ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
Государственное образовательное учреждение
высшего профессионального образования
«ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Утверждаю
Декан АВТФ
_____ С.А.Гайворонский
“___”_____2008 г.

ЛАБОРАТОРНЫЙ ПРАКТИКУМ ПО ИНТЕГРИРОВАННЫМ КОМПЬЮТЕРНЫМ СИСТЕМАМ УПРАВЛЕНИЯ

методические указания по выполнению
лабораторных работ по курсам:
проектирование мехатронных систем,
интегрированные компьютерные системы проектирования
и управления

Издательство
Томского политехнического университета
2008

УДК 658.52,011.56
ББК 32.97

Лабораторный практикум по интегрированным компьютерным системам управления: Учебное пособие для вузов.- Томск: изд-во ТПУ,2008. – 90 с.

Лабораторный практикум предназначен для выполнения студентами циклов лабораторных работ по дисциплинам инновационной образовательной программы магистерской подготовки «Управление в технических системах (мехатронные системы управления)»: проектирование систем управления для гибких автоматизированных производств, разработка интегрированной компьютерной системы управления для гибкого автоматизированного производства. Сост. Е.И. Громаков, В.А. Рудницкий В.А.

УДК 658.52,011.56
ББК 32.97

Рецензент

В.Н. Скороспешкин – доцент кафедры автоматики и компьютерных систем Томского политехнического университета, кандидат технических наук.

Методические указания рассмотрены и одобрены методическим семинаром кафедры интегрированных компьютерных систем факультета АВТ «_____»_____ 2008г.

Зав. кафедрой
проф., доктор техн. наук А.М. Малышенко

Рекомендовано к печати Редакционно-издательским советом Томского политехнического университета

СОДЕРЖАНИЕ

Лабораторная работа №1. Ознакомление с программным комплексом INFINITY SERVER

Цели работы:

1) знакомство с интерфейсом конфигуратора сервера Infinity и модулями, составляющими заданную конфигурацию сервера;
2) создание и редактирование сигналов в конфигураторе.

Теоретическая часть

Программный комплекс Infinity разделен на три части:

- Infinity SCADA , которая предназначена, как следует из названия, для создания SCADA-систем;
- InfintyFactory 1.0 – инструмент для построения MES-систем;
- InfinitySolutions, представляющий собой группу специализированных решений для автоматизации некоторых специфичных отраслей и производств.

Описание Infinity SCADA[1]

Назначение:

- сбор, обработка, хранение производственных и технологических данных;
- объединение технологических и производственных данных в единое информационное пространство;
- оперативное планирование, распределение и контроль состояния ресурсов в режиме реального времени;
- диспетчеризация технологических и производственных процессов;
- формирование отчетности, сводок, балансов.

Надёжность:

- горячее резервирование серверов;
- резервирование потоков данных;
- резервирование узлов;
- резервирование клиентских подключений;
- режим работы 24x7.

Масштабируемость:

- от предприятий с локальным производством, до предприятий с распределенной производственной структурой, оперирующих большим объёмом данных (десятки – сотни тысяч показателей);

[1] В описании лабораторных работ использованы методические материалы С.И.Борисова по SCADA Iconics Genesis.

- свыше миллиона сигналов и показателей, обрабатываемых системой в реальном времени;
- десятки распределенных коммуникационных узлов и диспетчерских пунктов;
- тысячи мнемосхем, отчетов, сводок.

Безопасность:

- управление правами доступа к данным и функциям на серверных и клиентских приложениях;
- интеграция с системой безопасности Windows.

Открытость:

- применение стандартных промышленных протоколов ModBus, CAN, IEC 870.5;
- поддержка OPC, OLE DB, ODBC для обмена данными;
- применение Visual Basic для разработки скриптов;
- Microsoft .NET технология.

Интегрированность:

- интеграция разрозненных информационных систем в единое информационное пространство
- возможность реализации прикладных задач, автоматизирующих сквозные бизнес-процессы предприятия;
- обеспечение информационной основы для анализа данных реального времени и оперативной производственной информации и построения отчетов в масштабе предприятия.

Производительность:

- 80 000 операций ввода/вывода в секунду;
- 40 000 операций чтения/записи истории в секунду;
- Конфигурация до 250 000 сигналов на один сервер.

Удобство использования:

- визуальные средства администрирования и конфигурирования;
- возможность настраивать и управлять профилями пользователей и групп с необходимыми уровнями безопасности, настраивать конфигурационные параметры работы отдельных компонент.

Многоязыковая поддержка:

- интерфейс на русском и английском языках;
- встроенный механизм поддержки дополнительных языков.

Компоненты InfinitySCADA

InfinityServer – OPC-сервер ввода-вывода. InfinityServer выполняет непрерывный контроль технологического процесса, циклически опрашивая системы автоматики и телемеханики. Важным достоинством данного сервера ввода-вывода является модульная архитектура, то есть,

состав модулей может меняться в зависимости от назначения проекта автоматизации. Такая структура экономически выгодна – пользователь выбирает только необходимые модули. Расширение системы заключается только в подключении необходимого модуля и его настройки.

Серверные модули обработки данных выполняют математическую и логическую обработку значений сигналов в соответствии с заданными алгоритмами, а также формируют логику управления. InfinityServer контролирует изменения значений сигналов на соответствие нормативным или пороговым величинам и уведомляет пользователя о нарушениях соответствий, произошедших в ходе технологического процесса.

Важными достоинствами данного сервера являются: наличие поддержки горячего резервирования, обеспечения надежности доставки данных и безопасности доступа к данным.

InfinityHistoryServer – сервер истории, обеспечивает управление историческими технологическими данными. Сервер истории собирает и архивирует значения технологических параметров и сообщения о событиях. Сбор данных ведется по протоколам OPC DA и OPC AE одновременно от нескольких источников.

InfinityHistoryServer поддерживает набор стандартных интерфейсов доступа к историческим технологическим данным, реализуя все преимущества открытых технологий. Доступ OPC-клиентов к технологическим данным поддерживается по протоколам OPC HDA, а доступ к архиву истории по протоколам OLE DB, ODBC. InfinityHistoryServer позволяет использовать исторические данные в задачах MES-систем.

Удобные средства конфигурирования дают возможность оптимально настроить хранение исторических данных в соответствии с решаемыми задачами контроля технологического процесса. Например, гибкая настройка состава позволяет исключить архивирование избыточной информации, а настройка времени хранения истории обеспечивает оптимизацию использования дискового пространства путем своевременной очистки устаревших значений.

Логично организованное одновременное хранение исторических данных в нескольких исторических архивах обеспечивает возможность быстрого доступа к оперативной истории за последние сутки и длительное хранение архивированной информации.

InfinityHMI – инструмент для создания человеко-машинного интерфейса (Human Machine Interface). Графические средства создания интерфейса пользователя. InfinityHMI предоставляет расширенный набор функций по созданию и редактированию графических элементов. InfinityHMI позволяет разрабатывать и исполнять графические мнемосхемы, реализующие отображение информации о ходе выполнения тех-

нологического процесса и управление технологическим процессом в реальном времени.

InfinityHMI предоставляет расширенный набор функций по созданию и редактированию графических элементов. Для ускорения разработки интерфейса диспетчера, разработчик может сам создавать образы технологических объектов или использовать имеющуюся библиотеку элементов. При этом библиотека позволяет сохранять используемые образы с приписанными им функциями анимации и конкретным набором свойств, который изменяется в зависимости от реальных технологических условий и параметров.

Технология псевдонимов динамически позволяет менять источники данных во время исполнения. Для выполнения математических, логических, функциональных и других операций с данными используется встроенный редактор выражений. Редактор поддерживает целочисленные, вещественные, логические и строковые типы данных.

InfinityTrends – клиентское приложение для построения графиков изменений технологических параметров. InfinityTrends позволяет отображать изменения значений технологических параметров в виде графиков или в табличном виде.

InfinityTrends позволяет работать, как в историческом режиме, когда данные об изменении значения сигнала поступают из сервера истории, так и в оперативном режиме (режим самописца), когда данные поступают из сервера ввода-вывода в реальном режиме времени.

InfinityAlarms – клиентское приложение для отображения сообщений о событиях и авариях. InfinityAlarms предоставляет оперативную и историческую информацию в удобном для восприятия виде и ранжировании в зависимости от типа и степени важности сигнала. Полученные сообщения могут сопровождаться голосовым оповещением. Все сообщения могут квитироваться диспетчером.

InfinityAlarms позволяет задавать цвета для сообщений разных типов и важностей. Реализация в виде ActiveX-компоненты, позволяет отображать оперативную и историческую информацию о событиях в других приложениях.

Программа выполнения

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1*: Знакомство с базой данных, порядком запуска и остановки сервера, создание сигналов.*

1.Найдите и запустите

Пуск\Программы\ЭлеСи\Infinity Lite\Инструменты\Управляющий

У Вас появится окно, как показано на рис.1.1.Если под надписью «Сервер Ввода/Вывода» кнопка со стрелкой вверх зеленого цвета, то сервер отключен. Чтобы включить сервер нажмите на эту кнопку. Через

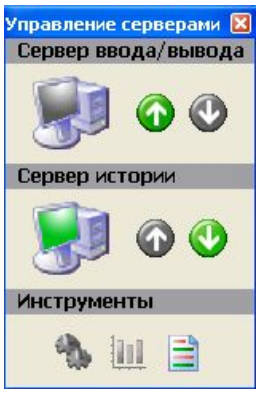


Рис.1.1

несколько секунд сервер запустится.

2. Теперь нужно обнулить базу данных сервера.

Откройте конфигуратор сервера, нажатием на кнопку «Конфигуратор». У вас откроется окно (рис. 1.2).

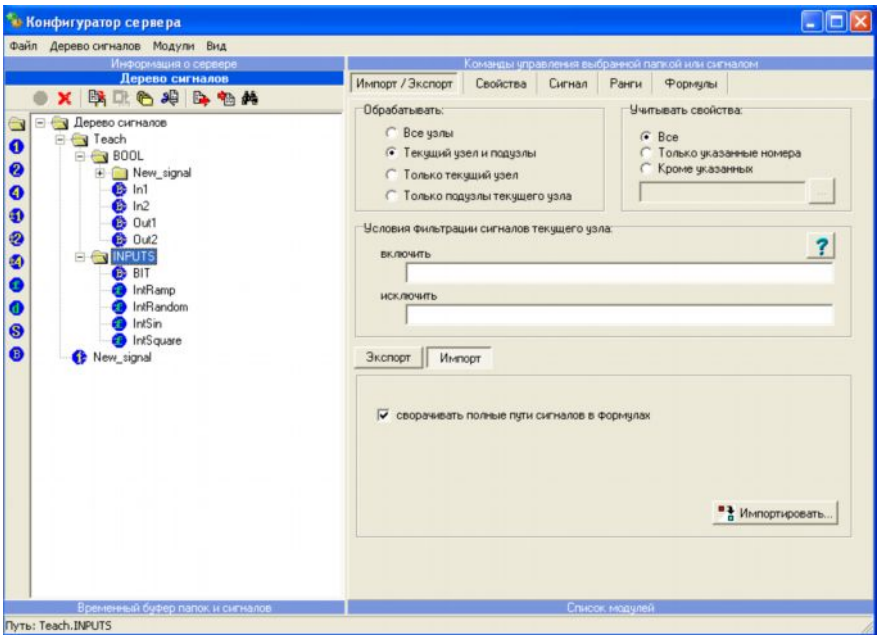


Рис. 1.2

Нажмите **(”Файл→ Загрузить конфигурацию”)** укажите путь к требуемой базе данных (с помощью кнопки **Открыть** справа от строки ввода, в нашем случае это база **abc.EC** в папке **C:\ws326-xx\abc**), и подтвердите свой выбор нажатием кнопки **ОТКРЫТЬ**. Дождитесь окончания загрузки базы.

3.*При следующем запуске ПК Infinity, выполнение пунктов 1-2 обязательно.*
4.Сохраните пустое дерево сигналов под именем **Фамилия.EC ("Файл**. **Сохранить конфигурацию").** С этим файлом вы и будете работать. При следующей работе с ПК Infinity загружайте свою конфигурацию
5. Работа с ПК.

 Все изменения в **Конфигураторе сервера** нужно записывать в файл **Фамилия.EC**, который является базой данных сервера на время выполнения лабораторной работы. База данных должна располагаться в рабочей директории:
C:\WS326-XX\Фамилия.

Собственно, файл **Фамилия.EC** и будет являться одним из результатов лабораторной работы. Чтобы не возникало путаницы с одноименными файлами других пользователей, предлагается повторить пункты с 1-2 и запомнить порядок работы, описанный выше.

***Упражнение 2**: Создание сигналов в Конфигураторе*
1. Откройте **конфигуратор**, запустите свою сохранённую ранее конфигурацию.
Если вы правильно выполнили все пункты, описанные выше, то Дерево Сигналов у вас будет пустым.
2. Создайте в **Дереве сигналов Конфигуратора** папку **Teach** и в ней создайте папки **INPUTS** и **BOOL.** (**Дерево Сигналов→ Создать Папку**).
3. **Создайте в папках сигналы: (Дерево Сигналов→ Создать Сигнал→).**

Имя сигнала	**Тип**
BOOL /**In1**	**Boolean**
BOOL /**In2**	**Boolean**
BOOL /**Out1**	**Boolean**
BOOL /**Out2**	**Boolean**
INPUTS/**Bit**	**Boolean**
INPUTS/**IntRamp**	**Float**
INPUTS/**IntSin**	**Float**
INPUTS/**IntRandom**	**Float**
INPUTS/**IntSquare**	**Float**

Разрешается называть сигналы по-русски, но смешанное именование не приветствуется.

После всех проделанных операций результат должен выглядеть как на рис 1.3:

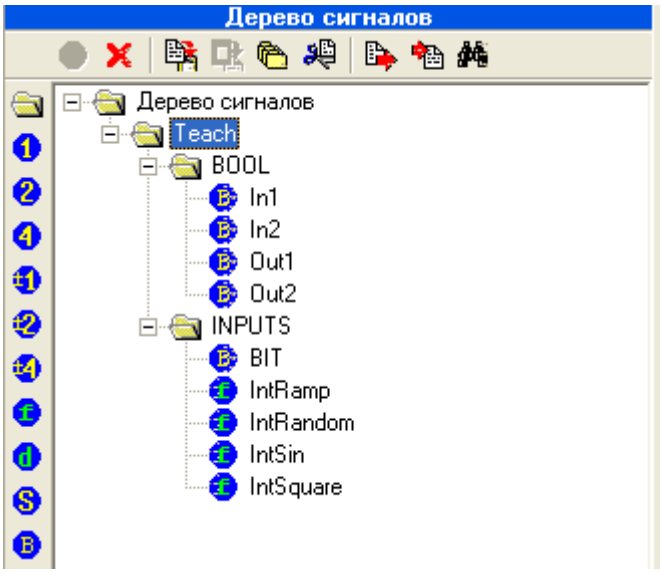


Рис 1.3

3.В **Конфигураторе сервера**, используя вкладку **Свойства**, добавьте **КАЖДОМУ** сигналу стандартное свойство **Quality**, равное **216 (Свойства→ Добавить стандартное свойство и выбираем нужное свойство)** (см. рис. 1.4).

4.Добавьте **КАЖДОМУ** сигналу стандартное свойство **Value**, равное **0** (см. рис.1.4). В меню **команд управления выбранной папки или сигнала** поставьте галочки напротив **Метки времени** и **Права доступа**. Проверьте, есть ли в списке модулей модуль **Логики (Calculation Module)** и модуль **OPC-сервер**, и в случае отсутствия добавьте их в перечисленном выше порядке, используя пункт главного меню **“Модули -> Добавить модуль... Computation Module.., OPC Server Module”**. После этого в окне **Команды управления…** появятся команды, соответствующие добавленным модулям: **Логика** и **OPC.**

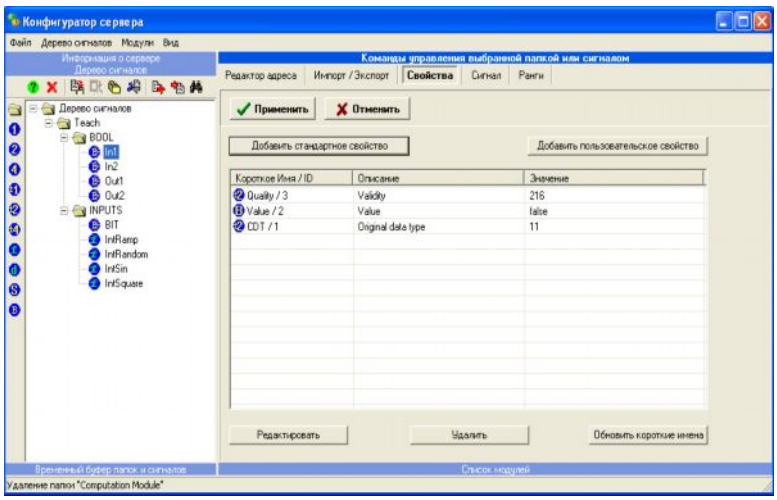


Рис. 1.4

5. Для дальнейшей работы закройте **Конфигуратор** и сделайте перезапуск сервера

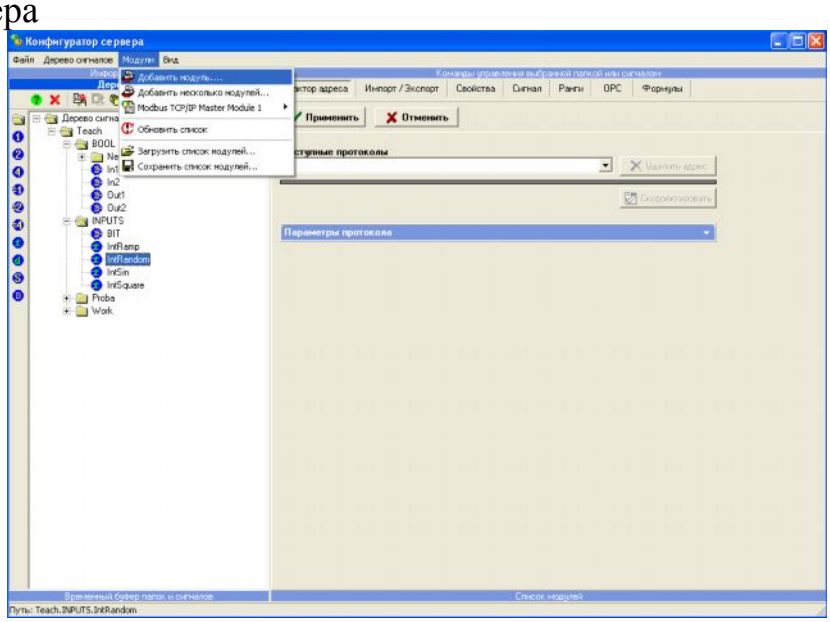


Рис. 1.5

На этом лабораторная работа закончена.

Контрольные вопросы

1. Перечислить компоненты Infinity Scada
2. Для чего используется Infinity Server?
3. Как добавить свойство сигналу в конфигураторе сервера?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе.

Лабораторная работа №2. Ознакомление с пакетом INFINITY HMI. Простые мнемосхемы

Цели работы:

1) ознакомление с InfinityHMI;
2) создание графических объектов, привязка их к OPC сигналам, задание элементарной динамики для этих объектов.

Теоретическая часть

InfinityHMI – инструмент для создания человеко-машинного интерфейса (Human Machine Interface). Графические средства создания интерфейса пользователя. InfinityHMI предоставляет расширенный набор функций по созданию и редактированию графических элементов. InfinityHMI позволяет разрабатывать и исполнять графические мнемосхемы, реализующие отображение информации о ходе выполнения технологического процесса и управление технологическим процессом в реальном времени.

InfinityHMI предоставляет расширенный набор функций по созданию и редактированию графических элементов. Для ускорения разработки интерфейса диспетчера, разработчик может сам создавать образы технологических объектов или использовать имеющуюся библиотеку элементов. При этом библиотека позволяет сохранять используемые образы с приписанными им функциями анимации и конкретным набором свойств, который изменяется в зависимости от реальных технологических условий и параметров.

Технология псевдонимов динамически позволяет менять источники данных во время исполнения. Для выполнения математических, логических, функциональных и других операций с данными используется встроенный редактор выражений. Редактор поддерживает целочисленные, вещественные, логические и строковые типы данных.

Программа выполнения

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1*: Создание объекта «Прямоугольник» с динамикой «Размер», «Цвет плавно», встроенным объектом «Значение параметра».*

1. Загрузите в Конфигуратор файл **Фамилия.EC** из вашей личной директории, сохраните его там под этим же именем, после чего Конфигуратор можно свернуть.
2. Запустите **Infinity HMI**.

