты – это простейшие свойства графического элемента. Они задаются на вкладке (**Основные свойства**) окна **Свойства объекта**.

В окне свойств атрибуты могут быть сгруппированы – наименования таких групп выделены подчеркиванием, при двойном нажатии на них ЛК раскрывается список свойств.

Существуют 2 вида атрибутов ГЭ:

- **статические** – атрибуты, которые не изменяются при работе в реальном времени;
- **динамизируемые** – атрибуты, которые могут быть как статическими, так и динамическими (изменяющимися при работе в реальном времени в зависимости от значения привязанного аргумента). Разделы конфигурирования таких атрибутов выделены красным цветом и содержат пункт **Вид индикации**.

Статические атрибуты ГЭ

В данном разделе описано задание типовых статических атрибутов ГЭ с помощью вкладок окна **Свойства объекта**.

Видимость при старте

При переходе в режим реального времени (т.е. при запуске проекта в МРВ или при переходе в режим эмуляции) графические элементы по умолчанию видимы. Чтобы скрыть ГЭ при запуске режима реального времени, нужно установить для него атрибут **Скрыть при старте**:

В режиме редактирования видимы все ГЭ.

В реальном времени управлять видимостью ГЭ можно с помощью функции управления ГЭ и с помощью ГЭ **Свободные формы**

Всплывающая подсказка

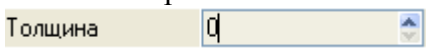
В разделе **Подсказка** для ГЭ задается всплывающая подсказка, отображаемая на экране при наведении курсора на ГЭ в режиме реального времени:

Цветовые атрибуты

Разделы конфигурирования цветовых атрибутов ГЭ (**Цвет**, **Цвет линии**, **Цвет заливки**, **Цвет текста** и т.п.) в окне **Свойства объекта** содержат типовой инструмент задания цвета – кнопку 

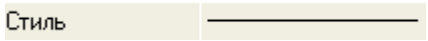
Толщина линии

Раздел конфигурирования атрибута **Толщина** содержит инструмент выбора толщины линии и окно отображения численного значения толщины (в пикселях):



Стиль линии

Раздел конфигурирования атрибута **Стиль линии** содержит кнопку  :

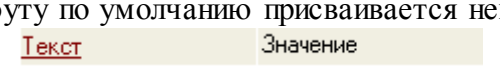


Стиль заливки

Раздел конфигурирования атрибута **Стиль заливки** содержит кнопку  :

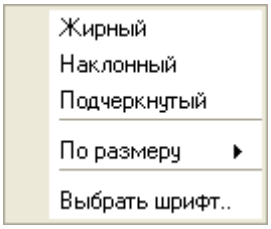
Текстовые атрибуты

Раздел конфигурирования текстовых атрибутов (**Текст**, **Надпись** и т.п.) содержит окно обычного текстового редактора для задания значения атрибута (для некоторых ГЭ подобному атрибуту по умолчанию присваивается некоторое значение):

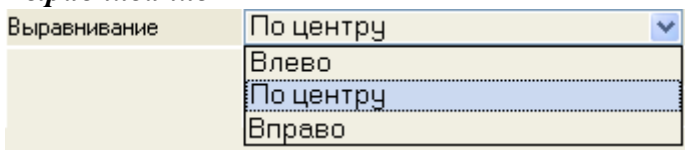


Шрифт

Раздел конфигурирования атрибута **Шрифт** содержит кнопку, при нажатии которой на экран выводится меню выбора параметров шрифта:



Выравнивание



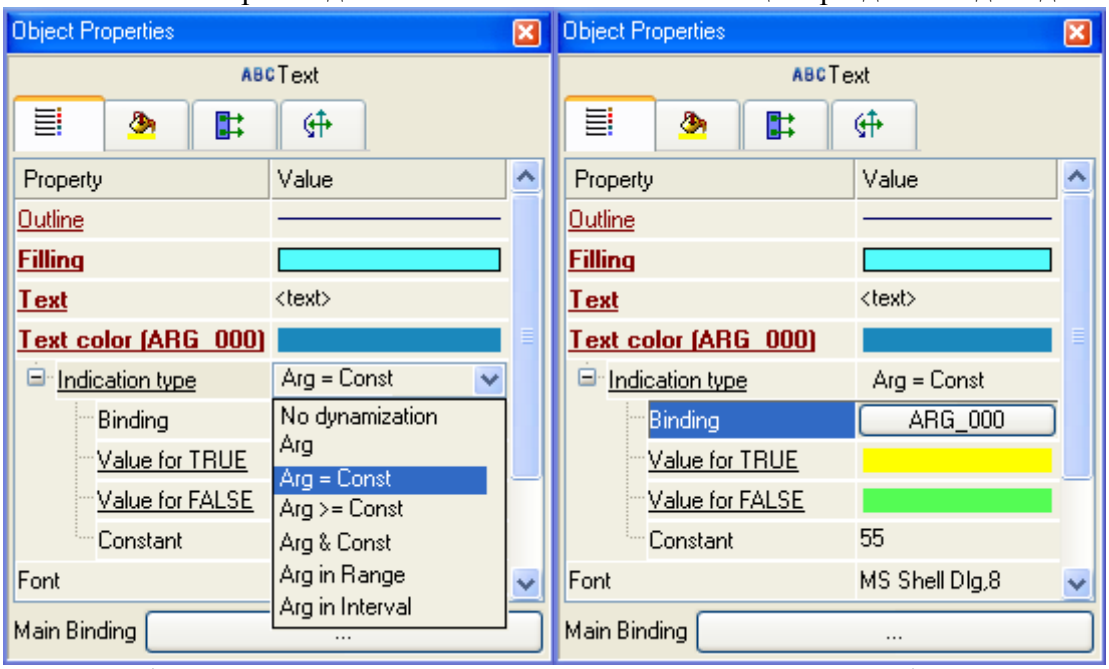
Раздел конфигурирования атрибута **Выравнивание** содержит стандартные инструменты выравнивания текста: **Влево**, **По центру** и **Вправо**.

Динамизация атрибута ГЭ

Динамизацией атрибута называется задание условий его изменения в зависимости от значения привязанного аргумента. При динамизации атрибута графический элемент становится индикатором выполнения заданных условий.

При размещении ГЭ на экране все его динамизируемые атрибуты по умолчанию статические, и разделы их конфигурирования на вкладке **Основные свойства** окна свойств содержат инструмент задания соответствующего статического параметра. Например, при размещении ГЭ **Текст** раздел динамизируемого атрибута **Цвет текста** содержит инструмент выбора цвета  :

Чтобы динамизировать атрибут, нужно дважды нажать на названии ЛК мыши, и в раскрывшемся списке настроить динамические свойства с помощью раздела **Вид индикации**:



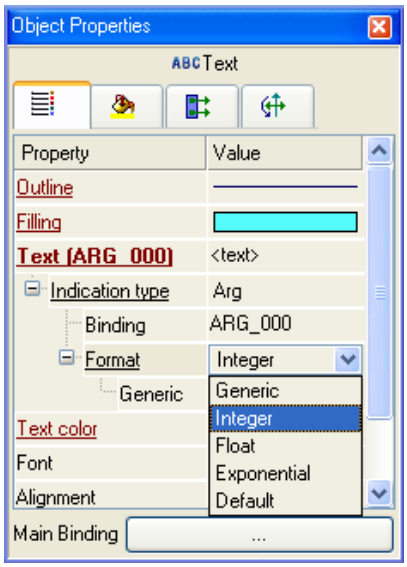
Вид условия (и, соответственно, вид индикатора, создаваемого из ГЭ), выбирается в разделе **Вид индикации**:

- **Значение** – индикация значения аргумента;
- **Arg = Конст.** – индикация равенства аргумента заданной константе;
- **Arg >= Конст.** – индикация превышения аргументом заданного порога;
- **Arg & Конст.** – индикация результата побитового умножения аргумента на значение заданной константы;
- **Arg в диапазоне** – индикация нахождения аргумента в заданных диапазонах;
- **Arg в интервале** – индикация нахождения аргумента в интервалах привязанного канала.

В зависимости от выбранного вида индикации меняются инструменты его конфигурирования.