Пуск\Программы\ЭлеСи\Infinity Lite\Инструменты\Infinity HMI\ Infinity HMI

3. **С**охраните файл как **Фамилия1.xml.**
4. Нарисуйте прямоугольник в рабочей области окна **InfinityHMI**, как показано на Рис. . Для этого воспользуйтесь инструментом **Прямоугольник/Квадрат** панели Рисование. После выделения объекта рамкой становятся доступными большинство инструментов динамики, которые можно применить к объекту.

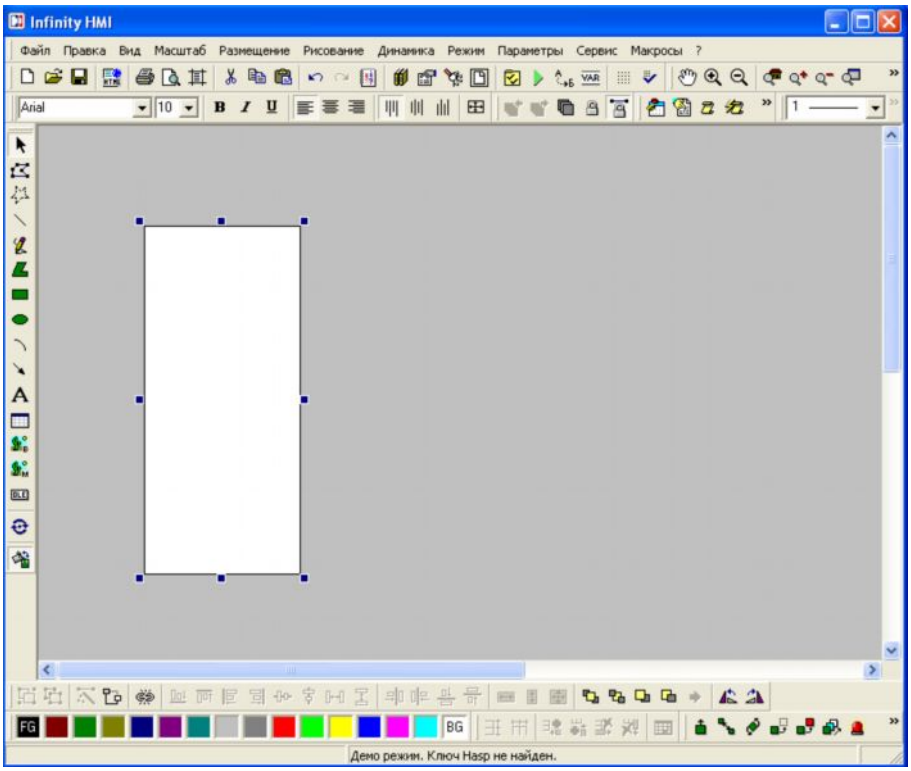


Рис. 2.1. Создание объекта «Прямоугольник»

5. Двойным щелчком по объекту вызовите панель **Свойства объекта**. Первая вкладка позволяет настроить, например, статические визуальные свойства: цвет, форму, градиентную заливку, угол на плоскости и т.п. Рис.
6. Закройте Свойства объекта и выберете инструмент Динамический Размер в панели Динамика. На экран будет выведена диалоговая панель Свойства объекта с выбранной закладкой Размер, показанная на Рис. .

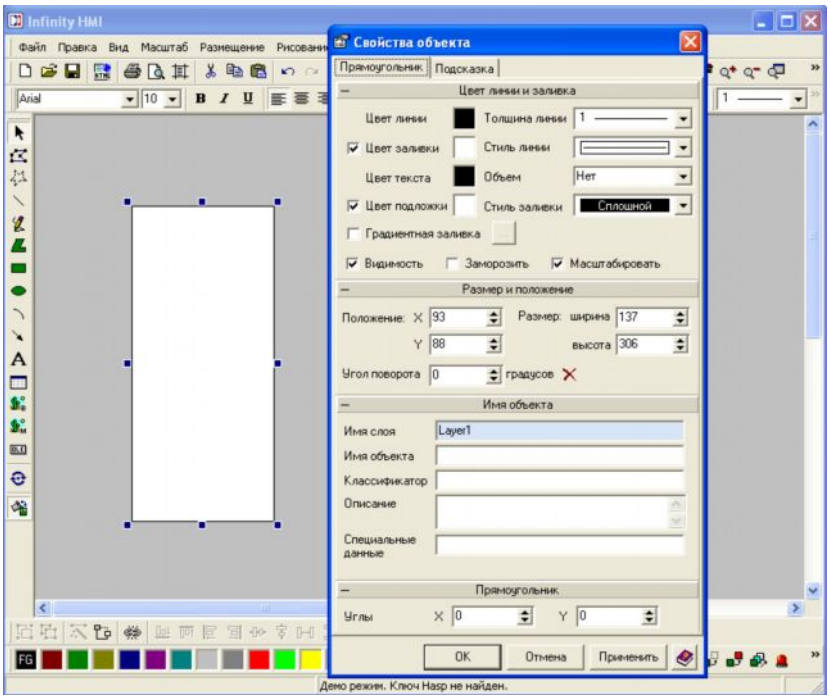


Рис. 2.2. Основные свойства графических объектов

7. Следующими действиями Вы заставите изменяться размер прямоугольника: по вертикали снизу – вверх в зависимости от значения сигнала **IntRamp** Конфигуратора.

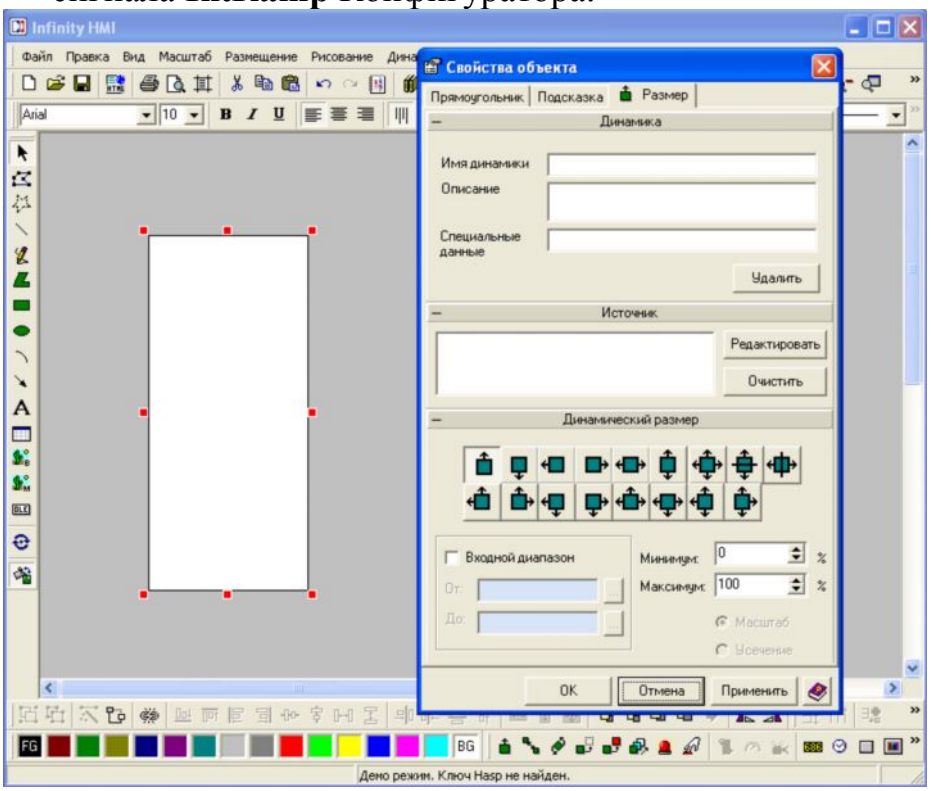


Рис. 2.3. Настройка динамики «Размер»

8. Нажмите кнопку **Редактировать** или два раза по области **Динамика** в диалоговой панели **Свойства объекта**. Появится окно. **Добавить источник данных**. Нажмите кнопку **OPC…**. На экран монитора будет выведено окно **Дерево OPC сигналов**, разверните дерево папок, как показано на Рис. .

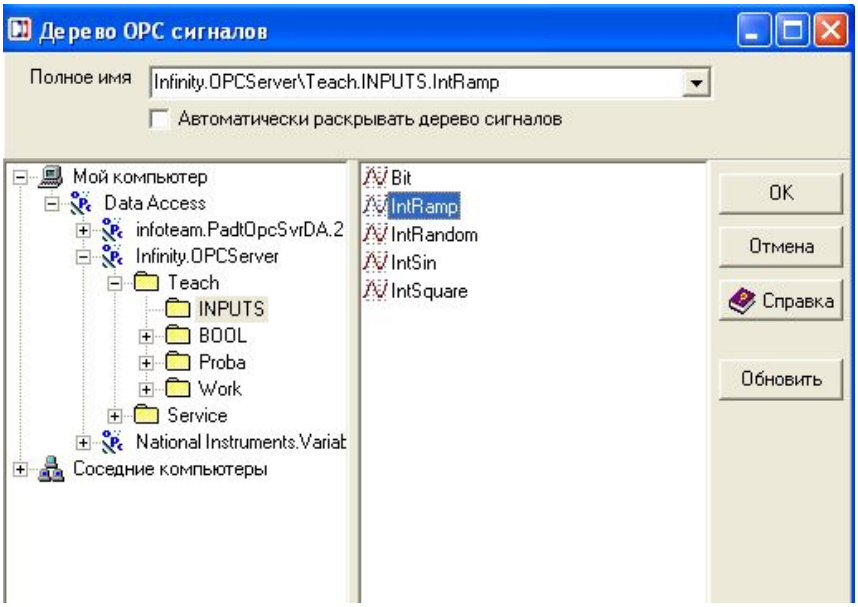


Рис. 2.4. Диалог выбора дерева OPC сигналов

9. Выберите сигнал **IntRamp** и проконтролируйте появление тэгов в окне Полное имя, как показано на Рис. . Нажмите кнопку ОК, после чего окно навигатора будет закрыто. При этом в поле **Источник** вкладки **Размер** диалоговой панели **Свойств объекта** появится имя выбранного OPC тэга, как показано на **Ошибка! Источник ссылки не найден.**.
10. Отметьте флажок **Входной диапазон** и введите значение 1000 в поле **До**:, как показано на **Ошибка! Источник ссылки не найден.**. Таким образом, в режиме **Проект – Старт** размер прямоугольника по вертикали будет изменяться от 0 до 100%, что определяется полями **Минимум/Максимум**, при изменении значения сигнала от 0 до 1000. Нажмите кнопку. **Применить**, затем **ОК**.

Для того чтобы посмотреть динамику движения необходимо, чтобы сигнал Teach.Inputs.IntRamp изменялся по закону пилы. Для этого в конфигураторе сервера это сигнал необходимо настроить на получение значения этого сигнала из контроллера или загрузить модуль Calculation Module (Модуль математико-логических операций) и на закладке Формулы – Процедура по таймеру ввести формулу: if (#0<1000) #0=#0+100; else #0=0;.
Не забудьте сохранить изменения в **Конфигураторе** и перезапустить сервер при

помощи «Управляющего».

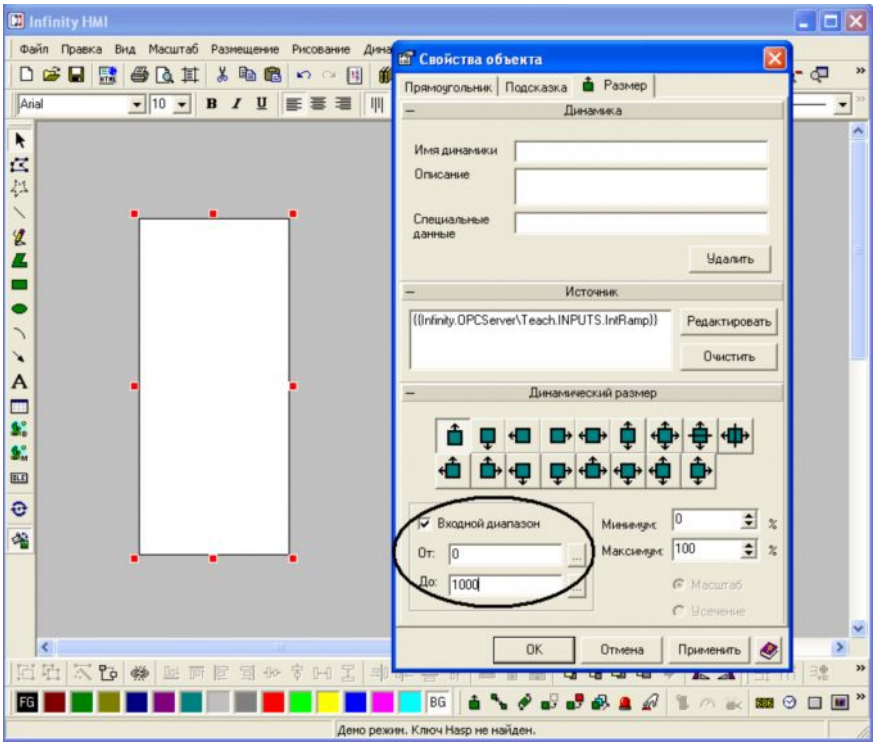


Рис. 2.5. Настройка входного диапазона для динамики «Размер»

11. И в Infinity HMI войдите в режим **Проект-Старт** (или нажмите **F9**). Если все сделано правильно, то Вы увидите изменение размера прямоугольника снизу- вверх. Перейдите в режим **Проект – Стоп**.
12. Выделите объект **Прямоугольник** (обратите внимание, что рамка красного цвета – это означает, что к объекту применена какая-то динамика, в данном случае пока одна – Размер). Выберите инструмент **Динамический аналоговый цвет** в панели инструментов **Динамика**.
13. На экран монитора будет выведена диалоговая панель **Свойства объекта** с выбранной закладкой **Аналоговый цвет**, как на Рис. .

Закладка слева, в Свойствах объекта, уже содержит, настроенную вами, динамику Размер. Для того чтобы просмотреть и откорректировать динамику, примененную к объекту, нужно открыть Свойства данного объекта. Для этого существует два классических для программ, работающих под Windows, способа:

- Двойной клик по объекту.
- Вызов на объекте правым кликом контекстного меню и выбор Свойств объекта.

Выделение объекта, а затем применение, к примеру, инструмента динамики **Размер**, приведет к появлению у этого объекта еще одной динамики Размер.

Вывод: сверяйте настройку динамики с приведенными примерами, вкладок не должно быть больше (классическая ошибка). Для удаления ненужной динамики на вкладках Свойств объектов есть кнопка. Удалить.

14. Установите соединение с тегом **IntRamp** для заливки объекта выберите начальный цвет – красный, а конечный цвет – синий. Введите входной диапазон от 0 до 1000. Проверьте настройку динамики в соответствии с Рис. , после чего закройте **Свойства.** Перейдите в режим **Проект – Старт** и наблюдайте плавное изменение цвета от красного к синему. Очевидно, что при размере 50% (то есть при значении **IntRamp=500**) происходит равное смешивание цветов.

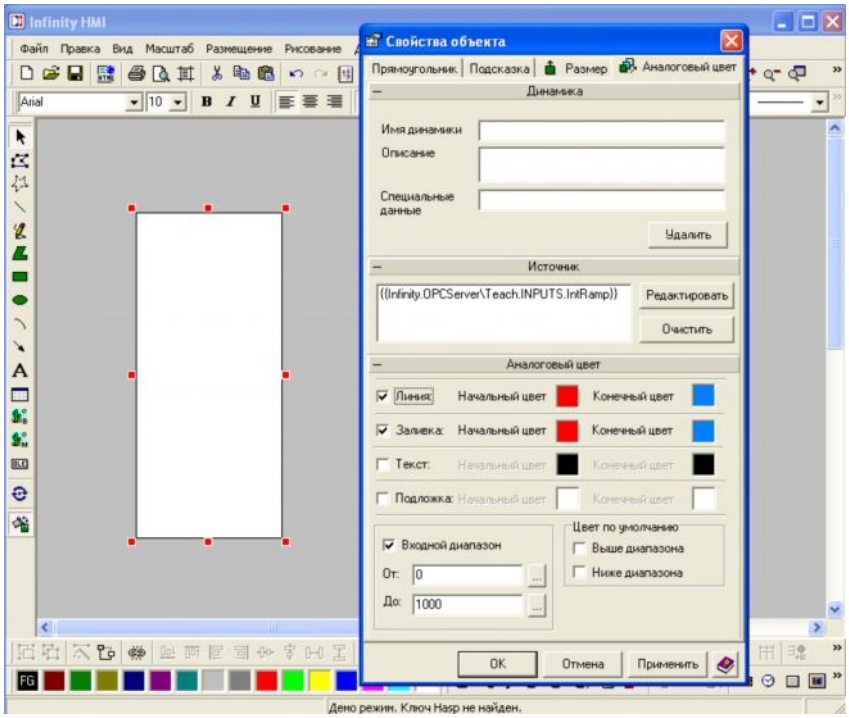


Рис. 2.6. Настройка динамики «Аналоговый цвет»

15. Перейдите в режим **Проект – Стопобъекта**.
16. Для визуализации численного значения сигнала **IntRamp** выберите **Динамический объект** 888 - значение параметра на панели инструментов **Динамика**.

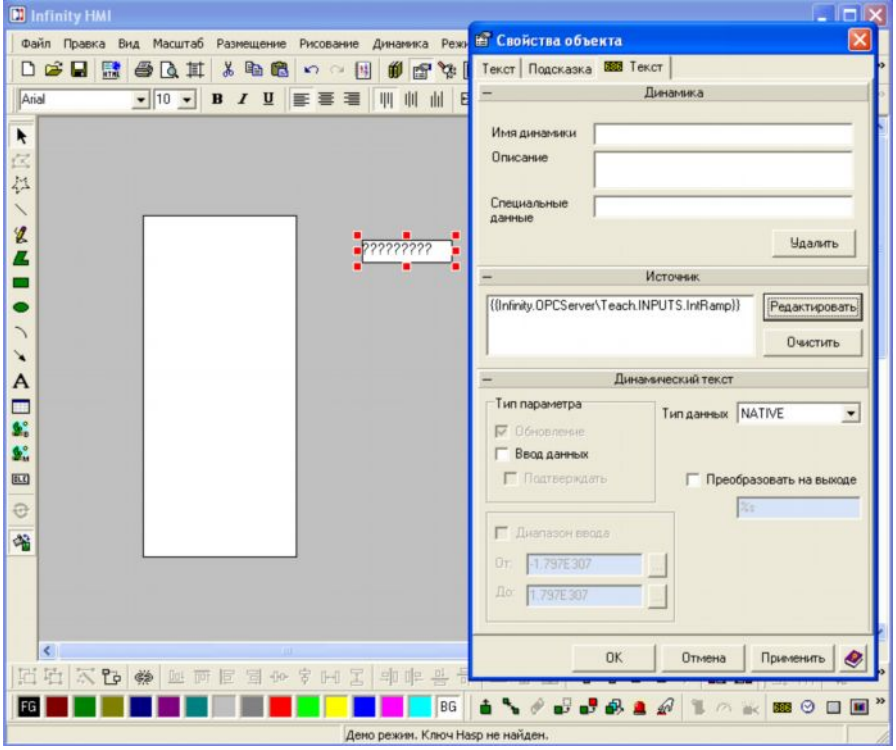


Рис. 2.7. Настройки связи объекта «Значение параметра» с OPC сигналов

17. Щелкните мышью справа от объекта **Прямоугольник**. Настройте **Свойства объекта** как на рисунках (Рис.).
18. Примените настройки и закройте **Свойства объекта.** Перейдите в режим **Проект - старт.** Если все сделано, верно, то в поле объекта Значение параметра будет отображаться численно.
19. Щелкните мышью справа от объекта **Прямоугольник**. Настройте **Свойства объекта** как на рисунках (Рис.).

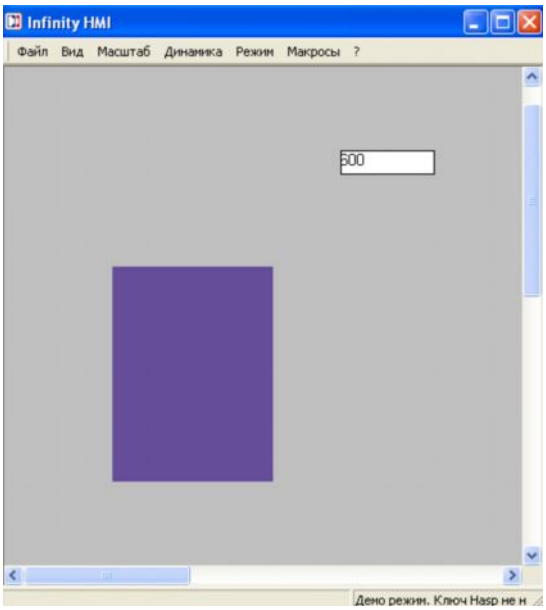


Рис. 2.8. Мнемосхема с прямоугольником в режиме исполнения

20. Примените настройки и закройте **Свойства объекта.** Перейдите в режим **Проект - старт.** Если все сделано, верно, то в поле объекта Значение параметра будет отображаться численное значение сигнала

IntRamp в выбранном вами формате (**Ошибка! Источник ссылки не найден.**).

21. В режиме **Проект – Стоп** сохраните файл под именем **Фамилия1. xml**.

Упражнение 2. *Изучение встроенного динамического объекта Кнопка, работа с битовыми сигналами*

Запустите Конфигуратор сервера **и создайте в** Дереве сигналов **в своей папке** Teach **папки** Proba **и** Work **(не забудьте вложенные папки** Pump **и** Valve).

22. Создайте в папках сигналы:

Proba →
Сигнал **Состояние1**, тип **Boolean.**
Сигнал **Состояние2**, тип **Boolean.**
Сигнал **Состояние3**, тип **Boolean.**
Сигнал **Состояние4**, тип **Boolean.**
Сигнал **Управление1**, тип **Boolean.**
Сигнал **Управление2**, тип **Boolean.**
Сигнал **summa**, тип **Word.**
Work →
Сигнал **Reset**, тип **Boolean.**
Сигнал **Level**, тип **Float.**
Сигнал **Tank**, тип **Word.**
Work\Pump→
Сигнал **Control**, тип **Boolean.**
Сигнал **In**, тип **Boolean.**
Сигнал **Status**,тип **Boolean.**
Work\Valve→
Сигнал **Control**, тип **Boolean.**
Сигнал **In**, тип **Boolean.**
Сигнал **Status**, тип **Boolean.**
Сигнал **Level**, тип **Float.**

23. Добавьте **КАЖДОМУ** сигналу стандартное свойство **Quality**, равное **216** и стандартное свойство **Value**, равное **0.** В меню **команд управления выбранной папки или сигнала** поставьте галочки напротив **Метки времени** и **Права доступа**.
24. Закройте Конфигуратор сервера и перезапустите сервер при помощи Управляющего.
25. Запустите **InfinityHMI** и создайте новый файл, сохранив его под именем **Фамилия3.xml.**
26. Выберите в панели инструментов **Динамика** объект **Кнопка**  и щелкните на рабочей области экрана.

27. Перейдите на первую вкладку **Свойств объекта**, которая описывает статические параметры объекта, и настройте ее следующим образом, Рис. .

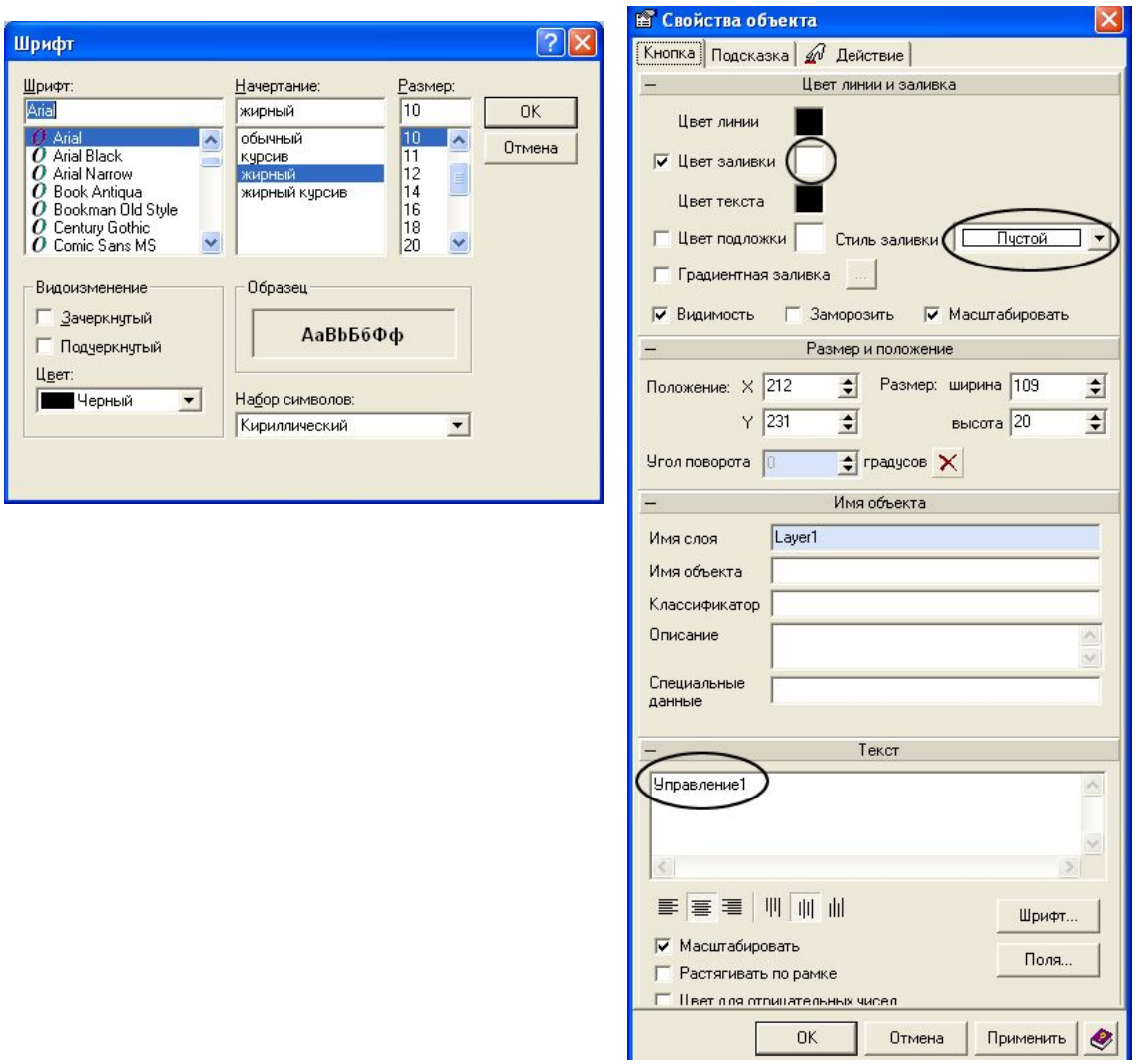


Рис. 2.9. Настройка основных параметров объекта «Кнопка»

28. Вкладку **Действие** настройте следующим образом: если левая клавиша мыши нажата, **а затем отпущена**, то **Значение1** (единица) передается в сигнал **Управление1**, если нажать кнопку во второй раз, то в этот сигнал передается **Значение2** (ноль), (Рис.). **Настоятельная рекомендация:** перед выполнением этой настройки прочитать пояснение ниже.

Для соединения с тегом можно в поле Источник прописать тег вручную, но незначительная ошибка (пропущена точка, буква и т.п.) приведет к тому, что тег с пропущенной, например, точкой не будет найден, так как такой тег (без точки) в симуляторе не создавался. Пользуйтесь во время ввода тега кнопкой Редактировать-OPC… и ищите нужный тег в Дереве OPC сигналов. После того, как тег найден и подсвечен одиночным щелчком мыши, нажимайте кнопку OK: тег появится в Свойствах объекта именно в том поле, где до вызова стоял курсор. В ситуации, когда созданный сигнал не появился в Дереве сигналов OPC, нажмите кнопку Обновить.

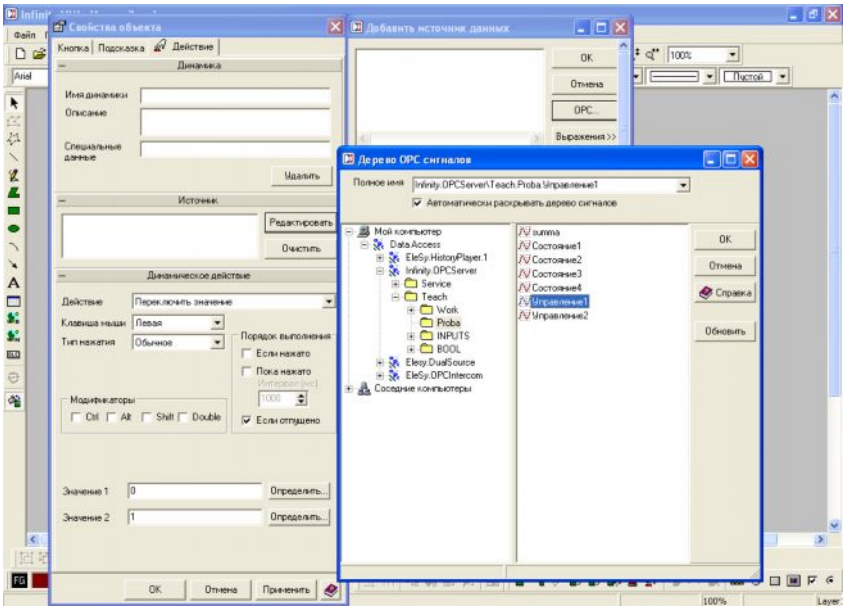


Рис. 2.10. Настройка объекта «Кнопка» на связь с сигналом

29. Для визуализации состояния **Управление1** создайте объект, к которому примените динамику **Цвет**: нарисуйте прямоугольник с темно-серым цветом заливки и размерами, близкими к размерам объекта **Кнопка** и поместите его над кнопкой. Примените к объекту **Прямоугольник** динамику **Цвет**:  .
30. Нажмите кнопку **Добавить** и добавьте тег **Управление1**. После этого станет доступной настройка цвета объекта. Если сигнал **Infinity.OPCServer\Teach.Proba.Управление1** равен 1 (True), то **Цвет заливки** зеленый (Рис.). (Всегда применяйте ярко – зеленый цвет).
31. Еще раз нажмите кнопку **Добавить** и добавьте тег **Infinity.OPCServer\Teach.Proba.Управление1** еще раз. Измените **Цвет заливки** на красный. Также поменяйте **Изменить цвет, если False** (равно 0) (Рис.).

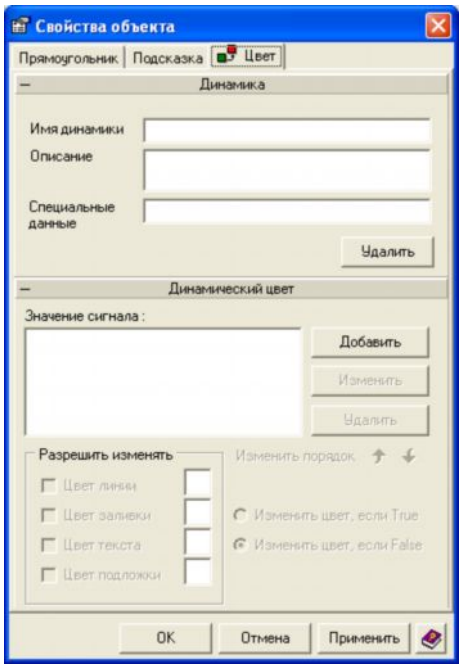


Рис. 2.11. Настройка индикатора-сигнализатора

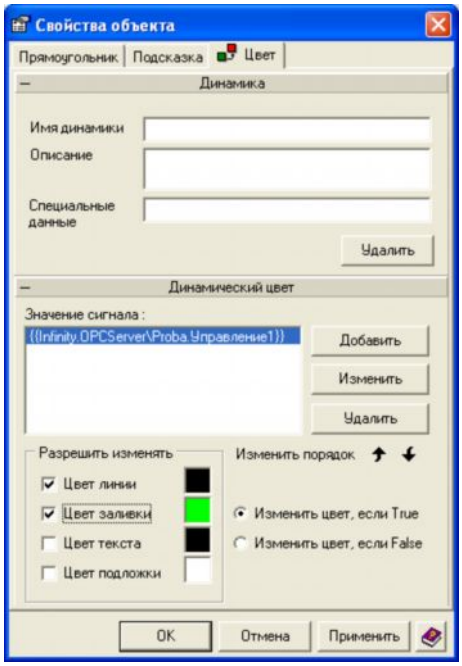


Рис. 2.12. Добавление изменения цвета индикатора на зеленый

ЦВЕТОВАЯ ПОЛИТИКА!

Раскраска объектов носит название цветовой политики. Пусть те объекты, которые являются индикаторами состояния сигналов (например, объект Прямоугольник) будут в режиме Разработка нейтрального цвета (примем серый и темно – серый цвета). В режиме Проект – Старт при единичном значении сигнала – ярко – зеленого, при нулевом – красного. Это и будет лабораторная цветовая политика, которой надо строго придерживаться. Проверяйте ваши объекты на соответствие цветовой политики и, если необходимо, приводите в соответствие!

32. Закройте **Свойства объекта**, перейдите в режим **Проект – Старт** и проверьте работу мнемосхемы. Первое нажатие и отпуск кнопки переводит **Управление1** в единицу, второе нажатие и отпуск переводит **Управление1** в ноль.

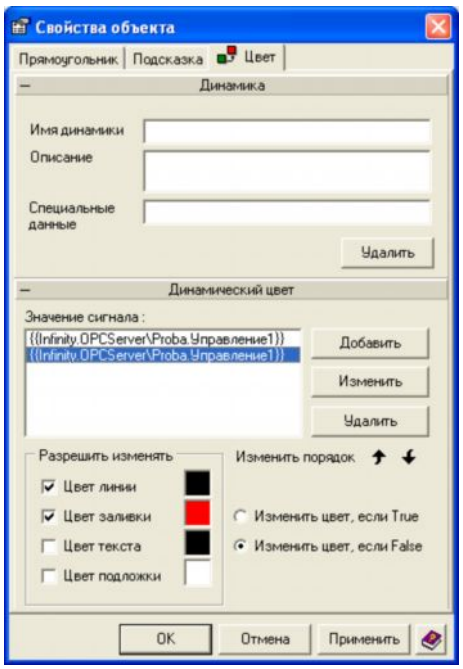


Рис. 2.13. Добавление изменения цвета индикатора на красный

Выходите из режима Проект – Старт корректно, то есть перед выходом верните сигнал в начальное (в данном случае нулевое) состояние. Выполняя лабораторные работы, возьмите за правило перед выходом из режима Проект – Старт возвращать сигналы в начальное состояние.

33. Выберите объект **Значение параметра** . В качестве источника данных используйте тег **Infinity.OPCServer\Teach.Proba.summa**. Настройте вкладку **Текст** по своему усмотрению, после чего закройте **Свойства объекта**.
34. Примените к объекту **Значение параметра** инструмент **Динамическое действие** и в нем операцию **Передать значение** (Рис.). В поле **Значение (пока нажато)** введите выражение суммирования тегов: **x={{Infinity.OPCServer\Teach.Proba.summa}}+{{Infinity.OPCServer\Teach.Proba.Управление1}},** соблюдая последовательность:

- Нажмите кнопку **Определить**, отмеченную на Рис. .

Рис. 2.14. Строка ввода значения для операции «Передать значение»

- Далее кнопку **Выражения**. Введите **x=** в окне ввода.
- Выберите тег **Infinity.OPCServer\Teach.Proba.summa**.
- Нажмите кнопку **Арифметические** и в всплывающем меню выберите операцию **Сложение**. (Знаки арифметических операций удобнее вводить с клавиатуры). Далее выберите тег **Infinity.OPCServer\Teach.Proba.Управление1**.
- Закройте окошко нажатием кнопки **ОК**.

Выражение **x={{Infinity.OPCServer\Teach.Proba.summa}} + {{Infinity.OPCServer\Teach.Proba.Управление1}}** – это выражение инкремента. В первый момент в сигнал summa запишется значение сигнала Управление1, в следующий момент времени к измененному summa снова добавится значение сигнала Управление1 и так до переполнения summa (так как summa имеет тип WORD, то в него можно записать число 65535).

35. Проверьте вкладку **Действие** объекта **Значение параметра** на соответствие Рис. .
36. Измените время подсчета формулы с 50 мс. На один раз в секунду.

Выражение будет выполняться по щелчку на объекте Значение параметра, а условия применения щелчка могут быть разными, если настроить эти условия, так как показано на Рис. , то выражение будет выполняться раз в 50 миллисекунд все время с момента перехода экранной формы в режим Проект – Старт. Действительно, условие – Пока нажато, выполниться сразу, так как отмечен флажок Начальное состояние Нажато, а тип нажатия С фиксацией. Напрашивается аппаратная аналогия с кнопкой, которая после подачи напряжения (режим Проект - Старт) переходит в нажатое состояние (отмечен флажок Начальное состояние Нажато), фиксируется в нем (тип С фиксацией) и через ее замкнутые контакты происходит передача сигналов (выражения из поля Значение (пока нажато)) в схему (в summa). Оперируя подобными аналогиями легко понять логику работы динамики Динамическое действие при других настройках полей Тип и Порядок выполнения. Для выполнения выражения можно выбрать любой объект мнемосхемы, можно создать новый, но по смыслу в данном случае удобно применить динамику пересчета выражения на том объекте, где этот пересчет индицируется, то есть на объекте Значение параметра, который привязан к сигналу summa. Это облегчает поиск ошибок и отладку динамики мнемосхемы.

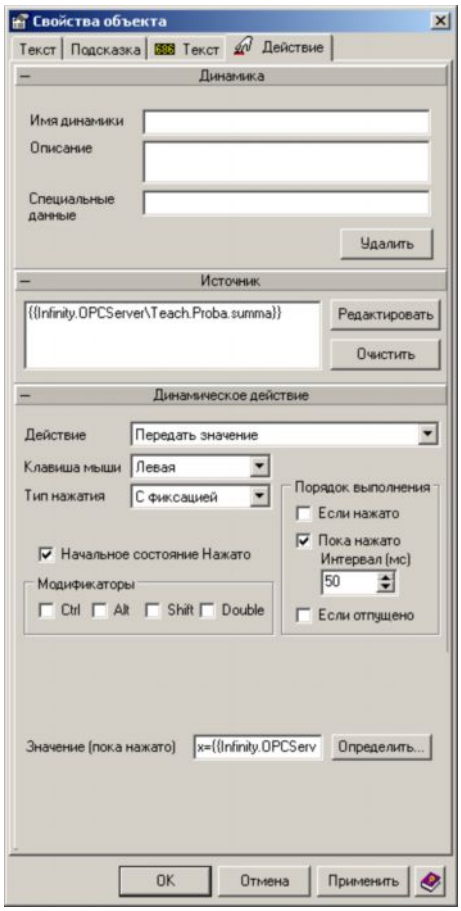


Рис. 2.15. Настройка динамики «Действие» для задания расчетной Формулы

37. Закройте **Свойства объекта** и проверьте мнемосхему в режиме **Проект – Старт**. Нажатие на кнопку записывает единицу в сигнал **Управление1** и сразу же происходит увеличение значение сигнала **summa** в соответствии с выражением.
38. Модифицируйте выражение инкремента так, чтобы нажатие на кнопку **Управление1** приводило к пятикратному увеличению значения summa (0-5-10-15-…) один раз в 1 сек.
39. Создайте кнопку, при нажатии на которую, будет обнуляться значение сигнала Summa.
40. Создайте второй объект **Кнопка** с именем **Управление2** и такими же свойствами, как объект **Кнопка** с именем **Управление1**, но в качестве тега выберите **Infinity.OPCServer\Teach.Proba.Управление2** .
41. По предыдущей аналогии создайте объект **Прямоугольник** и примените к нему динамику **Цвет**, для визуализации значения сигнала **Управление2** (Рис.)
42. Модифицируйте выражение инкремента так, чтобы нажатие на кнопку **Управление2** приводило к обратному инкременту сигнала

summa (величина подсчета единичная). Внешний вид мнемосхемы в режиме **Проект – Старт** показан на Рис. .