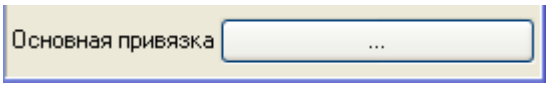
Индикация значения

Вид индикации **Значение** может быть задан при динамизации атрибута **Текст**. Формат вывода выбирается в списке **Формат**:



Основная привязка

Окно свойств некоторых ГЭ содержит раздел **Основная привязка.**



При нажатии на кнопку выводится стандартный диалог выбора аргумента.

После установки какого-либо аргумента в качестве основной привязки, он действует для всех динамизируемых атрибутов данного ГЭ.

Динамические свойства ГЭ

К динамическим свойствам графических элементов относятся **динамическая заливка,** 3 вида **динамической трансформации** (**перемещение, масштабирование** и **вращение**) и **динамический контур**.

Динамические свойства ГЭ, как и динамизированные атрибуты, используются для графического отображения значений аргументов экрана при работе в реальном времени.

Динамические свойства настраиваются соответственно на вкладках **Динамическая заливка** ( ), **Динамическая трансформация** ( ) и **Динамический контур** ( ) окна **Свойства объекта.**

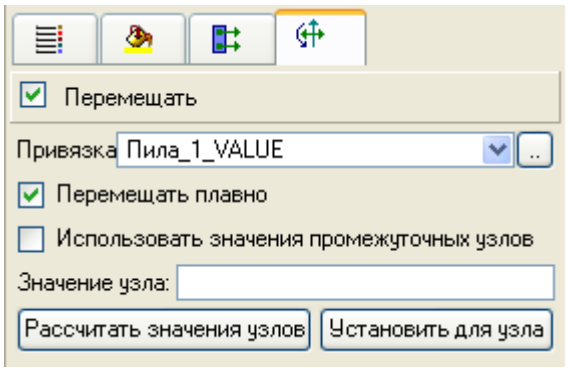
Динамическая заливка ГЭ

При использовании данного свойства ГЭ отображает значение привязанного аргумента числового формата в виде закрашенной области (слоя). Поддерживаются два вида динамической заливки – **одноуровневая** (отображает значение одного аргумента) и **многоуровневая** (отображает значения нескольких аргументов). Оба вида настраиваются на вкладке **Динамическая заливка** ( ) окна **Свойства объекта**. Для использования динамической заливки нужно на этой вкладке установить флаг **Использовать.**

Можно задать несколько слоев заливки. Для добавления/удаления нового слоя используется контекстное меню, вызываемое нажатием ПК мыши на названиях пунктов **Слои/Слой** соответственно. Настройки для всех создаваемых слоев имеют одинаковое назначение.

Динамическое перемещение ГЭ

Это свойство настраивается в разделе **Перемещать** вкладки **Динамическая трансформация** ( ) окна **Свойства объекта**:

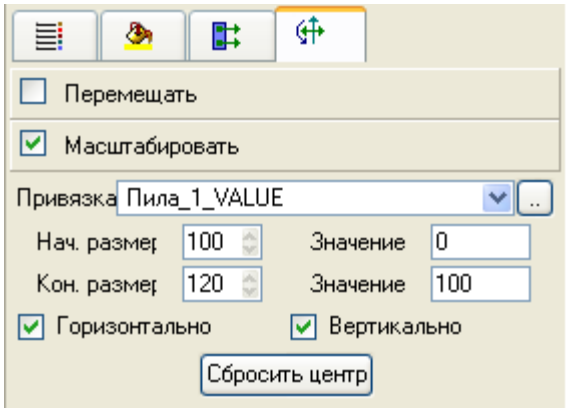


Чтобы использовать данное динамическое свойство, надо установить флаг **Перемещать.**

При работе в реальном времени графический элемент перемещается вдоль траектории, которая задается как ломаная линия (количество узлов ломаной не ограничено). Текущее положение ГЭ зависит от значения привязанного аргумента (числовой аргумент для привязки выбирается в списке **Привязка**, от значений, заданных для узлов траектории, и флага **Перемещать плавно**.

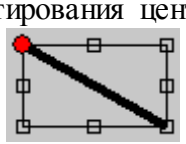
Динамическое масштабирование ГЭ

Это свойство настраивается в разделе **Масштабировать** вкладки **Динамическая трансформация** ( ) окна **Свойства объекта**:

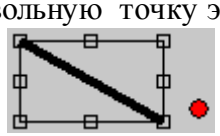


Для перехода в этот раздел надо нажать ЛК на заголовке раздела. Чтобы данное динамическое свойство было использовано при работе в реальном времени, надо установить флаг **Масштабировать**.

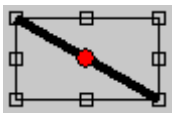
Применение свойства **Масштабировать** изменяет координаты точек ГЭ относительно **центра масштабирования**, который по умолчанию располагается в точке привязки ГЭ (в режиме редактирования центр масштабирования отображается на экране в виде красной точки):



С помощью метода **drag-and-drop** центр масштабирования может быть перемещен в произвольную точку экрана:

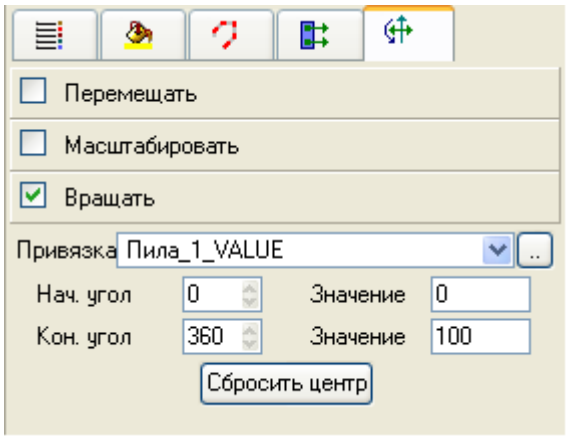


При нажатии кнопки **Сбросить центр** центр масштабирования устанавливается в точку пересечения диагоналей прямоугольника, ограничивающего ГЭ:



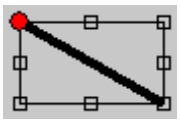
Динамическое вращение ГЭ

Это свойство настраивается в разделе **Вращать** вкладки **Динамическая трансформация** ( ) окна **Свойства объекта**:

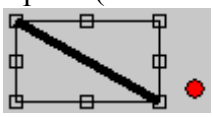


Для перехода в этот раздел надо нажать ЛК на заголовке раздела. Чтобы данное динамическое свойство было использовано при работе в реальном времени, надо установить флаг **Вращать**.

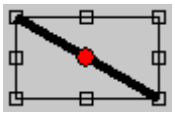
Точка, относительно которой вращается ГЭ (**центр вращения**), по умолчанию располагается в точке привязки ГЭ (в режиме редактирования данного динамического свойства центр вращения отображается на экране в виде красной точки):



С помощью метода **drag-and-drop** центр вращения может быть перемещен в произвольную точку экрана (в том числе за пределами ГЭ):

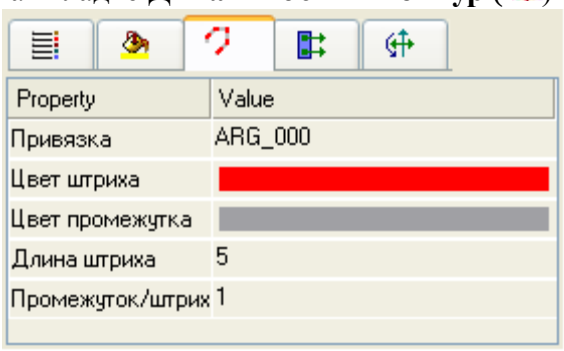


При нажатии кнопки **Сбросить центр** центр вращения устанавливается в точку пересечения диагоналей прямоугольника, ограничивающего ГЭ:



Динамический контур ГЭ

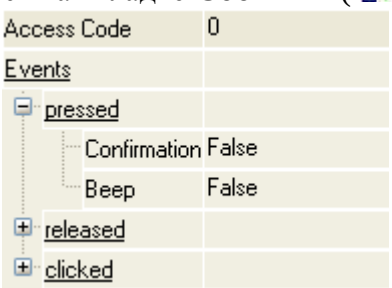
Динамический контур представляет собой прокручиваемый по часовой стрелке пунктир (под прокруткой здесь подразумевается дискретное перемещение с шагом, равным длине штриха). Это свойство настраивается на вкладке **Динамический контур** ( ) окна **Свойства объекта.**:



Функции управления ГЭ

Функции управления ГЭ – это действия, заданные для графических элементов на этапе редактирования проекта АСУ; выполнение этих действий при работе в реальном времени инициализируется оператором с помощью мыши. Задание функций управления для графических элементов придает графическим экранам свойство интерактивности и обеспечивает одно из важнейших качеств АСУ – управление техпроцессом с помощью графических средств.

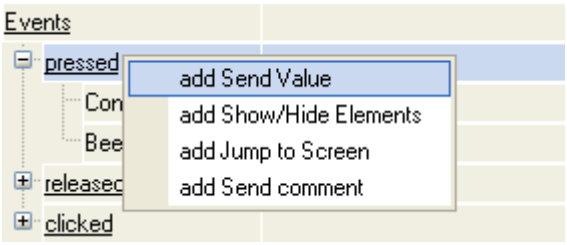
Функции управления задаются на вкладке **События** ( ) окна **Свойства объекта**:



Определены следующие **события**, по которым инициализируется выполнение действий в реальном времени:

- **нажатие** ЛК на ГЭ;
- **отжатие** ЛК на ГЭ;

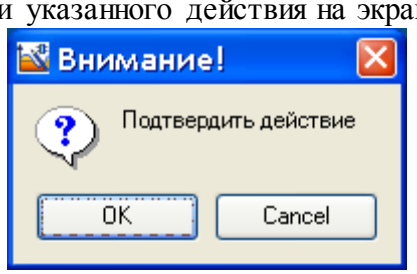
Для каждого из событий может быть независимо задано несколько функций управления, выбираемых из контекстного меню (меню открывается при нажатии ПК мыши на названии события):



- **передать значение**;
- **показать/скрыть элементы**;
- **перейти на экран**;
- **послать комментарий.**

Функции управления отображаются в виде новых разделов списка свойств объекта (для каждой функции создается отдельный раздел). Для удаления функции управления используется контекстное меню, вызываемое нажатием ПК мыши на ее названии.

Для каждого события можно задать подтверждение и звуковой сигнал. Для этого используются атрибуты **Подтверждение** и **Сигнал**. При установке подтверждения в режиме реального времени при совершении указанного действия на экран выводится следующее окно:



Чтобы подтвердить действие нужно нажать клавишу **ОК**. Для отмены – **Cancel**.

Если установлен **Сигнал**, то при совершении указанного действия система воспроизводит **Стандартный звук**, заданный в Windows.

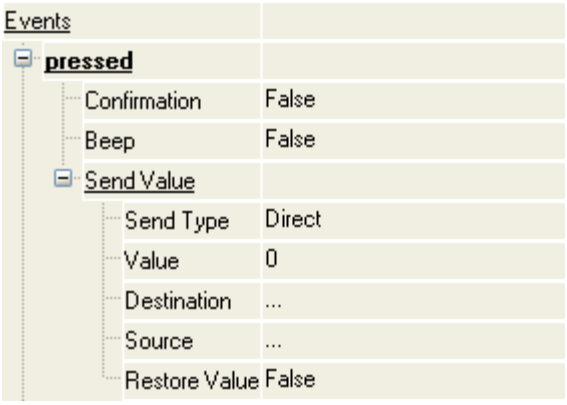
Код доступа – код доступа к функциям управления (0-255). Права на доступ к функциям управления задаются для пользователя в виде маски в разделе **Доступ** / **Формы** канала **Пользователь**. При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к функциям управления разрешен, в противном случае – запрещен.

Функция передачи значения

Функция передачи значения используется для изменения значения аргументов экрана.

Для одного ГЭ можно задать несколько функций передачи значения, применительно к различным аргументам.

При добавлении этой функции управления в списке свойств объекта появляется следующий раздел:



Поле **Тип передачи** содержит следующие варианты для передачи значений:

- **Прямая**;
- **Ввести и передать**;

- **НЕ-ИЛИ**;
- **ИЛИ**;
- **И**;
- **Прибавить**;
- **Добавить процент шкалы**;
- **Умножить**;
- **Разделить**;

Атрибут **Источник** задает исходный аргумент, с которым проводится выбранная операция. Результат операции записывается в аргумент, задаваемый атрибутом **Результат**. Атрибуты **Источник** и **Результат** могут иметь привязку к одному и тому же аргументу.

Атрибут **Восстанавливать значение** используется только для ГЭ, запускающих выполнение действий по нажатию мыши. Если этот атрибут имеет значение **True**, то по нажатию ЛК значение аргумента будет изменено, а по отпусканию – восстановлено обратно.

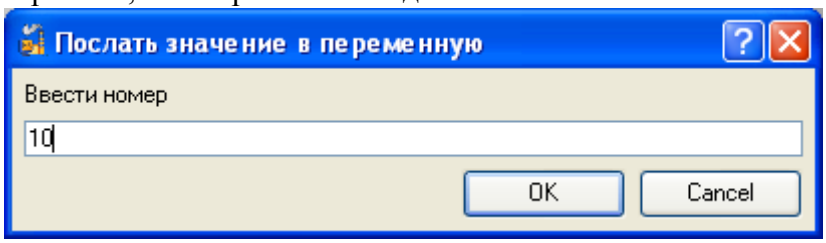
Значения, формируемые до сравнения с маской, задаются при помощи поля **Тип передачи**:

Прямая

Формируемое значение задается непосредственно в поле **Значение**.

Ввести & передать

Формируемое значение задается в диалоговом окне, появляющемся при запуске проекта в мониторе реального времени, если произошло заданное **событие**.



НЕ-ИЛИ

Формируемое значение является результатом логической операции исключающего сложения между аргументом и числом, указанным в поле **Значение**.

ИЛИ

Формируемое значение является результатом логической операции сложения между аргументом и числом, указанным в поле **Значение**.

И

Формируемое значение является результатом логической операции умножения между аргументом и числом, указанным в поле **Значение**.

Прибавить

Формируемое значение является текущим значением аргумента, увеличенным на число, заданное в поле **Значение**.

Добавить процент шкалы

Формируемое значение является текущим значением аргумента, увеличенным на процент от величины шкалы привязанного к аргументу канала. Процент шкалы задается в поле **Значение**.

Умножить

Формируемое значение является произведением текущего значения аргумента на число, заданное в поле **Значение.**

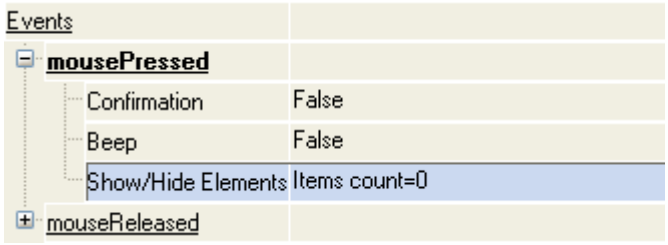
Разделить

Формируемое значение является частным от деления текущего значения аргумента на число, заданное в поле **Значение.**

Функция управления видимостью ГЭ

Функция управления видимостью ГЭ служит для скрытия и/или отображения (в зависимости от текущего состояния) одного или нескольких выбранных ГЭ на графическом экране.

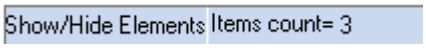
При добавлении функции **Показать/скрыть элементы** в списке настроек появляется соответствующая строчка:



Для выбора управляемых ГЭ нужно нажать ЛК мыши в поле **Значение** строки **Показать/скрыть элементы**. При этом его значение привет вид:

После этого на графическом экране нужно ЛК мыши выбрать ГЭ (или несколько ГЭ, нажимая на них ЛК мыши при нажатой клавише **Control**).

После этого появится надпись **Всего = N**, где **N** – число выделенных ГЭ.



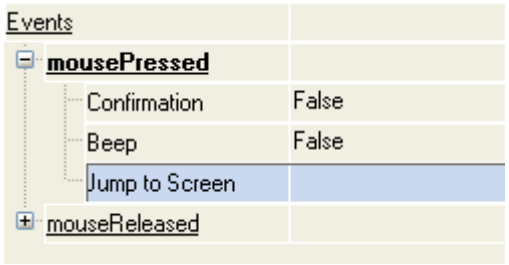
При запуске проекта в реальном времени при возникновении указанного **События** выбранные ГЭ скрываются с экрана, при повторном возникновении снова отображаются, и т.д.