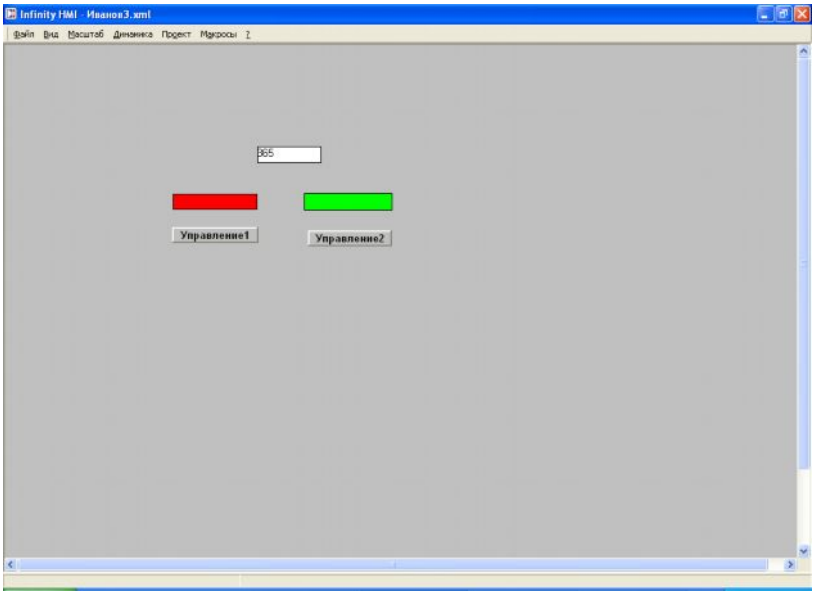


Рис. 2.16. Внешний вид мнемосхемы в режиме исполнения

Сохраните файл в личной директории под именем **Фамилия3.xml**. Результат лабораторной работы показать преподавателю.
На этом лабораторная работа закончена

Контрольные вопросы

1. Что такое Infinity HMI?
2. Как создать объект «Кнопка»?
3. Как применить динамику «Цвет» к объекту?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе

Лабораторная работа №3. Ознакомление с пакетом INFINITY HMI. Внутренние каналы управления (формулы, локальные переменные)

Цели работы:

1) закрепление навыков работы с интерфейсом Infinity HMI при создании внутренних каналов (формул, локальных переменных);
2) управление элементами экрана SCADA- системы.

Теоретическая часть

В SCADA различают внешние и внутренние каналы ввода/ вывода информации на элементы экранных форм (скриншоты) АСУТП. Внешние каналы используются диспетчером для ввода информации от измерительных приборов, подсистем аварийной защиты и сообщений от операторов, а также для вывода сигналов управления исполнительными устройствами и при передаче команд и сообщений операторам.

Внутренние каналы являются виртуальными и создаются разработчиками АСУТП для решения задач автоматизированного управления технологическими объектами. Такие каналы формируются на основе алгоритмов вычислений и логических операторов. И внутренние и внешние каналы ввода информации многими поставщиками используются при формировании их маркетинговой политики. Разработчик АСУТП должен выяснить у поставщика лицензионные ограничения по всей совокупности каналов ввода вывода.

Программа выполнения

Лабораторная работа содержит задания в виде упражнений по подготовке внутренних каналов управления в SCADA системе (каналов логической обработки информации). Все упражнения обязательны для выполнения.

***Упражнение 1**: Создание канала ввода/вывода информации скриншота «Графическая мнемосхема «Булевы функции»»*

1. Запустите **InfinityHMI** и создайте в личной директории новый файл **Фамилия4.xml**.
2. В меню **Сервис** выберите пункт **Настройки** или нажмите кнопку . Настройте данное окно как показано на **Ошибка! Источник ссылки не найден.**. Данным действием Вы включите сетку для удобства выполнения работы. Цвет сетки поставьте (или оставьте) серый.

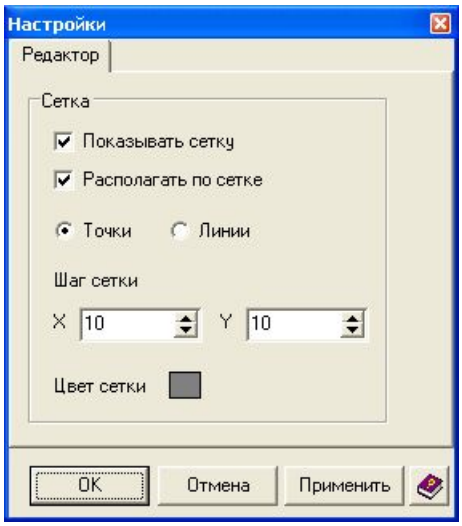


Рис. 3.1. Настройки сетки при создании мнемосхемы

3. В главном меню **Вид** отметьте **Цветовую палитру** (**Ошибка! Источник ссылки не найден.**). Цвет палитры выберите темно – серый. После этого все объекты типа **Прямоугольник** и **Эллипс/Круг** будут иметь темно–серую заливку и палитру, если она мешает можно убрать.

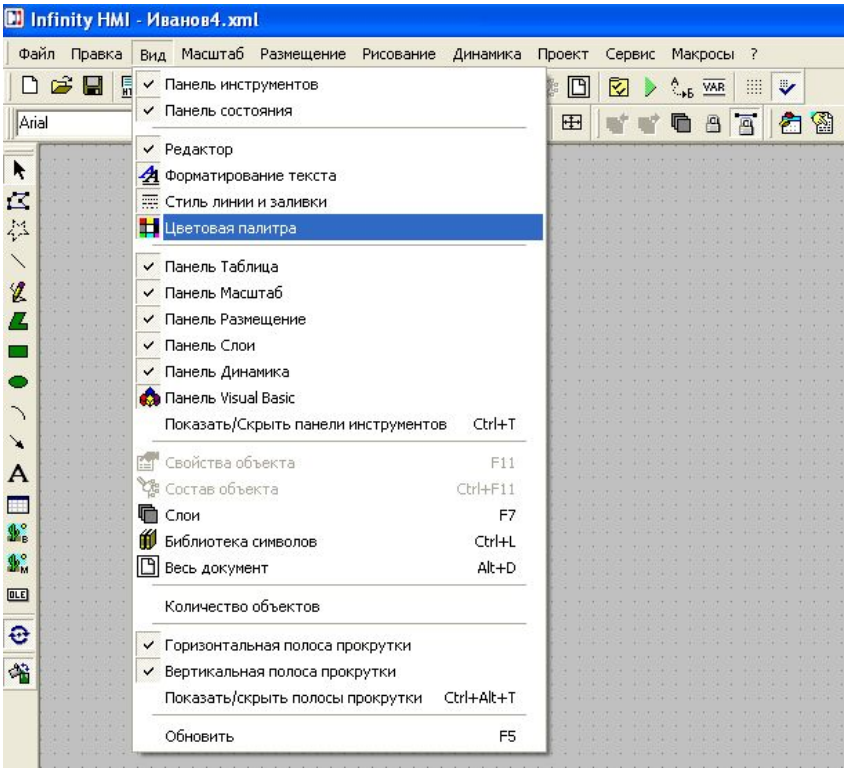


Рис.3.2. Выбор цветовой палитры

4. Нарисуйте объект Логический элемент И. Размеры приблизительные, визуально около 20мм × 40мм (**Ошибка! Источник ссылки не найден.**.3).

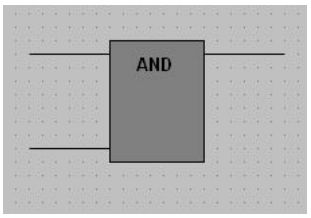


Рис. 3.3. Вид элемента «AND»

5. Создайте объект **Кнопка** и настройте **Свойства объекта**, как на **Ошибка! Источник ссылки не найден.**

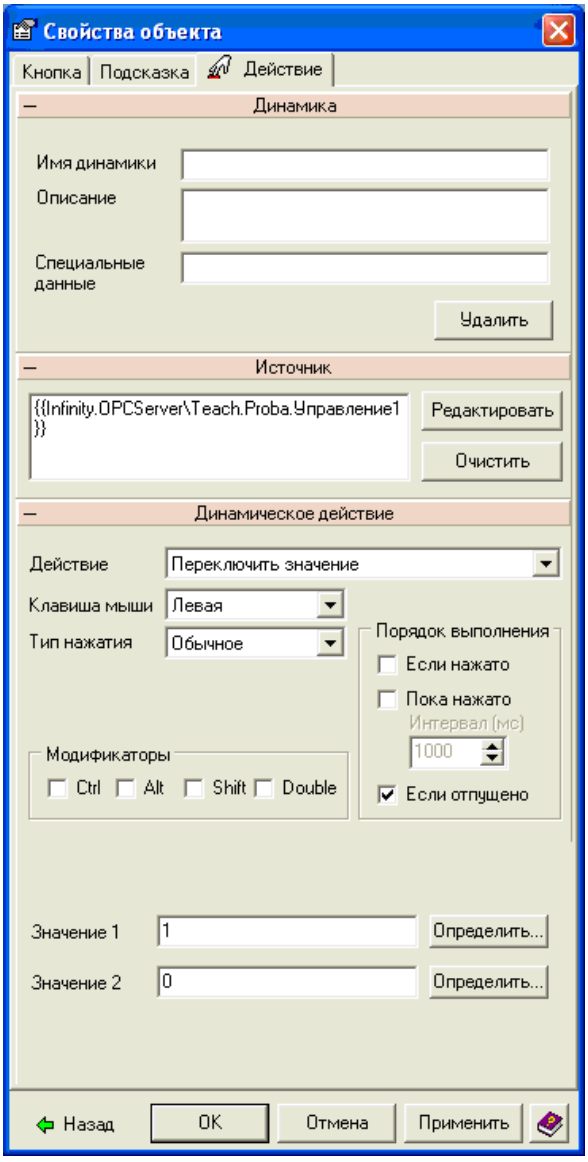
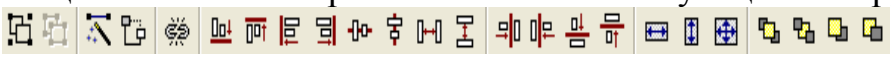


Рис.3.4. Настройка свойств для кнопки, передающей значения в сигнал «Управление1»

6. Повторите действия шага **Ошибка! Источник ссылки не найден.** для второго сигнала **Управление2**. Внешний вид экранной формы пока-

зан на **Ошибка! Источник ссылки не найден.**. Для эффективного размещения объектов применяйте соответствующие инструменты:



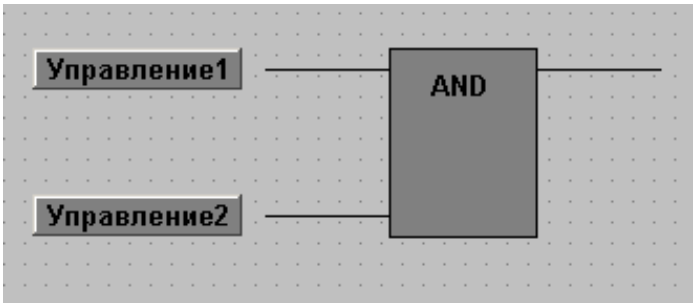


Рис. 3.5. Вид элемента «AND» с кнопками управления

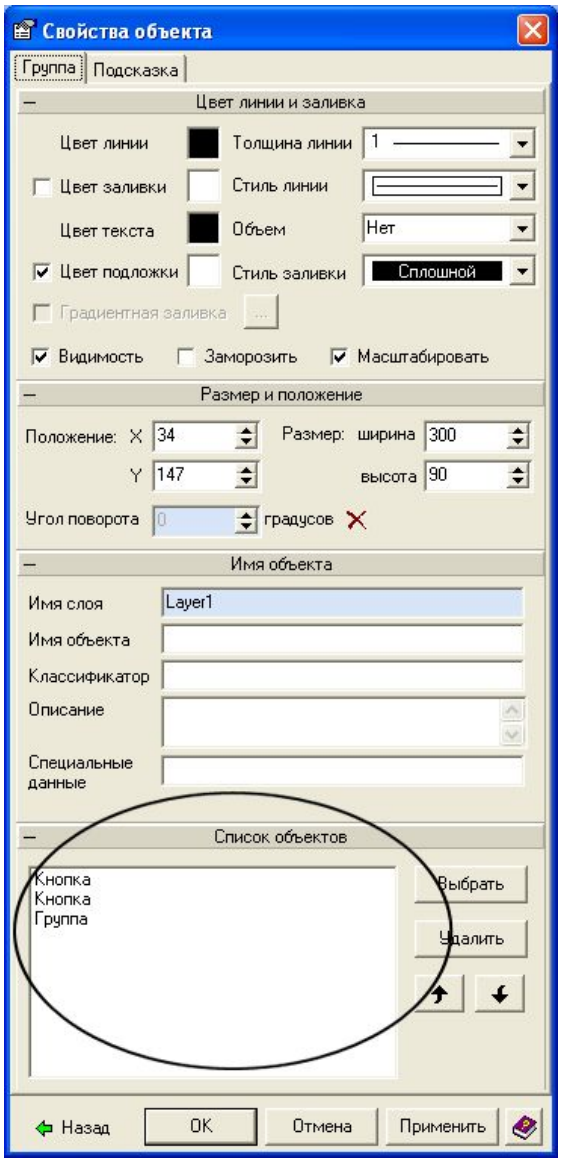


Рис. 3.6. Свойства группы, выбор объекта из группы

7. На **Ошибка! Источник ссылки не найден.** есть семь объектов, которые во избежание смещения относительно друг друга можно (и нужно!) сгруппировать. Доступ к любому объекту в сгруппированном символе осуществляется вызовом на выделенном символе из контекстного меню (правая кнопка мыши) опции **Свойства объекта.** После этого появится окно **Свойства объекта** (**Ошибка! Источник ссылки не найден.**). В данном окне можно выбрать любой объект, который находится в составе данного объекта, и изменить его свойства. После того, как необходимые изменения были произведены нужно нажать на кнопку **Назад** (**Ошибка! Источник ссылки не найден.**).

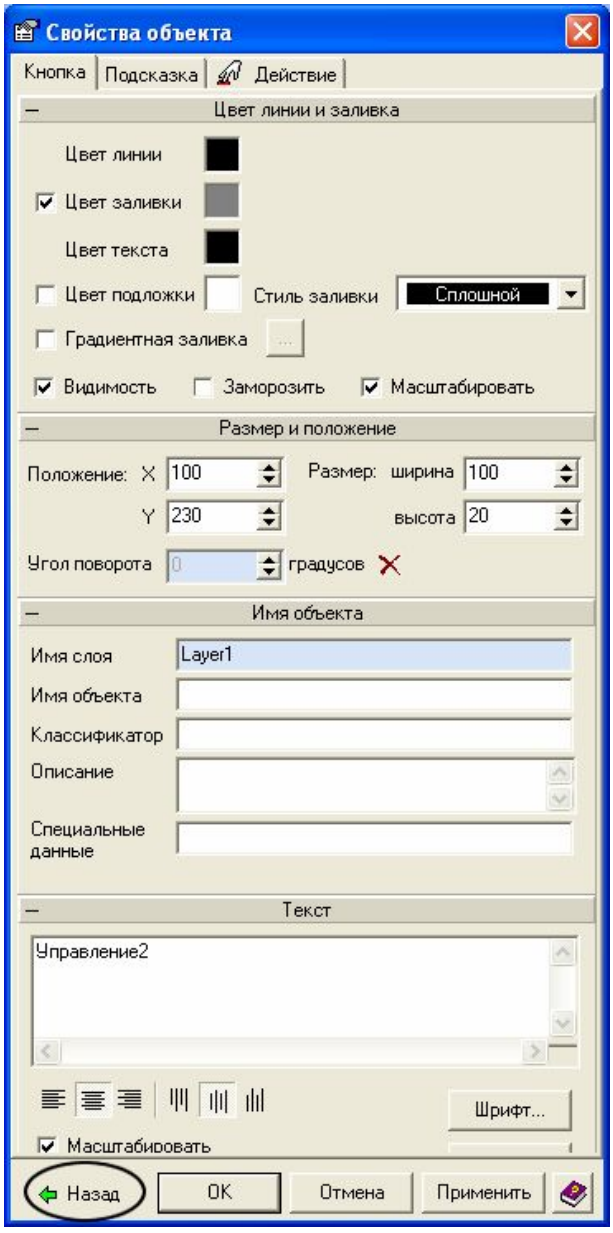


Рис. 3.7. Навигация по объектам группы

8. Представьте, что битовый сигнал **Управление1** будет приходить на верхний по схеме вход логического элемента, а **Управление2** на нижний. Битовый сигнал **Состояние1** – это выходной сигнал (выход элемента). Примените к этим линиям динамику так, чтобы при единичном значении сигналов соответствующие линии становились зелеными, а при нулевых значениях красными. В режиме **Проект – Старт** все линии должны быть черными. Сейчас и далее группируйте объекты в символы, а доступ к объектам в символах осуществляйте опцией **Свойства объекта**.

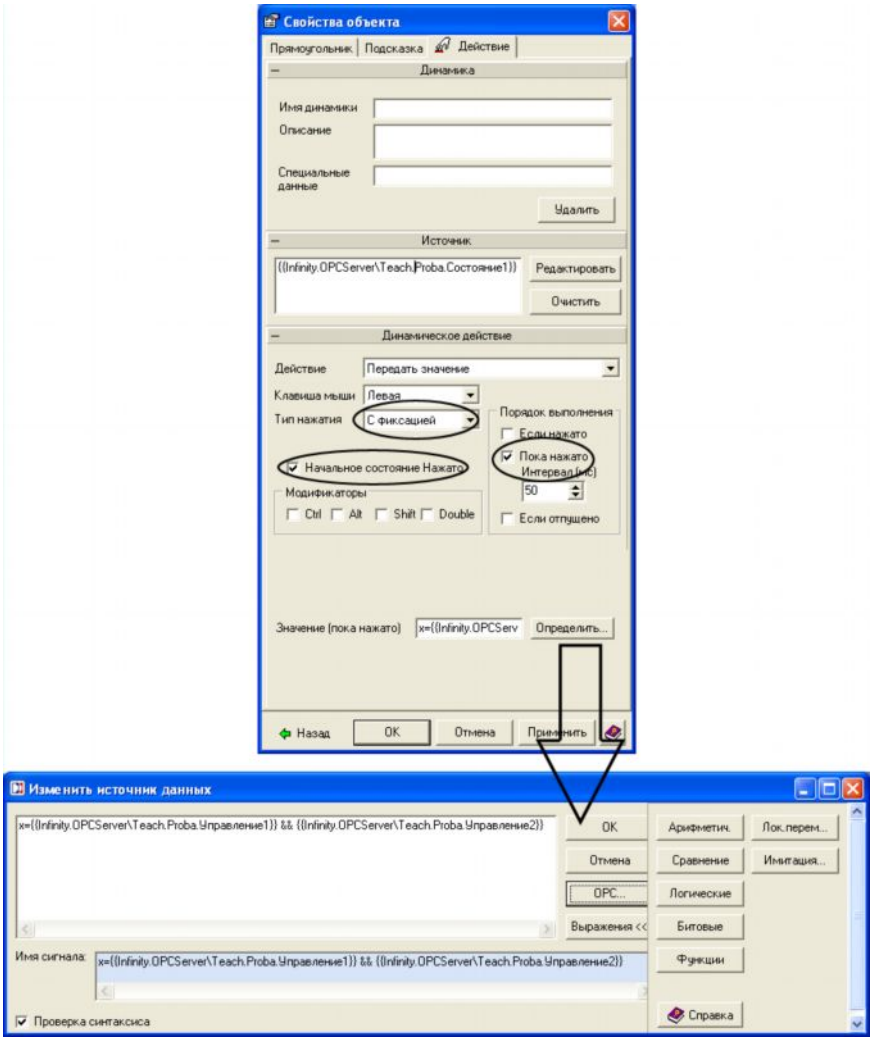


Рис. 3.8. Настройка логической операции

9. Примените к объекту **Прямоугольник** динамику **Динамическое действие**, как показано на **Ошибка! Источник ссылки не найден.**. В поле **Значение (пока нажато)** введите выражение логического умножения для сигналов **Управление1** и **Управление2**. Для ввода выражений служит **Изменение источника данных**, в котором есть кнопки ввода функций и, главное вызов тегов. Настраивая **Свойства объекта,** помни-

те, что Вы определяете условие, по которому выражение поля **Значение** будет записываться в **Источник данных**. **Свойства объекта** на **Ошибка! Источник ссылки не найден.** настроены таким образом, что выражение будет записываться в **Источник данных** один раз в 50 мс. Сразу после перевода мнемосхемы в режим **Проект – Старт**.

10. Закройте **Свойства объекта** и проверьте работу мнемосхемы. Любой ноль на входе даст ноль на выходе.

Еще раз напоминаем: перед выходом из режима Проект – Старт приводите сигналы в исходное (нулевое) состояние.