Чтобы редактировать выбранные ГЭ, нужно нажать на поле **Значение** строки **Показать/скрыть элемент** и добавить/снять выделение для соответствующих элементов. При этом кнопка **Сбросить** очищает список выбранных ГЭ.

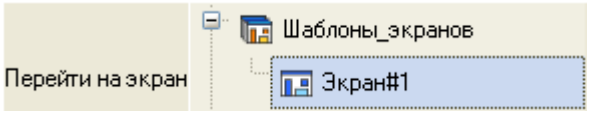
Функция перехода на экран

Функция перехода на экран осуществляет переключение с текущего экрана на другой при наступлении заданного **События.**

При создании функции управления в списке настроек появляется соответствующая строчка:



При нажатии ЛК мыши в поле **Значение** строки **Перейти на экран** появляется список, в котором можно выбрать экран, на который требуется перейти.



Переключением экранов в реальном времени можно также управлять с помощью канала класса **Вызов.**

7. Описание встроенных графических элементов

Группа ГЭ ‘Ломаные’

В эту группу входят следующие ГЭ.

- **Ломаная линия**
- **Многоугольник**
- **Труба**

Графические элементы **Ломаная линия** и **Многоугольник** не имеют специфических свойств и размещаются в графическом слое стандартным способом.

ГЭ ’Труба’

Данный ГЭ по сути является элементом объемной графики, и имеет все общие специфические свойства с ними.

Специфические атрибуты **Край 1** и **Край 2** задают разнообразную форму краев трубы:

Группа ГЭ ‘Прямоугольники’

В эту группу входят следующие ГЭ:

- **Прямоугольник**
- **Панель**
- **Рамка**

Данные элементы размещаются в графическом слое стандартным способом.

Наряду с типовыми параметрами, ГЭ **Панель** и **Рамка** имеют ряд специфических атрибутов (ниже показана вкладка **(Осн. свойства)** окна свойств для ГЭ **Панель** и **Рамка**):

Property	Value
Border	5
Border color	
Fill color	
Fill	True
Inset	False
System colors	True
Hide at Start	False
Tooltip	

- **Заливка** – если для данного атрибута установлено значение **False**, заливка пропадает и ГЭ становится прозрачным. Ниже показаны ГЭ **Панель** и **Рамка** с заливкой и без:

- **Утопленный** – этот атрибут определяет вид ГЭ – **выступающий** (**False**) / **утопленный** (**True**). Ниже показаны ГЭ **Панель** и **Рамка** в положении **выступающий** и **утопленный**:

- **Системные цвета** – при значении атрибута **True** для ГЭ игнорируются цвета, заданные настройками **Цвет контура** и **Цвет заливки**. Используются системные цвета MS Windows. Ниже показаны ГЭ **Панель** и **Рамка** при использовании системных цветов и заданных настройками ГЭ:

Группа ГЭ ‘Объемная графика’

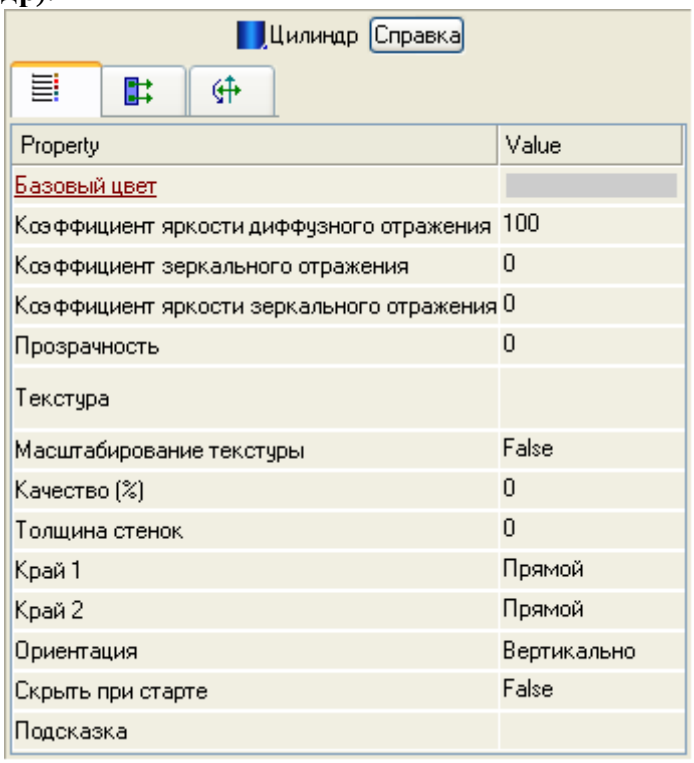
В эту группу входят следующие ГЭ:

- **Цилиндр**
- **Сфера**
- **Конус**
- **Тор**
- **Пирамида**
- **Емкость**

- **Клапан**

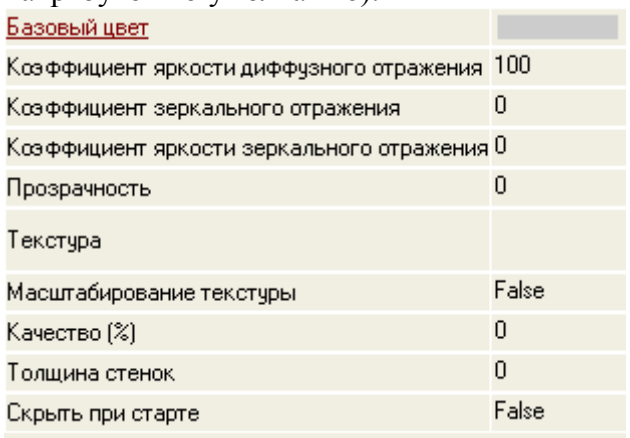
Данные элементы размещаются в графическом слое стандартным способом.

Наряду с типовыми параметрами **Скрыть на старте**, **Подсказка** и **Базовый цвет**, объемные ГЭ имеют ряд специфических атрибутов (ниже показана вкладка **(Осн. свойства)** окна свойств ГЭ **Цилиндр**):



Общие специфические атрибуты объемных ГЭ

Специфическими атрибутами, общими для всех ГЭ группы, являются следующие (на рисунке показаны значения атрибутов по умолчанию):

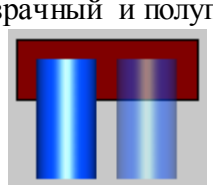


В РПД используется модель освещения объемного элемента одним источником белого света (положение источника задается в диалоге **Параметры экрана**, при этом цветовые и отражательные характеристики ГЭ задаются с помощью следующих атрибутов:

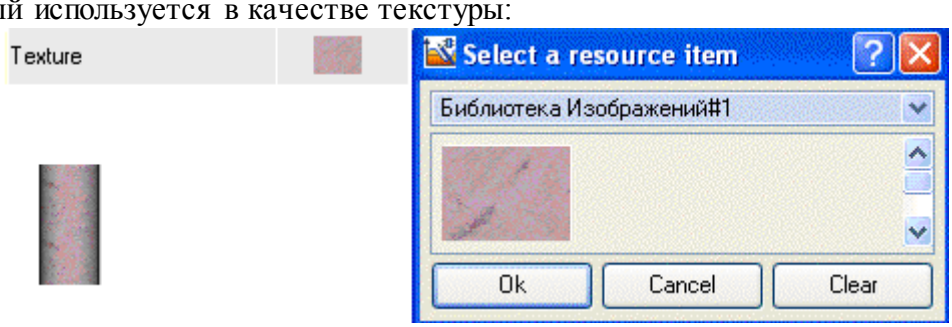
- **Коэффициент яркости диффузного отражения (dc)** – коэффициент (0-100), определяющий яркость диффузно отраженного света. Во внутреннем представлении значения коэффициента диффузно отраженного света равны соответствующим базовым значениям RGB (задаются с помощью атрибута **Базовый цвет**), умноженным на **0.01dc**.
- **Коэффициент зеркального отражения (k)** – количественная характеристика зеркального отражения, задается как число в диапазоне 0-128 (128 соответствует 100-процентному отражению).
- **Коэффициент яркости зеркального отражения (sc)** – коэффициент (0-100), определяющий яркость зеркально отраженного света. Во внутреннем представлении значения