11. Создайте объекты логических элементов: **И, ИЛИ, И-НЕ, ИЛИ-НЕ**. **Ошибка! Источник ссылки не найден.**

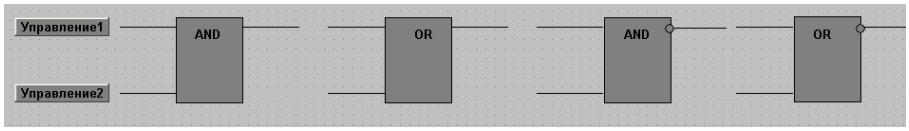


Рис. 3.9 Внешний вид мнемосхемы со всеми логическими элементами

12. Входные сигналы для всех прежние: **Управление1**, **Управление2**, а выходные – **Состояние2, Состояние3, Состояние4**, соответственно. К выходным сигналам примените динамику **Динамический Цвет** по аналогии с выходом созданного объекта **Логический элемент И.** Логические выражения, естественно, должны быть для каждого логического элемента свои. *Необходимо заметить, что логические отрицание в Infinity HMI обозначается как восклицательный знак ! и имеет высший приоритет, там, где это требуется, применяйте простые () скобки.* Очевидно, что это замечание будет актуально во время ввода выражений для функций И-НЕ, ИЛИ-НЕ. Отсутствие скобок перед инверсией выражения для этих **функций =!(классическая&&ошибка).***Также следует учитывать, что при написании формул каждую операцию необходимо заключать в простые скобки.*

13. Расположите объекты в рабочей области как показано на **Ошибка! Источник ссылки не найден.** и сгруппируйте их в один символ.

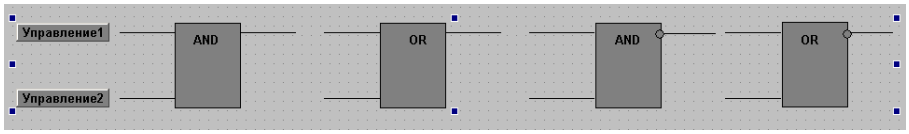


Рис. 3.10. Группировка всех логических элементов

14. Нарисуйте «основание» стенда, цвет заливки: **синий**, объем: **выпуклый**.

15. С помощью инструмента **Текст** A создайте название стенда.

16. С помощью необходимых инструментов создайте объект Таблица, где будут отображаться значение всех сигналов. Общий вид может быть таким, как показан на **Ошибка! Источник ссылки не найден.**.

17. Проверьте экранную форму в режиме **Проект – Старт**.

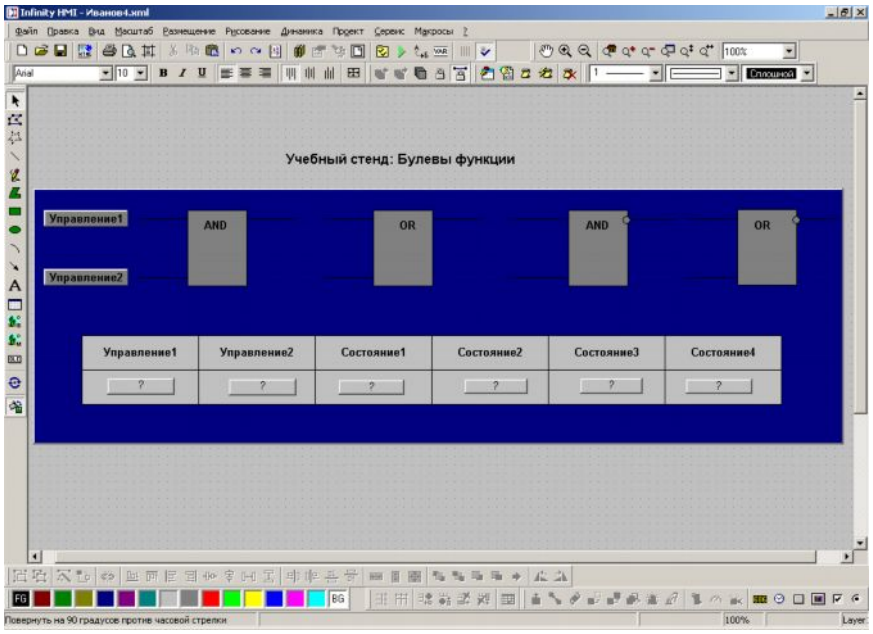


Рис. 3.11. Общий вид мнемосхемы «Учебный стенд: Булевы функции»

18. Сохраните экранную форму в файле **Фамилия4.xml** в личной директории.

Упражнение 2: *Создание графической мнемосхемы «RS – триггер с прямыми входами на элементах ИЛИ-НЕ»*

Создание мнемосхем цифровых элементов в лабораторной работе не преследует цели изучения логики их работы. Задача состоит в динамизации этих объектов, которая делает доступной для визуального восприятия поведение задействованных в мнемосхеме сигналов. Кроме того, поведение сигналов зависит от применения к ним выражений. Можно просто сложить два сигнала и получить функцию ИЛИ, а можно построить, например, мнемосхему триггера – элемента, широко применяемого в системах автоматики. Простой RS – триггер или триггер – защелку можно построить на основе двух элементов ИЛИ-НЕ с взаимными перекрестными связями (см. **Ошибка! Источник ссылки не найден.**). Функция, описывающая работу такого триггера, выглядит так: $Q = \overline{R + \overline{Q}}$, а $\overline{Q} = \overline{S + Q}$.

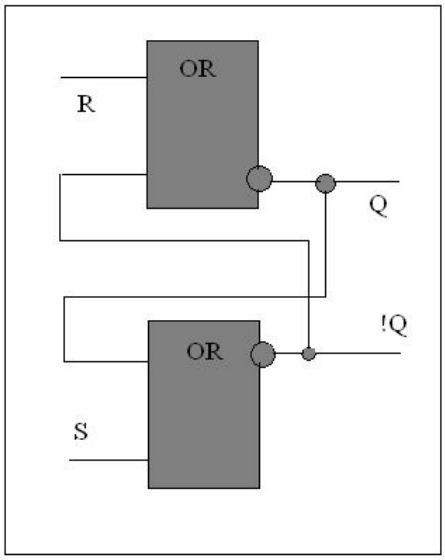


Рис. 3.12. Функциональная схема RS-триггера

В верхнем элементе происходит логическое сложение сигнала R с сигналом Q и инверсия этой суммы, поскольку элемент ИЛИ-НЕ. Работа нижнего элемента идентична, сигналы S и Q, соответственно. Активным входным уровнем является логическая единица.

19. Создайте новый файл **Фамилия5.xml.** Включите сетку и нарисуйте схему, подобную, показанной на **Ошибка! Источник ссылки не найден.**. Для этого создайте прямоугольник, цвет заливки – темно – серый, надпись внутри – OR(ИЛИ). Элемент **Круг** (заливка темно - серая) обозначает функцию логического отрицания, расположите его, как показано на **Ошибка! Источник ссылки не найден.** (верхний элемент).
20. Сгруппируйте три элемента в символ.
21. Сделайте копию символа и разместите оба символа друг над другом, с помощью панели инструментов **Расположение** добейтесь их выравнивания (например, по левому краю).
22. Выделите нижний символ и в вызванном контекстном меню, выберите опцию **Свойства объекта**.
23. Переместите элемент **Круг** вверх (пусть выходы логических элементов будут симметричны – внешний вид мнемосхемы имеет значение!) и выйдите из режима **Свойства объекта**. Нарисуйте перекрестные связи. Для эффективного выполнения этой операции воспользуйтесь описанной ниже последовательностью.
24. Выберите инструмент **Линия** и совместите указатель мыши с левой стороной нижнего символа строго напротив элемента **Круг**.
25. Нажмите левую клавишу мыши и ведите линию на 15-20 мм. влево, следя за тем, чтобы не было переломов линии. После того, как линия нужной длины нарисована, отпустите и снова нажмите левую клавишу

мыши. Ведите линию вверх, повторяйте эти действия пока ломанная не замкнется на элементе **Круг** верхнего символа. Выделите все отрезки ломанной и сгруппируйте в символ.

26. Проделайте тоже самое для второй перекрестной связи.
27. Добавьте элемент **Линия** для входов и выходов.
28. Нарисуйте точки пересечения (серая заливка), как показано на **Ошибка! Источник ссылки не найден.**.
29. Выделите символ первой ломанной, точку ее пересечения с выходом и сам выход, затем сгруппируйте их в один символ.
30. Тоже проделайте для второй ломанной.
31. Выберите объект **Кнопка**, назовите его **Сброс**, и настройте для него переключение значения с 0 на 1 и обратно при помощи динамики **Действие**, В качестве сигнала сброса будет использован сигнал **Управление1**.
32. Расположите кнопку **Сброс** возле соответствующего входа верхнего элемента ИЛИ-НЕ.
33. Примените к линии, обозначающей этот вход, динамику **Цвет** так, чтобы при значении сигнала **Управление1** – единица, линия становилась зеленой, иначе красной.
34. Создайте второй объект **Кнопка**, название **Запись**, сигнал **Управление2**. Настройка вкладки динамики аналогична. Расположите кнопку **Запись** возле соответствующего входа нижнего элемента.
35. Примените к линии, обозначающей этот вход, динамику **Цвет** так, чтобы при значении сигнала **Управление2** – единица, линия становилась зеленой, иначе – красной.
36. Выделите сгруппированный символ, куда входит выход **Q**.
37. Примените к этой группе динамику **Цвет**: выходным сигналом **Q** будет сигнал **Состояние1**. Настройка динамики показана на **Ошибка! Источник ссылки не найден.**3. Цветовая политика та же: красный цвет – нулевое значение сигнала, зеленый – единичное. В режим **Проект – Старт** ломаная линия, выход **Q** и контур точки их соединения будут расцвечиваться в соответствии с выбранной цветовой политикой. В режиме **Разработка** все элементы должны быть нейтрального цвета (мы выбрали серый цвет для объектов и черный для линий).

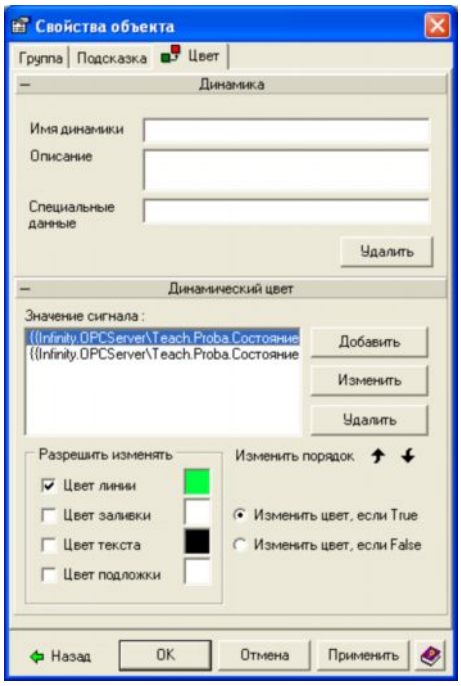


Рис. 3.13. Настройка переключения цвета для линии

38. Примените динамику **Цвет** к другому сгруппированному символу, куда входит инверсный выход **!Q**. В качестве сигнала используйте сигнал **Состояние2**.

39. Сохраните файл и проверьте работу кнопок и динамики цвет в режиме **Проект – Старт**. Если кнопки переключают сигналы **Управление1, Управление2** и корректно отображаются цвета линий соответствующих входов триггера, то можно приступать к реализации собственно триггера.

Применительно к сигналам, которые используются в мнемосхеме, логика работы триггера на элементах ИЛИ-НЕ будет выглядеть так:

Состояние1=!(Управление1||Состояние2)

Состояние2=!(Управление2||Состояние1).

Сигнал Управление1 (Сброс) складывается со значением сигнала Состояние2 (инверсный выход триггера), далее эта сумма инвертируется и записывается в сигнал Состояние2 (инверсный выход триггера). Поскольку путь к сигналу задается тегом, то логика примет вид:

40. Примените к верхнему по схеме символу динамику **Динамическое действие**. Настройте **Свойства объекта**, как показано на **Ошибка! Источник ссылки не найден.**4

41. Примените такую же динамику, но с соответствующими сигналами, к нижнему по схеме логическому элементу.

42. Проверьте работу мнемосхемы в режиме **Проект – Старт**. Как и реальный триггер, мнемосхема может некорректно работать при одновременной подаче на входы записи и сброса активных уровней, это будет наблюдаться в первый момент времени после перехода в режим **Проект – Старт**. Такой же эффект будет наблюдаться при слишком кратковременном нажатии на кнопку при переводе триггера в противоположное состояние. Нажмите любую из кнопок, и триггер займет предписанное логикой состояние. Если триггер защелкивает выходы, после снятия управляющих сигналов, значит, динамика настроена, верно. Вид экранной формы триггера на элементах ИЛИ-НЕ показан на **Ошибка! Источник ссылки не найден.**5.

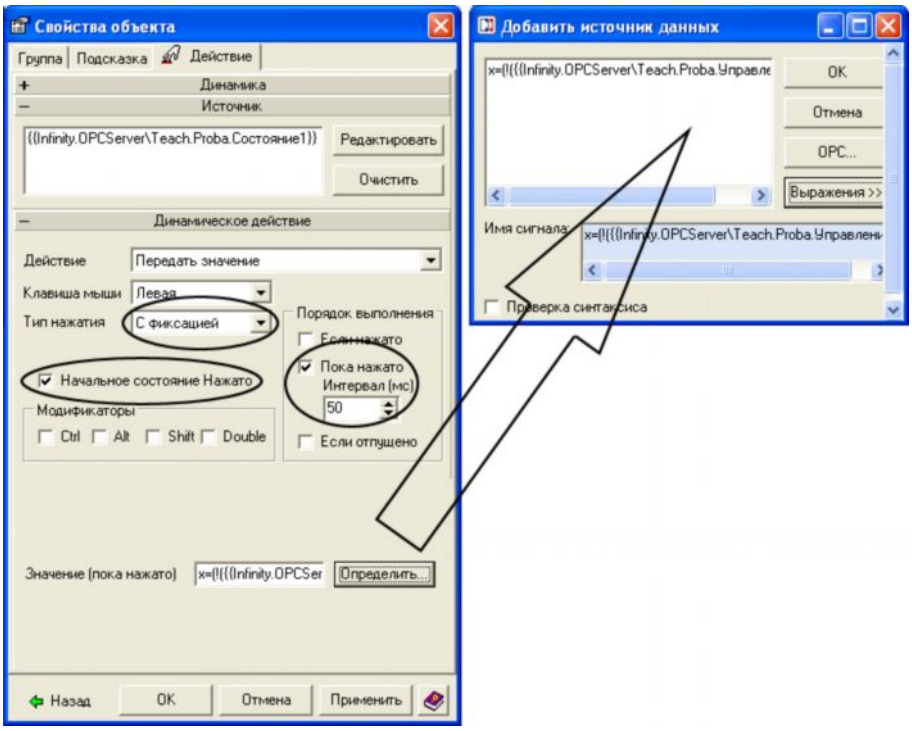


Рис. 14. Настройка логики для верхнего элемента триггера

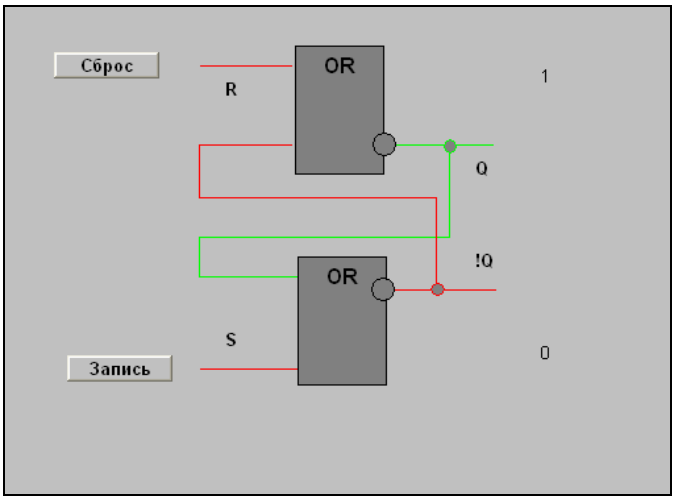


Рис. 3.15. Схема RS-триггера в действии

43. Сохраните файл под именем **Фамилия5.xml**.

Упражнение 3: *Создание графической мнемосхемы «функциональный блок Т».*

Функциональный блок T представляет собой тот же триггер – защелку, но с логикой, основанной на функции логического И-НЕ. В предыдущем упражнении было необходимо детализировать объект, вплоть до динамизации перекрестных связей, чтобы работа триггера была наглядна. Теперь возникает необходимость в создании одного функционального блока, как это принято, например, на электрических принципиальных схемах. Все это, конечно, не в ущерб возложенных на мнемосхему функций. Триггер выглядит так, как показан на **Ошибка! Источник ссылки не найден.**6. Логика работы $Q = \overline{\overline{S} \& \overline{Q}} \& \overline{R}$. Из формулы видно, что кратковременная подача на вход R единицы сбросит единственный выход Q в ноль, после чего кратковременной подачей единицы на вход S можно снова перевести выход Q в активное состояние.

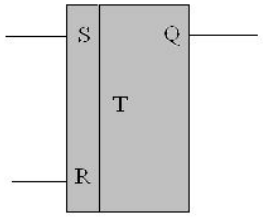


Рис. 3.16. Схема RS-триггера в виде одного блока

44. Создайте файл с именем **Фамилия6.xml**.Нарисуйте мнемосхему триггера с темно – серым цветом заливки. Внешний вид показан на **Ошибка! Источник ссылки не найден.**3.17. Сигналы: **Управление2** – Запись (S), **Управление1** – Сброс (R), **Состояние1** – Выход (Q).

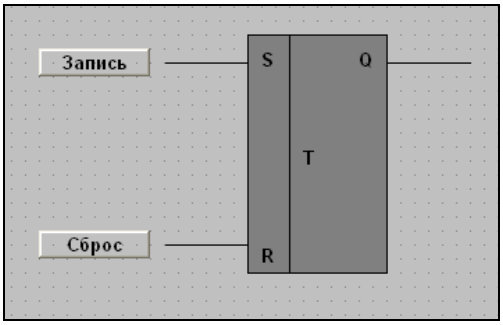


Рис. 3.17. Схема RS-триггера на мнемосхеме

45. Скопируйте кнопки **«Сброс»** и **«Запись»** из предыдущего файла и разместите их возле входов триггера соответственно. Примените к входам и выходу триггера динамику **Цвет** по аналогии с предыдущей мнемосхемой.

46. Для вычисления выражения примените к объекту (к любому, но по смыслу разумно это сделать на символе изображения триггера) динамику **Динамическое действие** и настройте ее в соответствии с упоминавшейся формулой, как показано на **Ошибка! Источник ссылки не найден.**3.18.

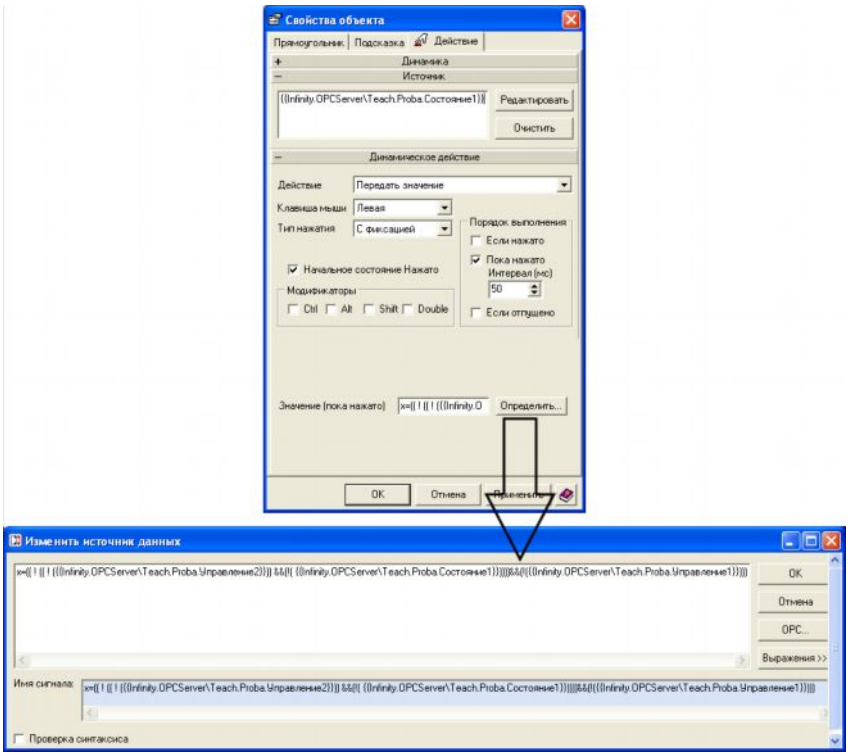


Рис. 3.18. Настройка динамики для RS-триггера

47. Проверьте мнемосхему в режиме **Проект – Старт**, если она работает корректно, сохраните мнемосхему в файле **Фамилия6.xml**. На этом лабораторная работа закончена

Контрольные вопросы

1. Определите число внутренних каналов управления на скриншоте, показанном на рис.3.5.
2. Приведите примеры практического применения внутренних SCADA-каналов для управления технологическими объектами.
3. Нарисовать скриншот «Логика триггера на элементе ИЛИ-НЕ»

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе

Лабораторная работа №4. Работа с пакетом INFINITY HMI. Библиотека объектов

Цели работы:

1) закрепление навыков работы с интерфейсом Infinity HMI;
2) работа с библиотекой символов;
3) применение инструмента **Слои;**
4) применение функций и условий для оптимизации работы мнемосхемы;
5) создание новых объектов.