коэффициента зеркально отраженного света равны соответствующим базовым значениям RGB (задаются с помощью атрибута **Базовый цвет**), умноженным на **0.01sc.**

Атрибут **Прозрачность** определяет степень прозрачности ГЭ. Этот параметр задается в процентах (0-100), 0 соответствует абсолютной непрозрачности. На рисунке показаны абсолютно непрозрачный и полупрозрачный цилиндры:



С помощью атрибута **Текстура** поверхность объемного ГЭ может быть текстурирована. При нажатии кнопки в разделе конфигурирования атрибута **Текстура** на экране появляется навигатор ресурсной библиотеки растровых рисунков. В этом навигаторе выбирается рисунок, который используется в качестве текстуры:



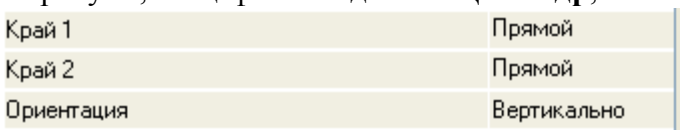
Атрибут **Масштабирование текстуры** при значении **True** масштабирует размер изображения текстуры до размеров ГЭ. При значении **False** текстура имеет оригинальный размер.

Атрибут **Качество (%)** определяет степень прорисовки текстуры. Ниже показана одна и та же текстура с установленным качеством 100, 50 и 20 %:

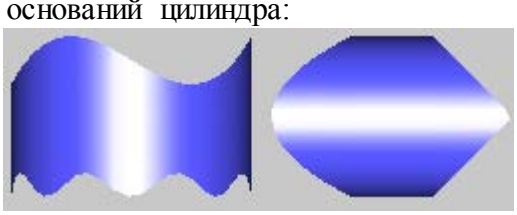
При помощи атрибута **Толщина стенок** можно сделать видимыми ограничивающие объемный ГЭ стенки.

Специфические атрибуты ГЭ 'Цилиндр'

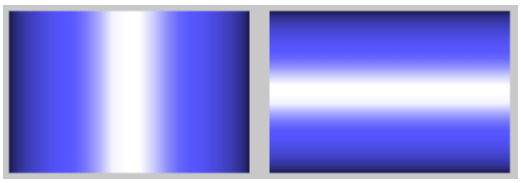
Атрибуты, специфичные для ГЭ **Цилиндр**, показаны на рисунке:



С помощью конфигурирования атрибутов **Край1** и **Край2** можно задавать разнообразную форму оснований цилиндра:



Атрибут **Ориентация** задает расположение образующей для ГЭ **Цилиндр** (не следует путать эту функцию с поворотом ГЭ). По умолчанию установлено значение **Вертикально**, что соответствует вертикальному расположению образующей. Ниже показан один и тот же цилиндр, для которого атрибуту **Ориентация** установлено значение **Вертикально** (левый рисунок) и **Горизонтально** (правый рисунок):

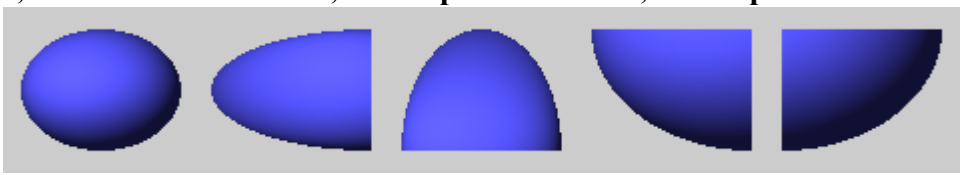


Специфические атрибуты ГЭ 'Сфера'

Специфическим атрибутом для ГЭ **Сфера** является **Отображаемая часть**. Атрибут может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:

Отображаемая часть	Часть
Параметр 1	Полностью
Параметр 2	Часть 1
Отображаемая часть	Сектор
Начальный угол	90
Угол разворота	360

В первом случае параметр **Часть** определяет, какая именно часть должна быть отображена (при выборе опции **Полностью** этот параметр игнорируется). На рисунках показаны варианты отображения ГЭ **Сфера** при следующих параметрах (слева направо): **Полностью; Половина – Часть3; Половина – Часть 4; Четверть – Часть 2; Четверть – Часть 1**.



Во втором случае для ГЭ задается сектор отображения. Угол разворота сектора (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и горизонтальным направлением к правой стороне экрана задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются по часовой стрелке. На рисунках показаны варианты отображения ГЭ **Сфера** при следующих параметрах (слева направо): 90 и 180, 180 и 270.

Специфические атрибуты ГЭ 'Конус'

Специфическими атрибутами для ГЭ **Конус** являются **Отображаемая часть**, **Отношение оснований, %** и **Ориентация**.

Инструменты конфигурирования атрибута **Отображаемая часть** предназначены для задания отображаемой части ГЭ **Конус**. Атрибут может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:

Отображаемая часть	Часть
Параметр 1	Полностью
Параметр 2	Часть 1
Отображаемая часть	Сектор
Начальный угол	90
Угол разворота	360

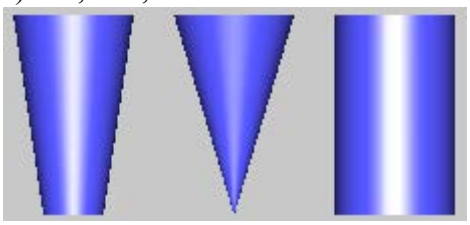
В первом случае параметр **Часть** определяет, какая именно часть должна быть отображена (при выборе опции **Полностью** этот параметр игнорируется). На рисунках показаны варианты отображения ГЭ **Конус** при следующих параметрах (слева направо): **Полностью; Половина – Часть3; Половина – Часть 4; Четверть – Часть 2; Четверть – Часть 1**.

Во втором случае для ГЭ задается сектор отображения. Угол разворота сектора (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и горизонтальным направлением к правой стороне экрана задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются по часовой стрелке. На рисунках показаны варианты отображения ГЭ **Конус** при следующих параметрах (слева направо): 90 и 180, 180 и 270.

Значение атрибута **Отношение оснований, %** задается в процентах и определяет соотношение диаметров оснований (0-100, значение по умолчанию – 50, при значении 100 диаметр меньшего основания равен 1 px, при значении 0 диаметры оснований равны):

Отношение оснований (%)	50

Ниже показан один и тот же конус при следующих значениях данного атрибута (слева направо): 50, 100, 0.



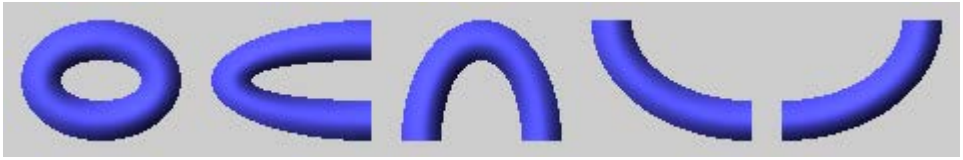
Специфические атрибуты ГЭ 'Тор'

Специфическими атрибутами для ГЭ **Тор** являются **Отображаемая часть** и **Толщина**.

Инструменты конфигурирования атрибута **Отображаемая часть** предназначены для задания отображаемой части ГЭ **Тор**. Атрибут может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:

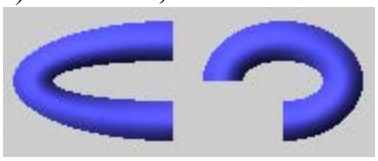
Отображаемая часть	Часть
Параметр 1	Полностью
Параметр 2	Часть 1
Отображаемая часть	Сектор
Начальный угол	90
Угол разворота	360

В первом случае параметр **Часть** определяет, какая именно часть должна быть отображена (при выборе опции **Полностью** этот параметр игнорируется). На рисунках показаны варианты отображения ГЭ **Тор** при следующих параметрах (слева направо): **Полностью; Половина – Часть3; Половина – Часть 4; Четверть – Часть 2; Четверть – Часть 1**.