Теоретическая часть

Инструменты **Слои** широко применяются во всех SCADA пакетах. Благодаря этим инструментам осуществляется «оживление» экранных форм. Это необходимо для привлечения внимания диспетчера к текущим событиям в технологическом процессе. Слои в экранных формах Infinity HMI являются удобным средством для объединения наборов графических объектов, когда объекты, входящие в данный набор должны отображаться при в определенных условиях.

В отсутствии этих условий объекты должны быть скрыты. Это очень разумно с точки зрения читаемости мнемосхемы в основное время, когда эти определенные условия не наступили. Экранная форма, таким образом, не перегружена лишними объектами, которые в обычное время мешают наблюдать за технологическим процессом. Условия могут быть разными: сигналы аварии, любые другие сигналы, изменение масштаба экранной формы. Каждая экранная форма изначально содержит один слой, называемый Первичным или Системным.

Программа выполнения

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1. *Создание модели технологического процесса*

1. Загрузите в Конфигуратор базу сигналов Teach из личной директории.
2. Запустите **Infinity HMI**.
3. Откройте свой файл **Фамилия7.xml** из личной директории и сохраните его под именем **Фамилия8.xml**
4. Проверьте работоспособность мнемосхемы.
5. Измените размеры символа так, чтобы на рабочем поле экрана справа от символа можно было создать новые объекты.
6. В панели инструментов **Рисование** найдите кнопку
7. **Показать библиотеку символов**  и в появившемся окне выберите

Файл→ открыть. Укажите путь к библиотеке Методические указания\Library_symbols.xml, Рис.

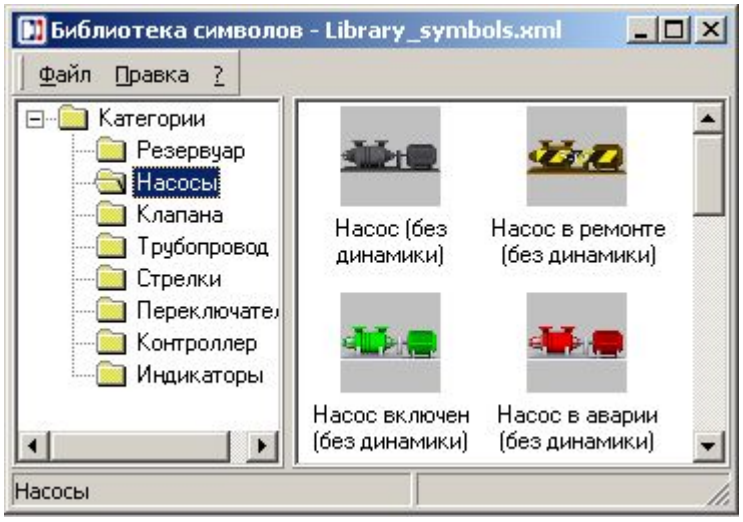


Рис. 4.1. Библиотека символов

Библиотека символов является отдельным независимым от Infinity HMI приложением. **Библиотека символов** предназначена для хранения графических символов с целью дальнейшего их использования. Интерфейс пользователя **Библиотеки** аналогичен применяемому в Проводнике Windows. Главное окно **Библиотеки символов** разделено на две области. Левая область содержит древовидный список, в котором находятся каталоги и категории символов. В правой части окна расположены изображения символов текущей выбранной категории. Любой графический объект, находящийся в экранной форме, может быть сохранён в текущей выбранной категории путём выполнения простой операции переноса. Загрузку символов из библиотеки в экранные формы выполняется аналогично, путём переноса.

8. Перенесите на свободное место экранной формы объекты, необходимые для визуализации процесса, показанного на Рис. . **Переносите только объекты с пометкой (с динамикой).** Закройте библиотеку символов.

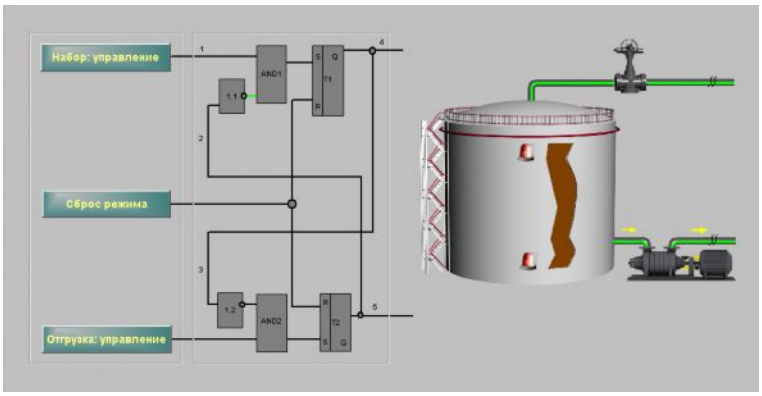


Рис. 4.2. Мнемосхема с добавленными объектами из библиотеки

9. Продолжите линии для соединения выхода **4** с задвижкой, а выхода **5** с насосом. Обязательное условие: *не применять разгруппировку символа и сохранять динамику, примененную к линиям***.** Не имеет значения, будут эти линии прямыми или ломанными, важно показать связь.
10. Запустите **Конфигуратор** и создайте в папке **Pump** и **Valve** по одному сигналу с именем **Status** (состояние) и свойствами, как у сигналов **Control** и **In**.

 Сигнал **Control** (управление) – это дискретный сигнал относится к классу сигналов телеуправления (ТУ). Это команда контроллеру включить (выключить) исполнительный механизм (задвижку или насос). В свою очередь исполнительный механизм должен подтвердить контроллеру (а контроллер в OPC-сервер и дальше в мнемосхему диспетчеру) изменение своего состояния выдачей дискретного сигнала, который относится к классу сигналов телесигнализации (ТС). В этой лабораторной работе будет использована программная имитация этого подтверждения путем перекладки в динамике сигнала управления ТУ (**Control**) в сигнал состояния ТС (**Status**) задвижки и насоса. Между приходом ТУ на механизм и выдачей ТС есть задержка, обусловленная особенностями самого механизма (например, срабатывание конечного выключателя задвижки из положения "закрыто" в положение "открыто" после получения сигнала на открытие). Эта задержка будет реализована так же программно.

11. Примените к изображению задвижки динамику **Динамическое действие** и настройте её таким образом, чтобы один раз в **2500** мс в сигнал **Status** директории **Valve** записывалось значение сигнала **Control.** Это и будет программная перекладка сигнала ТУ в сигнал ТС с необходимой задержкой на 2,5 секунды (точнее сказать, задержка будет от 0 до 2,5 скунд).
12. Примените к изображению задвижки динамику для визуализации значения её сигнала ТС. Для этого кликните правой кнопкой мыши на изображении задвижки и выберите пункт **Редактировать псевдонимы**. В появившемся окне (Рис.) замените в каждой строке, имя переменной **<<Valve>>** на путь к сигналу **Status** директории **Valve.**

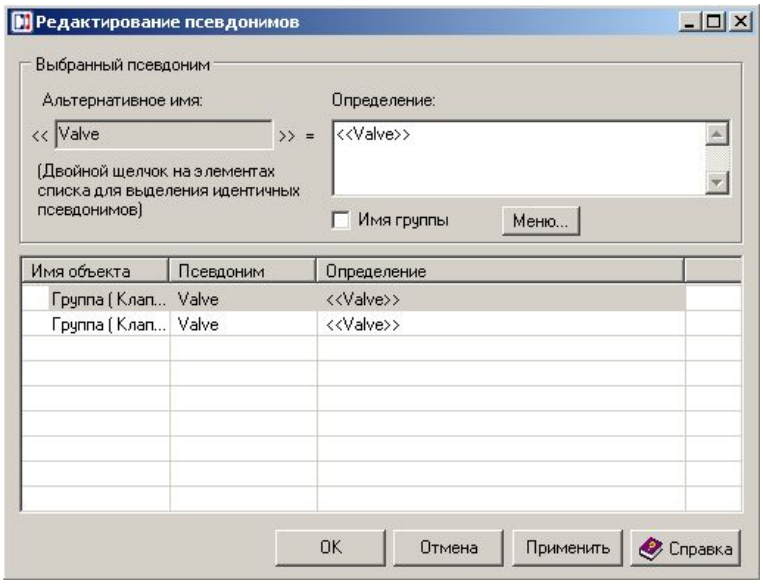


Рис. 4.3. Редактирование псевдонима <<Valve>>

13. Выделите изображение насоса и примените к нему соответствующую динамику для перекладки его сигналов ТУ в ТС с такой же задержкой.
14. Аналогичным образом настройте визуализацию значения сигнала ТС для насоса.
15. Примените настройки, сохраните файл и проверьте работу мнемосхемы. Через время, определенное задержкой в 2,5 секунды, после записи в сигнал ТУ задвижки единицы, её цвет должен измениться на зелёный, это будет говорить о единичном уровне ТС. Та же картина будет наблюдаться и с насосом. Выключение исполнительных механизмов будет происходить с такой же задержкой относительно сигналов ТУ.
16. Следующим шагом станет создание изменяющегося уровня в резервуаре.
17. Настройте псевдонимы в резервуаре, для визуализации уровня жидкости. (см. шаг 12).
18. Примените к изображению резервуара динамику для реализации следующей логики: уровень в резервуаре равен поступлению через задвижку (ТС) минус отгрузка через насос (ТС). Время записи вычисленного значения выражения в сигнал **Level** равно **50**мс.
19. Выберите объект **Значение параметра** для визуализации значения сигнала **Level.** Экранная форма в режиме **Проект- Старт** должна выглядеть, как на Рис. .4.
20. Сохраните файл под именем **Фамилия8.xml**.

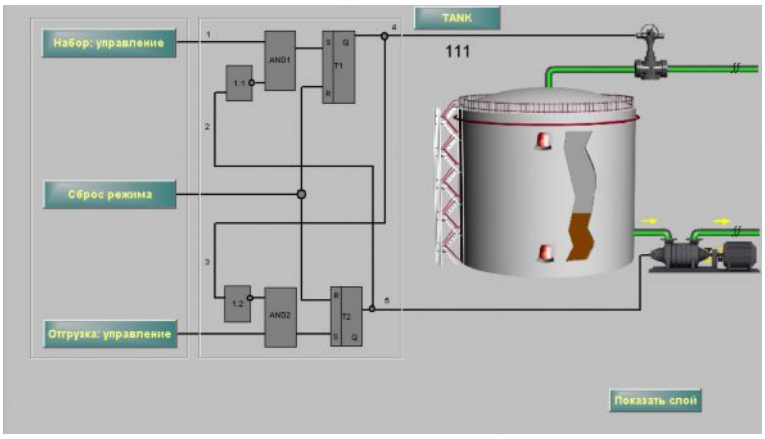


Рис. 4.4. Мнемосхема управления наливом/откачкой

Упражнение 2. *Применение слоёв*

Мнемосхема, содержащаяся в файле Фамилия8.xml находится в первичном слое и содержит объекты, которые оператор, предположим должен видеть всегда. По умолчанию редактор слоёв предлагает Имя слоя: Layer2

21. Увеличьте объект Значение параметра, в котором происходит индикация значения сигнал Tank и создайте над ним надпись.
22. Нажмите на кнопку и создайте новый слой. Название по умолчанию **Layer2.**
23. Добавьте кнопку **Показать слой** и настройте **Свойства объекта,** как показано на Рис. .

Представьте, что существует некая потребность иногда выводить значения всех сигналов, с которыми связана мнемосхема, не только цветной политикой, но и в цифровом виде. Реализация этого требования в Первичном слое приведёт к перегруженности экранной формы. Это удобно сделать в слое **Layer2**, который не будет виден в основном масштабе экранной формы.

24. Создайте в слое **Layer2** объекты **Значение параметра**, разместите их и привяжите к ним соответствующие сигналы и локальные переменные (Рис.).

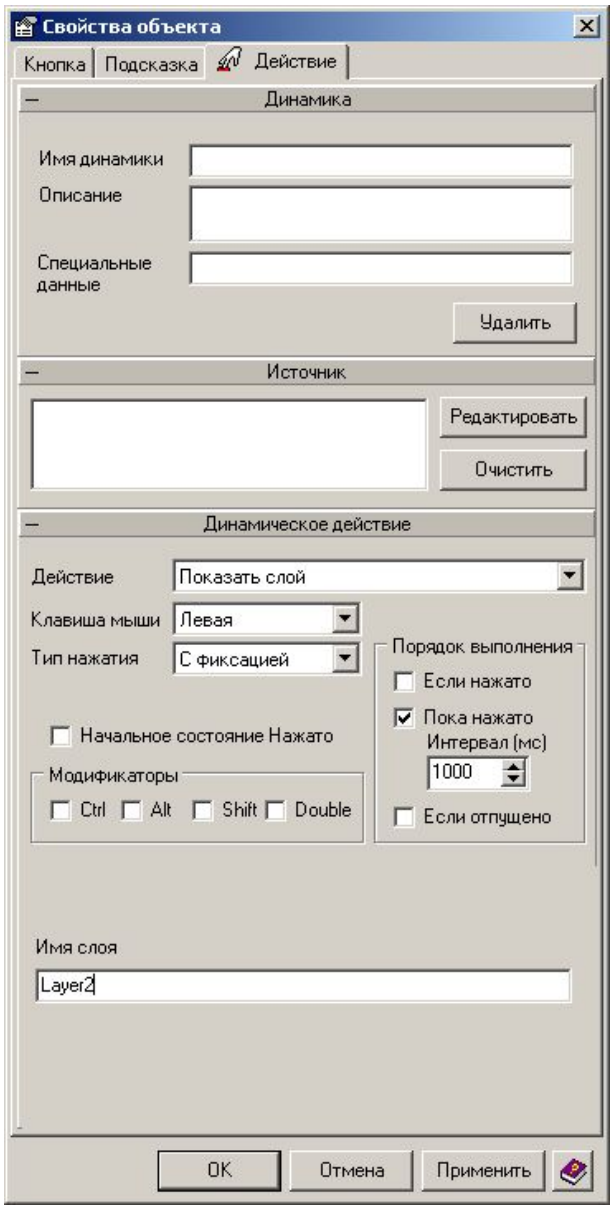


Рис. 4.5. Свойства кнопки «Показать слой»

25. Перейдите в режим **Исполнение,** слой **Layer2** должен быть виден при нажатии кнопки **Показать слой**.

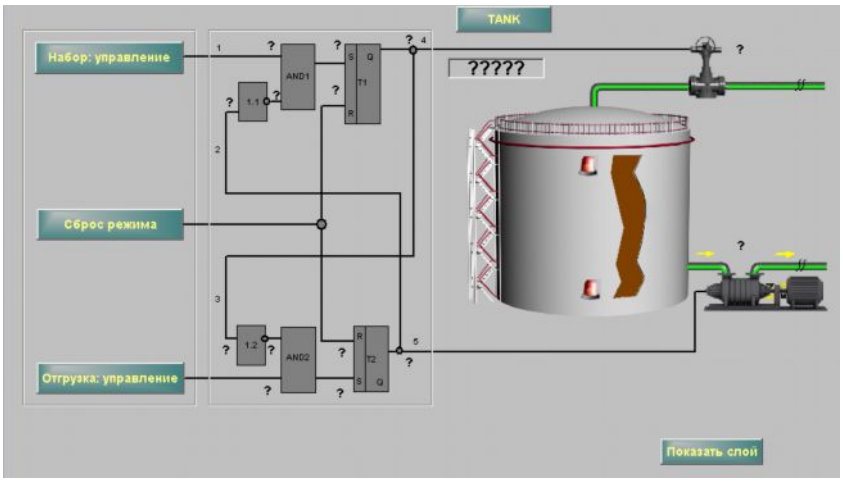


Рис. 4.6. Размещение объектов «Значение параметра» на втором слое мнемосхемы

Упражнение 3. *Оптимизация мнемосхемы технологического процесса с помощью функций и условий*

26. Обратите внимание, что уровень в резервуаре при включенной задвижке может далеко превысить 1000 единиц, которые определяют 100% наполнения резервуара. Необходимо модифицировать алгоритм: если сигнал **Level** больше, чем 500, то ему присваивается значение 1000, если меньше, то сигналу **Level** присваивается текущее значение, определяемое алгоритмом работы процесса. Откройте **Инспектор свойств**, в котором находится формула этого алгоритма, и с помощью функции **if** задайте подобное условие. Пример if (T>1000, 1000, T+З-Н), если T>1000, то T присваивается 1000, если T<1000, то в T записывается вычисленное значение T+З-Н. Для проверки работы условия увеличьте скорость набора и отгрузки в 10 раз.
27. Перенесите из библиотеки символов две лампочки.
28. Расположите их друг над другом на резервуаре рядом с разрезом. Верхняя будет сигнализатором переполнения, нижняя – индикатором предельно допустимого нижнего уровня (см. Рис.).

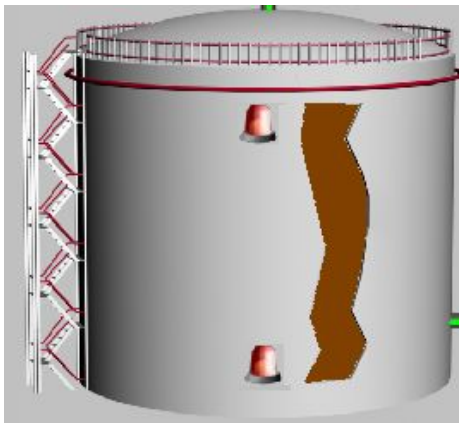


Рис. 4.7. Лампочки-сигнализаторы выхода за максимальный и минимальный уровни

29. Что бы настроить у элемента **Лампочка** динамику **Мигание**, необходимо вызвать Редактирование псевдонимов и ввести условие. Порог 800 единиц. (Рис.)

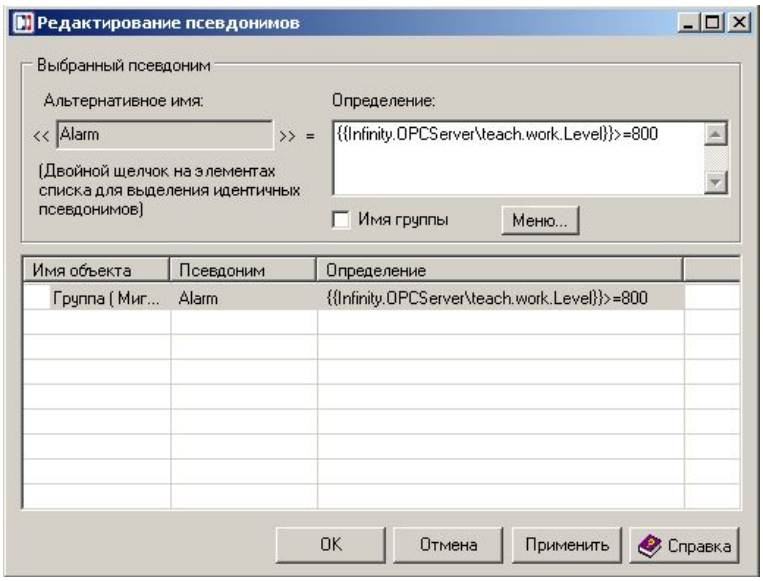


Рис. 4.8. Значения псевдонимов для лампочек

30. Примените такую же динамику для нижней лампочки, порог для нижнего индикатора уровня – 200 единиц.
31. Проверьте мнемосхему в режиме **Исполнение.** При достижении указанных уровней цвет лампочек будет меняться на красный один раз в 250мс. Значение уровня не должно превышать 1000 единиц.
32. Упражнение 4. Замена локальных переменных на выражения

 Как правило, локальные переменные используются в качестве источников данных для различных анимационных эффектов. Как уже отмечалось, локальные переменные действуют только в пределах одной экранной формы. Также причиной их применения в упражнениях №№ 2 и 3 стало стремление упростить процесс создания мнемосхемы, сделать его ступенчатым. Теперь необходимо модифицировать две формулы для элементов Т1 и Т2, раскрыв локальные переменные. Это упражнение будет проверкой на внимательность и понимание логики, обрабатываемой в мнемосхеме.