Во втором случае для ГЭ задается сектор отображения. Угол разворота сектора (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и горизонтальным направлением к правой стороне экрана задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются по часовой

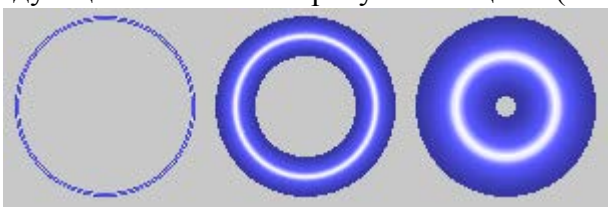
стрелке. На рисунках показаны варианты отображения ГЭ **Тор** при следующих параметрах (слева направо): 90 и 180, 180 и 270.



Атрибут **Толщина** определяет толщину тора и задается как число в диапазоне 0-100 (значению по умолчанию – 20, при значении 0 толщина тора равна 1 px, при значении 100 – 200 px):

Толщина	20

Ниже показан один и то же тор с размерами ограничивающего прямоугольника 90x90 px при следующих значениях атрибута **Толщина** (слева направо): 1, 10, 20.

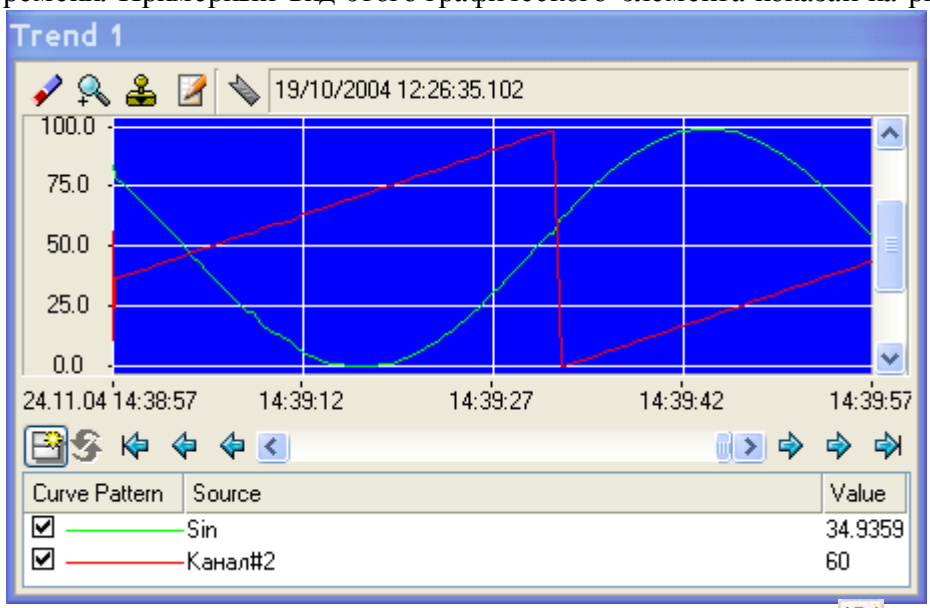


Группа ГЭ ‘Графики’

В эту группу входит единственный ГЭ **Тренд** .

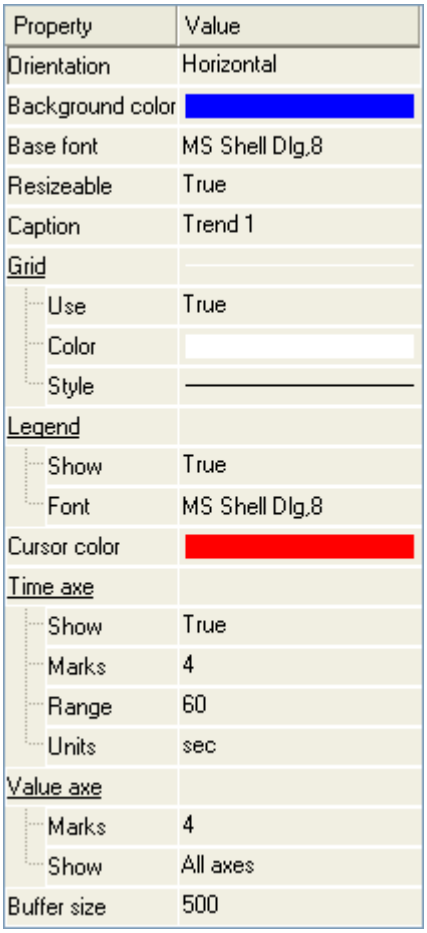
Графический элемент размещается в графическом слое стандартным способом.

ГЭ **Тренд** предназначен для отображения на графике изменения значения привязанных аргументов во времени. Примерный вид этого графического элемента показан на рисунке:



Окно свойств ГЭ содержит вкладки **Осн. свойства** ( ) и **Кривые** ( ).

Вкладка **Осн. свойства** представлена на рисунке:



Атрибут **Ориентация** определяет горизонтальное или вертикальное положение временной шкалы на графике.

Для возможности изменять размеры окна тренда в реальном времени предусмотрен атрибут **Масштабируемый.**

Название окна тренда можно задать с помощью атрибута **Заголовок.**

Раздел **Сетка** позволяет с помощью стандартных инструментов настроить цвет и стиль линий сетки на графике.

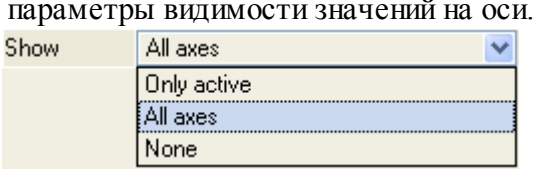
Раздел **Легенда** управляет видимостью легенды под графиком, а также позволяет настроить шрифт для текста легенды.

Раздел **Ось времени** содержит следующие настройки временной оси графика:

- **Показывать** – при установке значения **True** подписи временной оси видны на графике;
- **Разбиение** – определяет, на сколько частей разбивать видимую часть временной оси;
- **Диапазон** – определяет видимый диапазон значений временной оси (0-100);
- **Единицы** – единицы измерения диапазона. Выбираются из меню: **секунда, минута, час, день.**

Раздел **Ось значений** содержит следующие настройки оси значений графика:

- **Разбиение** – определяет, на сколько частей разбивать видимую часть оси значений;
- **Показывать** – задает параметры видимости значений на оси. Выбирается из меню:



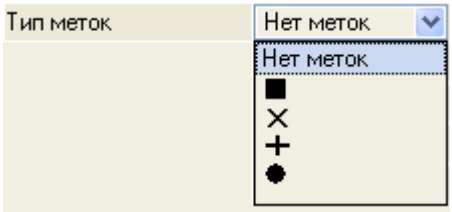
Атрибут **Буфер** задает количество хранимых в памяти значений кривых для вывода на экран в реальном времени (500-32000). Если кривая отображает значения архивируемого в **СПАД** канала, то на график выводятся все записанные значения из архива.

Для того, чтобы получить на экране график, необходимо добавить хотя бы одну кривую на вкладке **Кривые.**

Для добавления кривой (или ее удаления) используется контекстное меню, вызываемое нажатием ПК мыши на строке **Кривые** (или, соответственно, **Кривая**).

Для редактирования свойств кривой предусмотрены следующие инструменты

- **Заголовок** – название кривой.
- **Привязка** – задание привязки кривой к аргументу экрана. При нажатии ЛК в данном поле на экране появляется стандартный диалог выбора аргумента.
- **Цвет** – задание цвета кривой; при нажатии ЛК в этом поле на экране появляется стандартный диалог выбора цвета.
- **Стиль линии** – задание стиля линии; при нажатии ЛК в этом поле выводится выпадающий список стилей.
- **Толщина линии** – задает толщину линии кривой в пикселях.
- **Тип меток** – определяет, нужно ли выделять точку на экране на каждом такте пересчета. Тип выделения выбирается из меню:



- **Стиль линии вне границ** – стиль линии при переходе значений привязанного аргумента за границы, заданные атрибутами **Макс. значение** и **Мин. значение**; при нажатии ЛК в этом поле выводится стандартный список стилей;
- **Макс. значение** – предельное максимальное значение кривой;
- **Мин. Значение** – предельное минимальное значение кривой;
- **Интерполирование** – при установке этому атрибуту значения **True**, график сглаживается методом интерполяции.

После запуска в реальном времени с помощью панели инструментов можно использовать следующие настройки ГЭ **Тренд**:

 – кнопка возврата к исходному масштабу тренда;

 – увеличение. Кнопка имеет два положения – нажата и отжата. При нажатой кнопке включен режим увеличения – при нажатой ЛК нужно выделить прямоугольный диапазон графика для его увеличенного просмотра;

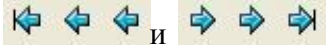
 – перемещать по точкам;

 – настройки кривых. При нажатии этой кнопки выводится диалог настройки кривых тренда, идентичный вкладке **Кривые** окна **Свойства объекта**. Окно предназначено для добавления/удаления кривых тренда, а также изменения их настроек, в реальном времени;

 – перейти к временной метке. При нажатии на эту кнопку появляется диалоговое окно ввода даты и времени. После нажатия кнопки ОК на тренд выводится информация, начиная с указанного времени.

 – показать панель инструментов. Если кнопка не нажата, панель инструментов не отображается;

 – индикатор синхронизации отображаемых данных с архивом СПАД. При синхронизации иконка становится цветной  ;

 – кнопки для перехода по временной оси графика. Соответственно **к началу, на день назад, на час назад** и **на час вперед, на день вперед, к концу**;

Для смещения по осям тренда можно также использовать линейки прокрутки.

Чтобы в реальном времени изменить масштаб по любой из осей надо нажать ЛК в области тренда, а затем нажатием сочетания клавиш **CTRL+<стрелки>** установить требуемый масштаб.

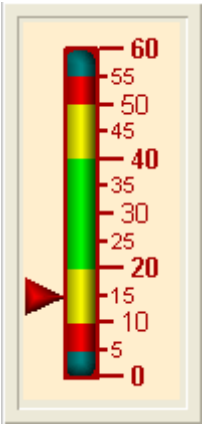
Группа ГЭ ‘Приборы’

В эту группу входят следующие ГЭ:

- **Ползунок**
- **Стрелочный прибор**

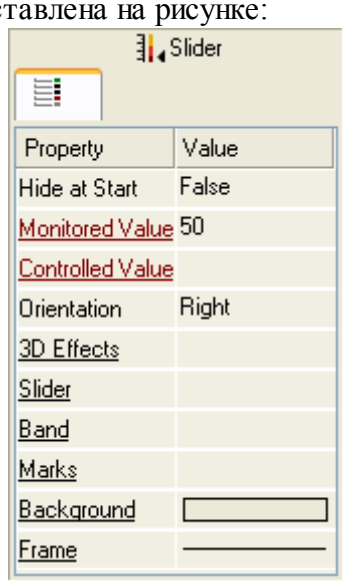
ГЭ ’Ползунок’

ГЭ **Ползунок** используется для задания и/или индикации значения привязанного аргумента, а также для индикации выполнения заданного условия. Примерный вид этого графического элемента показан на рисунке:



Атрибуты ГЭ **Ползунок** конфигурируются на вкладке (**Осн. свойства**) окна свойств данного графического элемента.

Вкладка **Осн. свойства** представлена на рисунке:



Атрибуты **Отображаемая величина** и **Задаваемая величина** определяют функциональное назначение ГЭ.

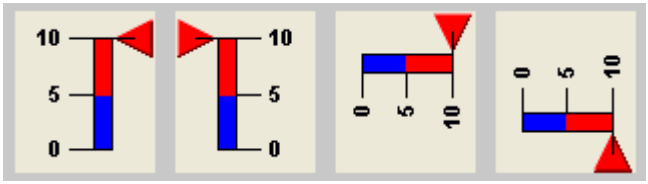
Чтобы ГЭ индицировал значение привязанного аргумента или выполнение некоторого условия, нужно динамизировать его атрибут **Отображаемая величина**).

При раскрытии списка в разделе конфигурирования атрибута **Задаваемая величина** появляется настройка выбора аргумента, чье значение ползунок будет задавать при работе в реальном времени.

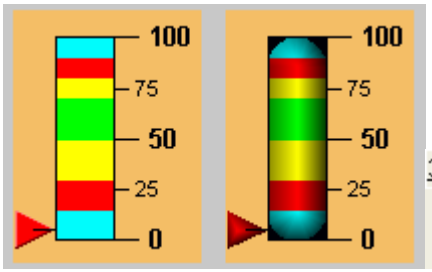
Код доступа – код доступа для возможности управления задаваемой величиной (0-255). Права на доступ задаются для пользователя в виде маски в разделе **Доступ** / **Формы** канала **Пользователь**. При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к управлению разрешен, в противном случае – запрещен.

Атрибут **Положение ползунка** задает положение ползунка относительно шкалы и соответствующее расположение штрихов и чисел на шкале. Ниже показан один и тот же ГЭ со

следующими значениями атрибута **Положение ползунка** (слева направо): **Справа**, **Слева**, **Сверху**, **Снизу**.



В разделе **3D-эффекты** при установке атрибуту **Использовать** значения **True** цветовая полоса и ползунок принимают объемный вид. **Края полосы** могут быть заданы как **Квадратные** или **Круглые**:



3D Effects	
Use	True
Band Edges	Round

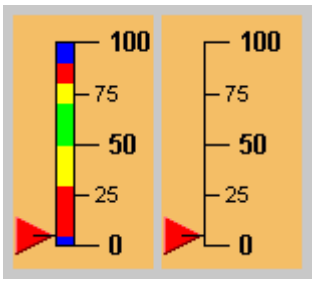
Для отображения ползунка на шкале надо установить атрибуту **Использовать** в разделе **Ползунок** значение **True**. Форма ползунка (**Квадрат**  , **Треугольник**  , **Домик** ), его цвет и размер (в пикселях) определяются соответствующими атрибутами. **Длина риски (%)** на ползунке задается в процентах относительно размера ползунка.

Slider	
Use	True
Type	Square
Color	
Size	31
Hairline Length (%)	50

Раздел **Полоса** содержит следующие атрибуты, определяющие параметры цветовой полосы шкалы:

Band	
Use	True
Width	10
⊞ High Scale Limit	100
⊞ High Limit	90
⊞ High Alarm	80
⊞ High Warning	70
⊞ Low Warning	50
⊞ Low Alarm	30
⊞ Low Limit	5
⊞ Low Scale Limit	0
Color (HW, LW)	
Color (HA, HW), (LW, LA)	
Color (HL, HA), (LA, LL)	
Color >HL, <LL	

Для отображения цветовой полосы шкалы нужно установить атрибуту **Использовать** значение **True** (по умолчанию установлено). Ниже показана шкала с цветовой полосой и без нее:

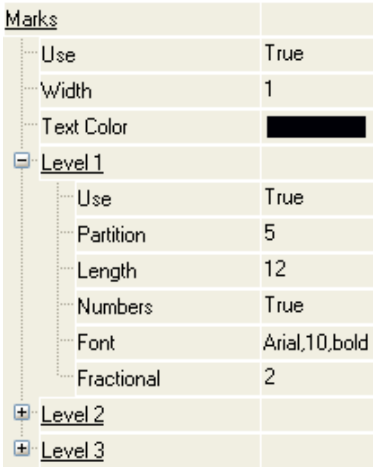


Ширина цветовой полосы задается в пикселях.

Цветовая полоса используется для цветовой кодировки интервалов значений отображаемой/задаваемой величины. По аналогии с каналом, интервалы определяются при помощи задания 6 границ.

Для различных интервалов можно задать свои цвета на полосе.

Раздел **Шкала** содержит параметры разбиения и подписей шкалы.



Атрибуты **Ширина** (этот параметр задается в пикселях) и **Цвет** определяют соответственно ширину и цвет штрихов и текста подписей всех уровней разбиения шкалы. Атрибут **Ширина** определяет одновременно толщину контура цветовой полосы шкалы.

Для шкалы могут быть заданы 3 уровня разбиения. Для каждого уровня (принадлежность к уровню указывает число в наименовании раздела) могут быть заданы следующие параметры:

- **Использовать** – при установке этому атрибуту значения **True** разбиение данного уровня используется, соответствующие штрихи отображаются на шкале.
- **Число делений** – число делений данного уровня.
- **Длина штриха** – длина штриха данного уровня (в пикселях).
- **Показать значения** – при установке этому атрибуту значения **True** отображаются численные значения, соответствующие штрихам данного уровня.
- **Шрифт** – шрифт, используемый для отображения численных значений данного уровня.
- **Десятичные знаки** – количество десятичных знаков в численных значениях данного уровня.