33. Откройте **Свойства объекта** с вкладкой **Динамическое действие**, которая выполняет функцию триггера. В приведенных примерах это символ триггера Т1 или Т2 (любой из них). В **Редакторе выражений** откройте эту формулу. Ниже приведена формула для Т1.

x = !(!~~AND1~~&&!
{{Infinity.OPCServer.Teach.Work.Valve.Control}}) &&
!{{Infinity.OPCServer.Work.Reset}}

34. Ваша задача в замене локальных переменных **~~AND1~~** и **~~AND2~~** на выражения, которые вычисляются динамикой, примененной к символам элементов AND1, AND2 и инверторов 1.1, 1.2. Закройте **Редактор выражений** и **Свойства объекта**.
35. Откройте Редактор выражений для формулы, записываемой в ~~invert_Pump.Control~~:!{{ Infinity.OPCServer.Teach.Work.Pump.Control}}
36. Скопируйте это выражение и подставьте вместо соответствующей локальной переменной в выражение для AND1:

{{ Infinity.OPCServer.Teach.Work.Valve.In}} &&
!{{Infinity.OPCServer.Work.Pump.Control}}.

37. Скопируйте это выражение и подставьте вместо выражения для Т1:

!(!({{ Infinity.OPCServer.Teach.Work.Valve.In}}&&
!{{Infinity.OPCServer.Teach.Work.Pump.Control}})&&
!{{Infinity.OPCServer.Teach.Work.Valve.Control}})&&
!{{Infinity.OPCServer.Teach.Work.Reset}}.

38. Выражение, выделенное крупным шрифтом, и есть замена ~~AND1~~. Обратите внимание на то, что выражение дополнительно заключено в скобки для общей инверсии, как и было для ~~AND1~~.
39. Проделайте действия пунктов 2-5 для Т2.
40. Замените локальные переменные в **Источнике данных** для четырех линий (динамика **Цвет**) на выражения инверсии и логического умножения, соответственно. То же проделайте для соответствующих 4 объектов в слое **Parametrs**.
41. Удалите динамику **Динамическое действие** на элементах AND1, AND2 и инверторах 1.1, 1.2. Теперь все выражения логики работы триггеров будут вычисляться в двух формулах.
42. Сохраните файл под именем **Фамилия9.xml**

43. Проверьте работу мнемосхемы. Внешне все должно выглядеть так же, как в мнемосхеме файла **Фамилия8.xml**, но при наведении указателя мыши на элементы AND1, AND2 и инверторы 1.1, 1.2 не будет отображаться применение к ним динамики **Динамическое действие**. На этом лабораторная работа закончена.

На этом лабораторная работа закончена

Контрольные вопросы

1. Для чего используются инструменты СЛОИ?
2. Могут ли локальные переменные действовать в пределе нескольких экранных форм? Объясните ответ
3. Как отображаются локальные переменные?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе.

Лабораторная работа №5. Организация логики изменения содержания экранной формы при помощи VBA

Цели работы:

1) изучение возможностей по созданию скрипта логики изменения содержания экранной формы на языке VBA;
2) организация доступа к значению сигналов из VBA и взаимодействия мнемосхемы (скриншота) с другими программами при помощи VBA.

В процессе достижения данной цели необходимо решить две задачи:

1. Обеспечить логику обработку сигналов на VBA;
2. Осуществить экспорт данных в программу Excel.

Теоретическая часть

Одной из важных и замечательных особенностей программы I**nfinityHMI** является возможность предоставлять свои объекты для использования посредством технологии **OLE Automation**. Данная возможность позволяет с одной стороны, создавать внешние приложения, способные работать с объектами **InfinityHMI**, а с другой стороны создавать логику обработки данных внутри мнемосхем при помощи технологии **Microsoft Visual Basic for Application (VBA).**

В даной работе будут использоваться основные классы, предоставляемые **InfinityHMI** и некоторые методы, которые могут быть полезны при разработке **VBA** скриптов.

Класс **`GwxDisplay`** представляет объекты экранных форм **InfinityHMI**. Экранные формы **InfinityHMI** содержат множество объектов классов, производных от **`GwxVisible`**, **`GwxDynamic`** и **`GwxPoint`**. Объект **`GwxDisplay`** предоставляет методы для создания новых и получения существующих экземпляров объектов трех перечисленных классов.

`GwxVisible` является базовым типом объекта (классом) для всех объектов экранных форм **InfinityHMI**, которые могут быть отображены в ее рабочей области (могут быть визуально восприняты пользователем). Все видимые объекты **InfinityHMI** (например, **`GwxRectangle`**, **`GwxEllipse`**, **`GwxText`** и т.д.) наследуются от **`GwxVisible`**, а значит, приобретают все его свойства и методы. Часто используемый класс **`GwxButton`** является наследованным от класса **`GwxText`**. Также в этой иерархии находятся классы для отображения растровых (bmp) и векторных (**Metafile**) рисунков и **OLE** компонентов и т.д. Особого внимания из данной серии классов заслуживает класс **`GwxSymbol`**, инкапсулирующий последовательность других визуальных объектов. Объекты данного типа используются для обеспечения группировки визуальных объектов.

`GwxDynamic` является базовым классом объектов экранной формы **GraphWorX32**, которые выполняют динамическое изменение визуальных атрибутов связанных с ними видимых объектов. Такие, например, как **`GwxSize`**, **`GwxLocation`**, **`GwxRotation`**, **`GwxPick`** и т.д. Например, объ-

ект класса **GwxSize**, связанный с объектом класса **GwxRectangle**, может изменять размеры последнего в зависимости от значения переменной в **OPC**-сервере. Динамические объекты не могут восприниматься визуально. Они служат для выполнения динамических действий над свойствами связанных с ними видимых объектов. Объект класса **GwxVisible** может иметь множество связанных с ним объектов класса **GwxDynamic**, тогда как один объект класса **GwxDynamic** может быть связан с одним и только одним объектом класса **GwxVisible**. То есть, когда при разработке мнемосхемы Вы выбираете пункт меню «Добавить динамику», то создается новый объект, наследованный от **GwxDynamic** и он связывается с визуальным объектом, к которому данная динамика применяется.

GwxPoint является базовым классом объектов в экранных формах **GraphWorX32**, которые связаны с элементами данных в серверах **OPC**, значениями выражений, локальными переменными или константами. С каждым динамическим объектом класса **GwxVisible** могут быть связаны один или более объектов класса **GwxPoint**. Объекты класса **GwxPoint** создаются и уничтожаются объектами класса **GwxDynamic.**

Объекты типа **GwxDisplay** генерируют ряд событий, интересных для нас. Это, например, такие события, как **DisplayLoad**, которое возникает после загрузки экранной формы; **PickPostDown**, которое возникает после совершения щелчка, на объект, обладающий динамикой типа «Динамическое действие» и т.д. Список всех событий можно посмотреть в редакторе **VBA**, для данного дисплея (**ThisDisplay**). Однако, вместо обработки этих событий, можно просто указать имя процедуры в **VBA** скрипте, которая должна быть вызвана, если наступит определенная ситуация (нажатие кнопки и т.д.).

Программа работы

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения

Упражнение 1. *Организация логики на VBA*

1. Возьмите за основу файл **Фамилия9.xml** и сохраните его под именем Фамилия10.xml. Модифицируете мнемосхему следующим образом: вместо блока логики управления поставим некоторый черный ящик, как показано на Рис. . Настройте источники сигналы (при помощи поиска и замены) так, чтобы они брали сигналы не из сервера ввода-вывода, а из DualSource.

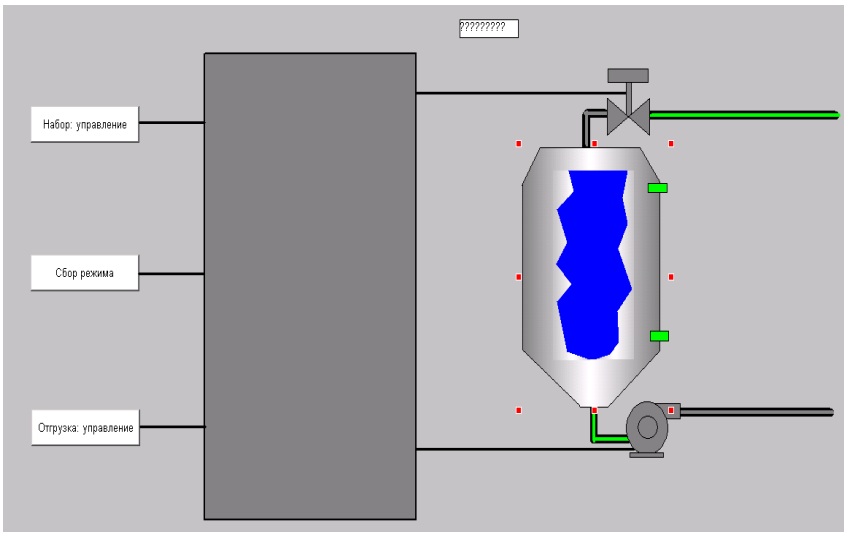


Рис. 5.1. Внешний вид мнемосхемы с черным ящиком скрипта

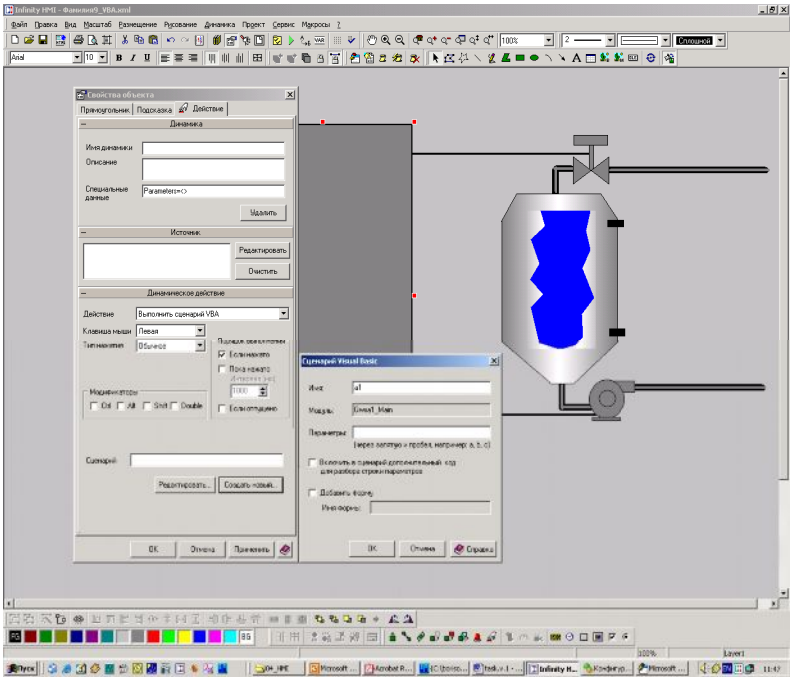


Рис. 5.2. Настройка вызова скрипта по щелчку на черный ящик

2. Для упрощения **отладки** VBA скрипта сделайте следующее: Все кнопки сделайте «С фиксацией», а на черном ящике добавьте динамику «Динамическое действие», как показано на рис. 5.2. Здесь a1 – это название VBA скрипта (можно использовать любое имя, соответствующее спецификации символьных имен: процедур, переменных и т.д.). Теперь VBA скрипт будет вызываться только по щелчку на черный ящик. Причем, начальное состояние входных сигналов можно будет задать при помощи кнопок до вызова скрипта. InfinityHMI может добавить код для разбора специальных параметров и/или форму, но в нашей работе этого не требуется.

3. По кнопке «Редактировать» в инспекторе свойств черного ящика будет открыта среда для разработки VBA скриптов. В модуль Gwxa1_Main (он у Вас должен открыться по умолчанию) запишем программу – как на Рис. . Некоторые комментарии к этому исходному тексту приведены ниже.
4. Для получения значения из OPC и записи значения в OPC мы описали две функции: GetOPCValue и SetOPCValue. Эти две функции сводятся к обращению к соответствующим функциям из динамической библиотеки DualSource. Есть и другие способы получения доступа к OPC, но в нашем случае проще всего обращаться через DualSource. Переданный в качестве параметра объект GwxPick здесь не используется вообще.
5. Вся логика реализована при помощи операторов **`if`**, это не самый лучший вариант, но вполне приемлемый. Преимущество VBA перед другими способами задания логики в том, что здесь можно создавать алгоритмы любой сложности.
6. При реализации алгоритма для простоты мы вводим 2 глобальные булевы переменные для хранения предыдущего состояния насоса и помпы. Существует некоторая сложность, связанная с записью OPC сигнал. Значение в этом сигнале меняется не мгновенно, а с некоторой задержкой, и, при повторном считывании значения сигнала оно не всегда успевает обновиться. Именно по этой причине мы воспользовались глобальными переменными для хранения предыдущих значений выходных сигналов.
7. Убедитесь, что на насосе и задвижке есть перекладка значения из сигнала **`control`** в соответствующий сигнал **`status`**, реализовано мигание для сигнальных лампочек.

 Стоит сохранять проект перед входом в режим исполнения!

8. Теперь можно пробовать запускать эту мнемосхему и тестировать полученную VBA программу. Для этого переведите проект в режим исполнения, наберите на кнопках управления нужный режим, например, «Набор: управление» и щелкните по черному ящику. Если все реализовано нормально, то при этом линия, идущая на задвижку должна загореться зеленым, через некоторое время задвижка должна стать зеленой и уровень начать прибывать. Теперь можно выключить кнопку «Набор: управление» и снова щелкнуть по черному ящику, но набор при этом должен продолжиться. Отладьте подобным образом все возможные комбинации управляющих команд.

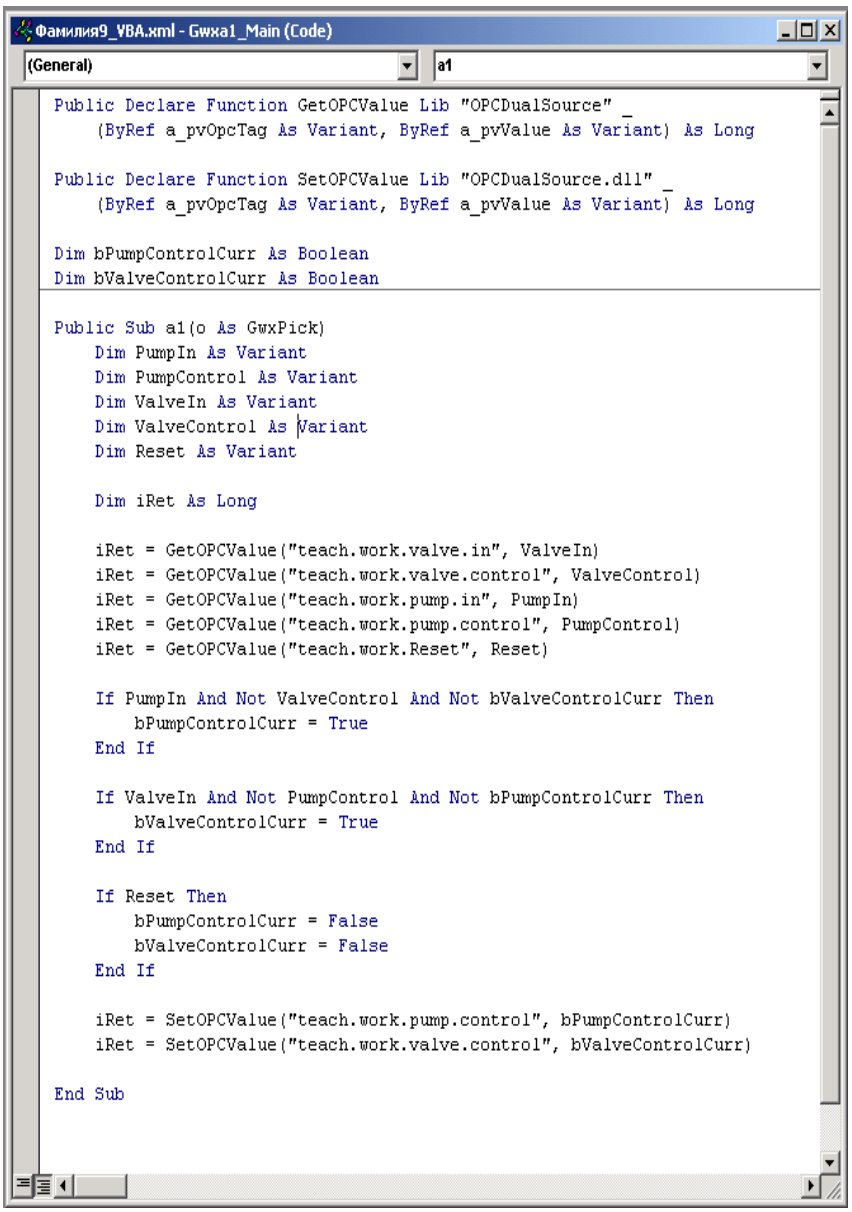


Рис. 5.3. Исходный текст VBA для организации логики управления

Еще раз напоминаем: перед выходом из режима исполнения приводите все сигналы в исходное (нулевое) состояние.

9. После того, как закончите отладку VBA скрипта, можно сделать, чтобы он работал непрерывно. Для этого верните назад настройки кнопок (тип нажатия - обычное, передача значения, если нажато – 1, если отпущено - 0) и установите динамику щелчка черного ящика, так, чтобы скрипт вызывался постоянно с интервалом 50мс (Тип нажатия – с фиксацией, выполнить сценарий VBA, пока нажато с интервалом 50мс, принято начальное состояние – нажато).

Упражнение 2. *Экспорт данных в Excel*

Теперь сохраним график изменения уровня с интервалом в 1 секунду в таблице Excel. На самом деле, использовать Excel для записи исторических значений сигнала, да еще и на стороне клиентской программы нелогично, но мы выполним здесь эту задачу, как пример взаимодействия InfinityHMI со

сторонним приложением, поддерживающим технологию OLE Automation, исключительно потому, что данное приложение есть на большинстве компьютеров.

10. Сохраните файл под именем **Фамилия11.xml**.
11. Создайте кнопку «Экспорт значения в Excel». Это у нас будет отладочный вариант. В окончательном варианте данной кнопки не будет. Настройте для этой кнопки выполнение сценария VBA и откройте этот сценарий на редактирование.
12. Теперь необходимо установить связь с данными, предоставляемыми Excel. Для этого в редакторе VBA в меню Сервис -> Ссылки (Tools -> References) выберите Microsoft Excel 10.0 Object Library, как показано на рис. 5.4 (версия Excel, установленная на Вашем компьютере может быть другая).

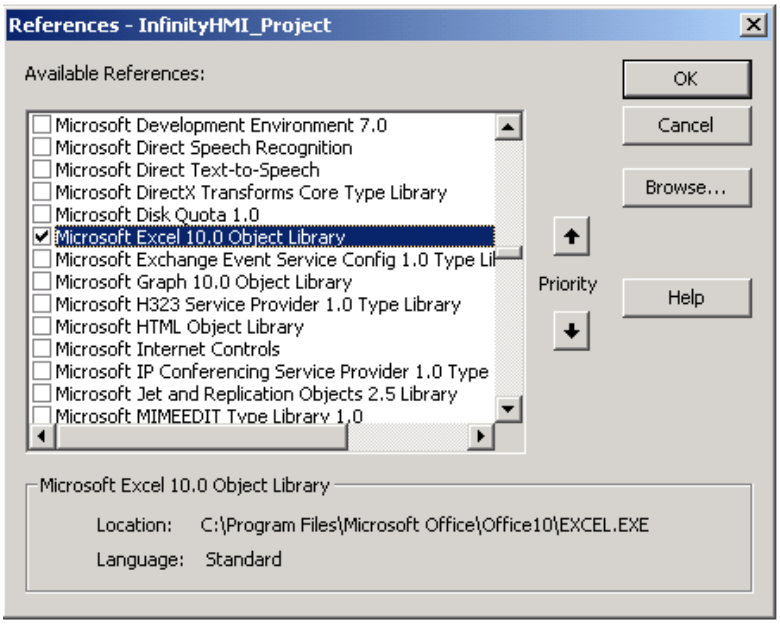


Рис. 5.4. Выбор компонента MS Excel для ссылки на него из скрипта

13. Выберите в дереве проекта объект Infinity HMI Objects ThisDisplay.
14. В окне редактирования исходного текста в левом верхнем выпадающем списке выберите элемент Общая область (General), а в правом верхнем углу Описания (Declarations) (обычно эта область открыта по умолчанию). Внесите здесь строку Option Explicit. В данной секции окна программы могут быть сделаны объявления глобальных переменных и введены прототипы функций, используемых во всех процедурах:

    Option Explicit
    Public g_Excel_App As Excel.Application
    Public g_Excel_Book As Excel.Workbook
    Public g_Excel_Sheet As Excel.Worksheet
15. В окне программы объекта ThisDisplay в левом верхнем выпадающем списке выберите элемент GwxDisplay, а в правом верхнем углу PreRunTimeStart. При этом будет сгенерирована процедура - обработчик

данного события. Данная процедура будет вызываться всякий при переходе в режим исполнения. Запишете в нее следующий код:

```
Private Sub GwxDisplay_PreRuntimeStart()
    ' Открыть Excel и сделать его видимым
    Set g_Excel_App = CreateObject("Excel.application")
    g_Excel_App.Visible = True
    ' Открыть лист Excel
    Set g_Excel_Book = g_Excel_App.Workbooks.Add
    Set g_Excel_Sheet = g_Excel_Book.Worksheets(1)
End Sub
```

16. Лишние процедуры, сгенерированные VBA можно удалить. В конечном итоге окно ThisDisplay должно выглядеть так, как на Рис. 5.5.