Если атрибуту **Использовать** присвоено значение **False**, остальные атрибуты для этого разбиения игнорируются.

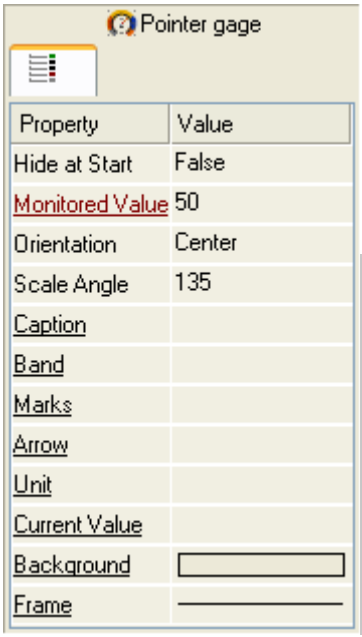
При конфигурировании этих атрибутов нужно учитывать приоритет уровней разбиения:

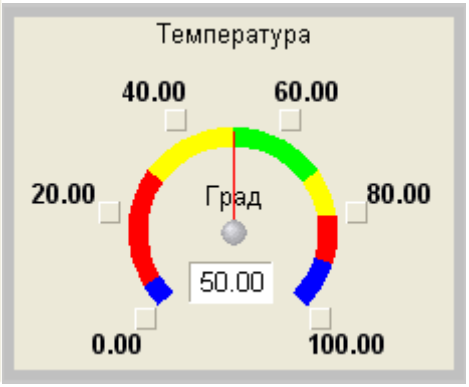
- Атрибут **Число делений N** задает число делений, на которое разбивается каждое деление ближайшего более высокого уровня (**N-1** или **N-2**) при его использовании. Если ни один из более высоких уровней разбиения не используется, атрибут **Число делений N** задает число делений, на которое разбивается весь диапазон шкалы.

ГЭ 'Стрелочный прибор'

ГЭ **Стрелочный прибор** предназначен для отображения текущего значения канала. Цветовая полоса этого ГЭ отображает интервалы значений канала.

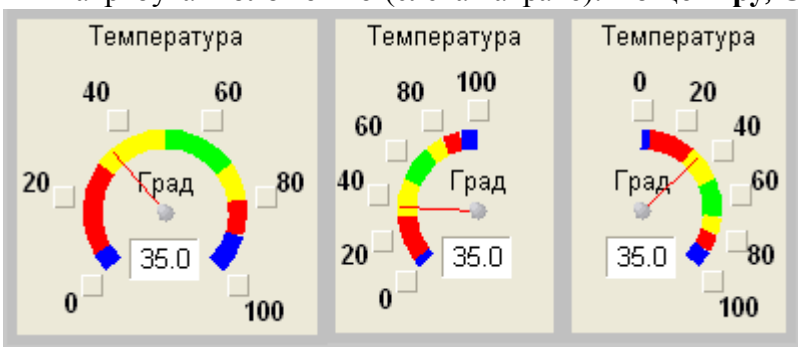
Ниже показаны вкладка **Осн. Свойства** и вид стрелочного прибора:



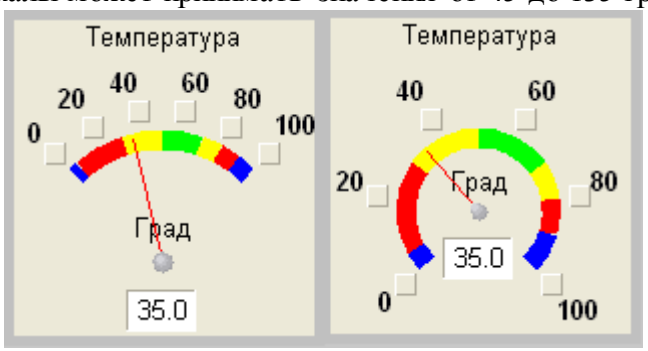


Динамизируемый атрибут **Отображаемая величина** задает значение для отображения на приборе.

Атрибут **Положение** задает ориентацию стрелки прибора относительно шкалы и соответствующее расположение штрихов и чисел на шкале. Ниже показан один и тот же ГЭ со следующими значениями атрибута **Положение** (слева направо): **По центру**, **Справа**, **Слева**.



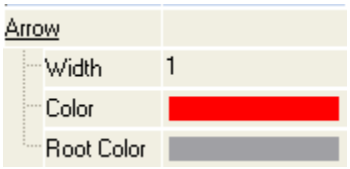
Угол разворота шкалы может принимать значения от 45 до 135 градусов:



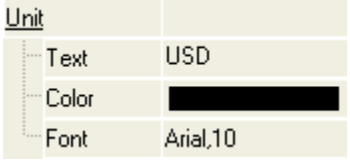
Раздел **Заголовок** содержит атрибуты для стандартные настройки текста заголовка, его цвета и шрифта.

Разделы **Полоса** и **Шкала** имеют атрибуты, аналогичные атрибутам ГЭ **Ползунок**.

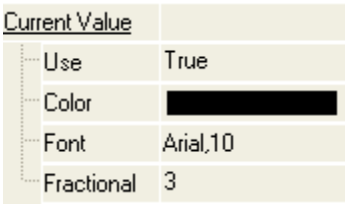
В разделе **Указатель** можно установить параметры стрелки, такие как **Толщина**, **Цвет** и **Цвет основания**. **Толщина** стрелки задается в пикселях.



Настройки раздела **Единицы** определяют параметры выводимой единицы измерения.



При помощи раздела **Индикатор** можно расположить на ГЭ индикатор текущего значения. Для него можно установить свойства **Цвет**, **Шрифт**, **Десятичные знаки**.



Разделы **Фон** и **Рамка** с помощью стандартных инструментов позволяют редактировать внешний вид ГЭ.

Дополнительные сведения для более глубокого изучения по работе в среде ***TRACE MODE 6*** находятся в справочном пособии на русском языке, включенном в состав этой системы (Меню **Справка → Содержание**).

Теперь необходимо изучить основные действия по созданию и редактированию проектов автоматизации в соответствии с выданным преподавателем вариантом и указаниями, содержащимися в файле Учебник по TRACE MODE 6 Вариант X-Y. doc, где X-Y – номер выданного варианта.

Вариант состоит из двух частей (X-Y): первая – общая для варианта X, вторая (Y) – индивидуальная для каждого студента.

Созданный в ходе изучения этого файла свой проект АСУ следует предоставить при защите лабораторной работы.

Контрольные вопросы

1. Структура и состав системы **TRACE MODE 6**.
2. Основные понятия системы **TRACE MODE 6**.
3. Графические элементы системы **TRACE MODE 6**.
4. Порядок работы с различными редакторами, входящими в состав **TRACE MODE 6**

Литература

Перечень основной литературы:

1 Цифровые системы автоматизированного проектирования. SCADA-системы : учебное пособие / И. А. Елизаров, А. А. Третьяков, А. Н. Пчелинцев [и др.]. — Тамбов : Тамбовский государственный технический университет, ЭБС АСВ, 2015. — 160 с. — ISBN 978-5-8265-1469-6. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/63849.html

2 Компьютерные технологии при проектировании и эксплуатации технологического оборудования : учебное пособие / Г. В. Алексеев, И. И. Бриденко, В. А. Головацкий, Е. И. Верболоз. — Саратов : Вузовское образование, 2017. — 171 с. — ISBN 978-5-4487-0004-0. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/65620.html

Перечень дополнительной литературы:

1 Алексеев, Г. В. Возможности интерактивного проектирования технологического оборудования : учебное пособие / Г. В. Алексеев. — 2-е изд. — Саратов : Вузовское образование, 2021. — 263 с. — ISBN 978-5-4487-0377-5. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/79618.html

2 Бойков, В. И. Цифровые системы автоматизированного проектирования / В. И. Бойков, Г. И. Болтунов, О. К. Мансурова. — СПб. : Университет ИТМО, 2010. — 161 с. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/68653.html