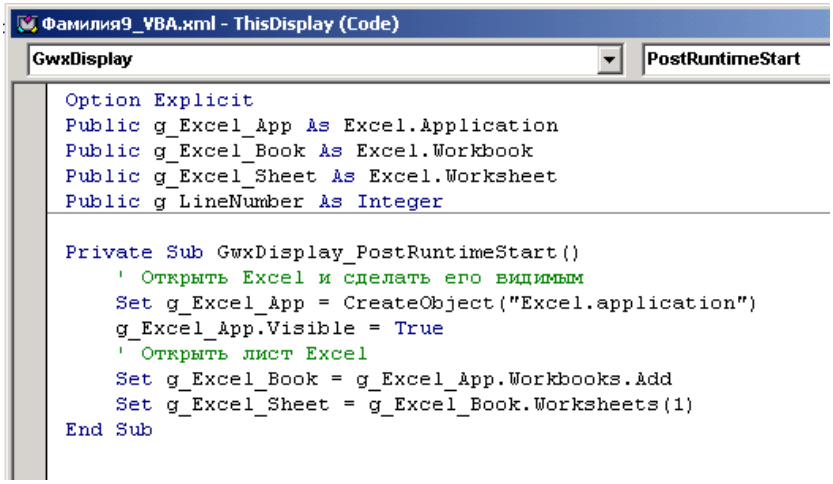


Рис. 5.5. Исходный текст (не законченный) процедуры старта экранной формы

17. Выберите в дереве проекта ветку GwxExcelExport_Main и в процедуру ExcelExport запишите код для сохранения значения сигнала Level в очередную строку первого столбца текущего листа Excel.

Как получать значение OPC сигнала было рассмотрено выше.

Вам могут потребоваться дополнительная глобальная переменная (для подсчета номера строки), инициализировать данную переменную необходимо при обработке события GwxDisplay_PreRuntimeStart().

Для значений в Excel можно воспользоваться, например, следующей конструкцией:

```
ThisDisplay.g_Excel_Sheet.Range(sCellName) = …
```

где sCellName это строка, идентифицирующая ячейку Excel, например, "A1".

Для того, чтобы перевести число в строку можно воспользоваться, например, следующей конструкцией Visual Basic:

```
Dim sLineName As String
sLineName = Str(g_LineNumber)
sCellName = "A" + Mid(sLineName, 2, Len(sLineName) - 1)
```

где **g_LineNumber** это номер строки.

18. Убедитесь, что созданный Вами сценарий корректно работает. В текущем варианте при переходе в режим исполнения у Вас должен открываться Excel и в нем создаваться несколько листов (так как это настроено по

умолчанию в Excel). В активном листе при нажатии на кнопку «Экспорт значения в Excel» должно появляться значение уровня жидкости в резервуаре. Каждое очередное значение должно появляться в новой строке. Закрывать Excel до закрытия мнемосхемы (перехода в режим редактирования) делать не стоит, так как это приведет к ошибке при нажатии на кнопку «Экспорт значения в Excel». При каждом запуске мнемосхемы должна открываться новая книга Excel. Если все работает правильно, то можно переходить к следующему пункту.


Некоторые сложности могут возникнуть при наличии антивирусных программ. Если на Вашем компьютере установлены антивирусные программы, то они могут не позволять приложению InfinityHMI запускать Excel, как OLE сервер. В этом случае, необходимо настроить Ваши антивирусные программы.
Например, программа OfficeGuard из пакета «Антивирус Касперского» по умолчанию не допускает запуска таких приложений, а спрашивает пользователя разрешения, при этом в области нотификации Windows появляется иконка OfiiceGuard, щелкнув на которую можно указать действия антивирусной программы в этом случае.

19. Теперь можно сделать окончательный вариант экспорта: вместо кнопки сделайте какому-нибудь объекту вызов сценария VBA при помощи динамики «Динамическое действие». В качестве имени сценария укажите имя сценария из созданной ранее кнопки. Настройте запуск сценария один раз в 1000 мс.
20. По желанию можно сделать автоматическое формирование графика средствами Excel, можно не показывать файл на экране и сохранять его автоматически при переходе в режим редактирования или выходе.

На этом лабораторная работа закончена

Контрольные вопросы

1. Что позволяет технология OLE?
2. Что такое VBA?
3. За что отвечает класс GwxSymbol?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе.

Лабораторная работа №6. Создание универсальных экранов

Цель работы

1) изучение возможностей InfinityHMI по созданию универсальных экранов для управления и контроля группой одинаковых технологических объектов (на примере управления насосными агрегатами).

Теоретическая часть

При решении задач АСУТП, часто возникает необходимость диспетчеризации группой однотипных технологических объектов, например, насосами насосной станции, однотипными станками с ЧПУ цеха. Для решения этой задачи используется технология псевдонимов.

Программа работы

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1. *Создание универсального экрана при помощи замены псевдонимов*

Создадим мнемосхему, из которого будет открываться другая мнемосхема (универсальная) для изменения режима работы насосов, расположенных на основной мнемосхеме. Для реализации этого используются псевдонимы. Пусть количество насосов равно четырем.

Откройте среду разработки Infinity HMI и сохраните новый файл как ФамилияXX_2.grf. В параметрах проекта укажите размер документа 600×350. Настройте параметры окна в режиме исполнения как показано на **Ошибка! Источник ссылки не найден.**. Расположите на экранной форме соответствующие объекты как показано на Рис. 6.2. Объект «Насос» находится в библиотеки символов Library_symbols.xml. Перенесите из данной библиотеки объект насос. Задайте ему динамику.

1. Для объекта кнопка «ВКЛ» задайте динамику «Действие» для передачи значения True источнику {{<<server>><<lab>><<pump>>.TU.on}}, а также динамику «Скрыть» для блокирования объекта, если значение источника {{<<server>><<lab>><<pump>>.TU.on}} равно True.
2. Для объекта кнопка «ВЫКЛ» задайте аналогичные динамики только передавать значение False и блокировать, когда False.
3. Определите значения псевдонимам, которые Вы указали в источниках. Для этого щелкните правой кнопкой мыши на форме и выберете пункт «Редактировать псевдонимы». В появившемся окне в поле «Определение» укажите следующие значения: для псевдонима «server» - Elesy.DualSource\, «lab» - Lab.plc05..Определение псевдонима «pump» будет меняться в зависимости от выбранного насоса.

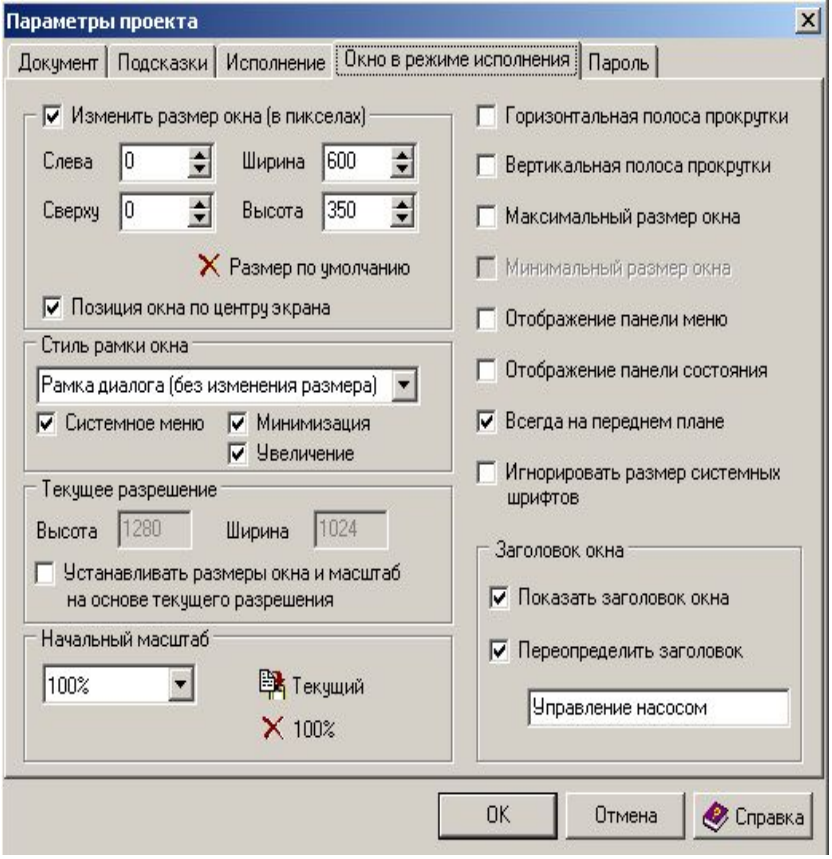


Рис.6.1. Параметры проекта

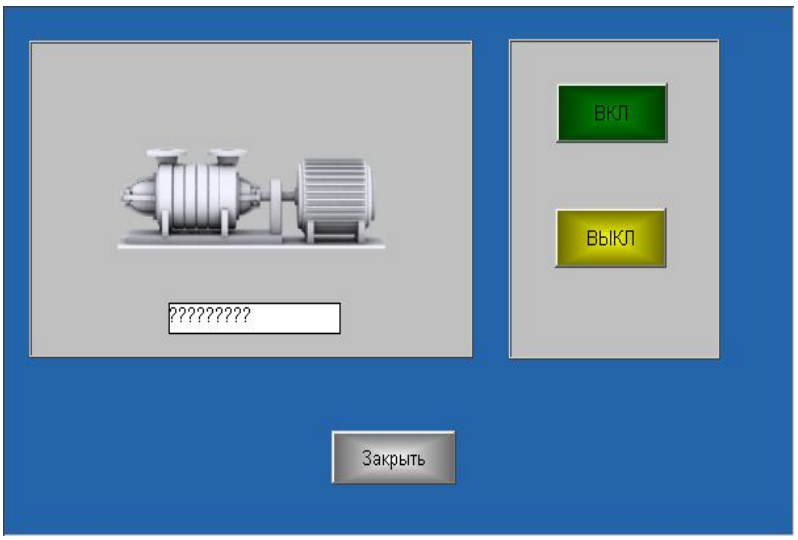


Рис. 6.2

4. Создайте новый документ, сохраните его как ФамилияXX_1.xml.
5. Расположите на экранной форме четыре объекта «Насос с динамикой» из библиотеки символов Library_symbols.xml.
6. Расположите над каждым объектом «Насос с динамикой» объекты «значение параметра», в качестве источника укажите {{<<server>><<lab>><<pump>>.name}}.
7. Для каждого из состояний включен и выключен объекта «Насос с динамикой» (всего состояний три: включен, выключен и неактивен) укажите в качестве источника x={{<<server>><<lab>><<pump>>.TU.on}} и настройте действие

«Скрыть» таким образом, чтобы при включенном состоянии насос был зеленного цвета, при выключенном – желтого.

8. Определите псевдонимы «server», «lab» аналогично пункту 5.

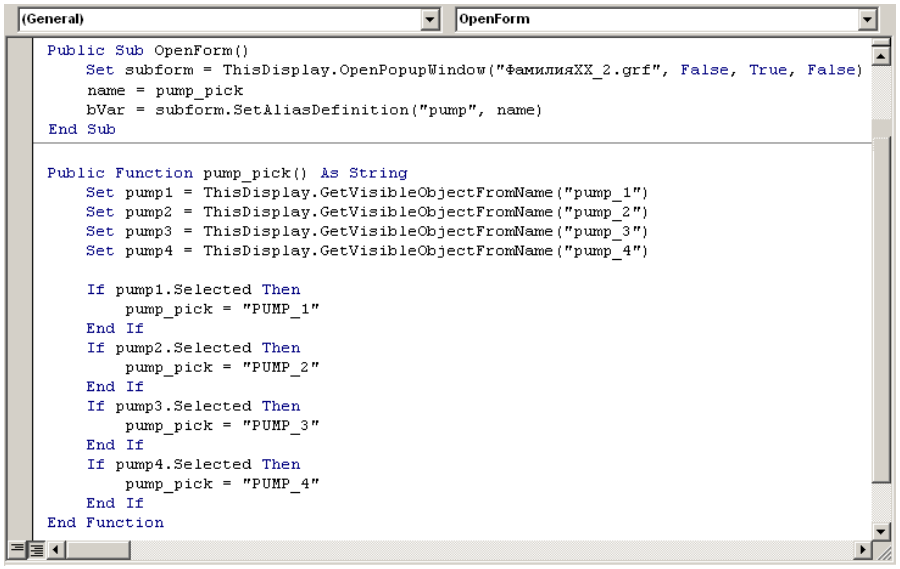


Рис. 6.3

9. Задайте для каждого насоса соответствующее определение псевдонима «pump». Для этого выделите насос, щелкните по нему правой кнопкой мыши и выберете пункт «Редактировать псевдонимы». В появившемся окне Вы увидите псевдонимы только данного объекта. Укажите определение псевдонима «pump» следующим образом: для насоса№1 – PUMP_1, для насоса№2 – PUMP_2, для насоса№3 – PUMP_3, для насоса№4 – PUMP_4.
10. Создайте новый модуль, в котором опишите процедуру открытия всплывающего меню OpenForm, как показано на Рис. . В данной процедуре задается определение псевдонима «pump». Функций pump_pick() возвращает значение псевдонима “pump” в зависимости от того, какой насос был выбран.
11. Создайте динамики «Действие» для объектов «Насос с динамикой», чтобы при щелчке на насос появлялось всплывающее окно.

В режиме исполнения окна должны выглядеть аналогично Рис. и Рис. .

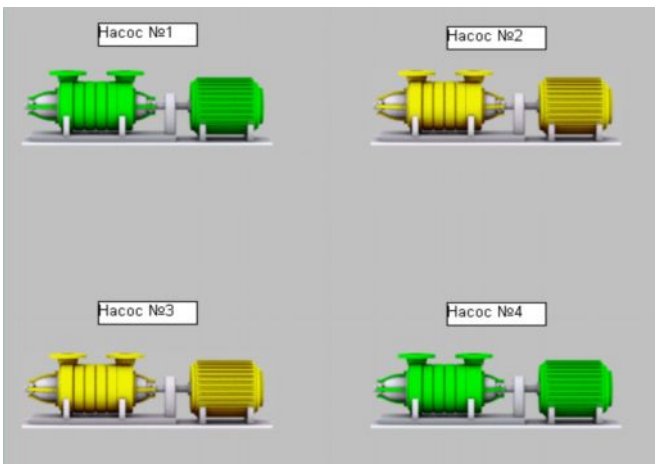


Рис. 6.4

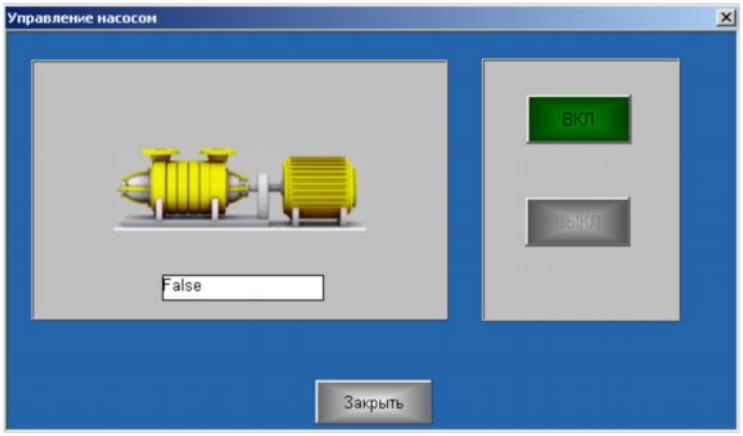


Рис. 6.5

Упражнение 2. *Повторить предыдущее упражнение таким образом, чтобы определение всех псевдонимов и в главном, и во всплывающем окне осуществлялось с помощью VBA.*

Для этого в процедуре GwxDisplay_PreRuntimeStart() задайте определение всех псевдонимов, используя функцию SetAliasDefinition, следующим образом: «pump» для первого насоса – «PUMP_1», «pump» для второго – «PUMP_2», «pump» для второго – «PUMP_3», «pump» для второго – «PUMP_4», «lab» - «Lab.plc05.», «server» - «Elesy.DualSource\». Для того чтобы определить разные значение для одного псевдонима «pump», вызывайте процедуру SetAliasDefinition для динамики, а не для всей формы или объекта. На этом лабораторная работа закончена.

Контрольные вопросы

1. Для решения каких практических задач используется технология универсального экрана?
2. Опишите последовательность шагов по формированию универсального экрана.
3. Что такое псевдоним и для чего он используется?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе

Лабораторная работа №7. Встраивание в мнемосхемы ACTIVE-X компонент

Цель работы:

1) получить практические навыки использования технологии встраивания ActiveX компонент в мнемосхемы InfinityHMI.

В процессе работы необходимо встроить два наиболее часто используемых компонента: InfinityAlarms и InfinityTrends.

Теоретическая часть

InfinityAlarms – клиентское приложение для отображения сообщений о событиях и авариях. InfinityAlarms предоставляет оперативную и историческую информацию в удобном для восприятия виде и ранжировании в зависимости от типа и степени важности сигнала. Полученные сообщения могут сопровождаться голосовым оповещением. Все сообщения могут квитироваться диспетчером.

InfinityAlarms позволяет задавать цвета для сообщений разных типов и важностей. Реализация в виде ActiveX-компоненты, позволяет отображать оперативную и историческую информацию о событиях в других приложениях.

InfinityTrends – клиентское приложение для построения графиков изменений технологических параметров. InfinityTrends позволяет отображать изменения значений технологических параметров в виде графиков или в табличном виде.

InfinityTrends позволяет работать как в историческом режиме, когда данные об изменении значения сигнала поступают из сервера истории так и в оперативном режиме (режим самописца), когда данные поступают из сервера ввода-вывода в реальном режиме времени.

Программа работы

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1. *Встраивание InfinityAlarmsView компонента*

1. Настройте генерацию алармов для сигнала Level на достижение предельного верхнего уровня и на достижение предельного нижнего уровня.
2. Создайте и сохраните фильтр, который позволяет просматривать алармы только данного сигнала.
3. Запустите программу InfinityAlarms с ключом FILECONFIG для записи ini-файла и настройте отображение оперативных алармов так, чтобы было видно только время сигнала, значение сигнала и текст сообщения.
4. Загрузите в InfinityHMI файл **Фамилия10.xml** и сохраните его под именем **Фамилия12.xml** Установите на экранную форму ActiveX компонент InfinityAlarmView. Для этого нажмите кнопку  на панели инструментов или выберите пункт OLE в меню рисование и выберите прямоугольную

область на экране. После этого появится диалог, как показано на Рис. . Выберите в нем элемент управления «InfinityAlarmView Control» и нажмите OK. После этого в выбранном Вами прямоугольнике появится элемент управления аналогично тому, как показано на Рис. .

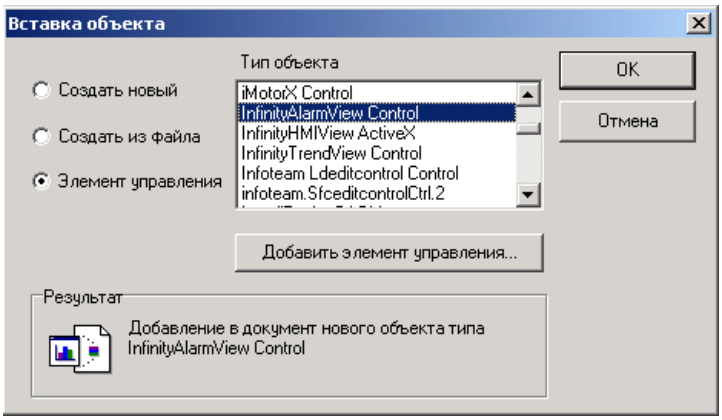


Рис. 7.1. Выбор ActiveX объекта, вставляемого в мнемосхему

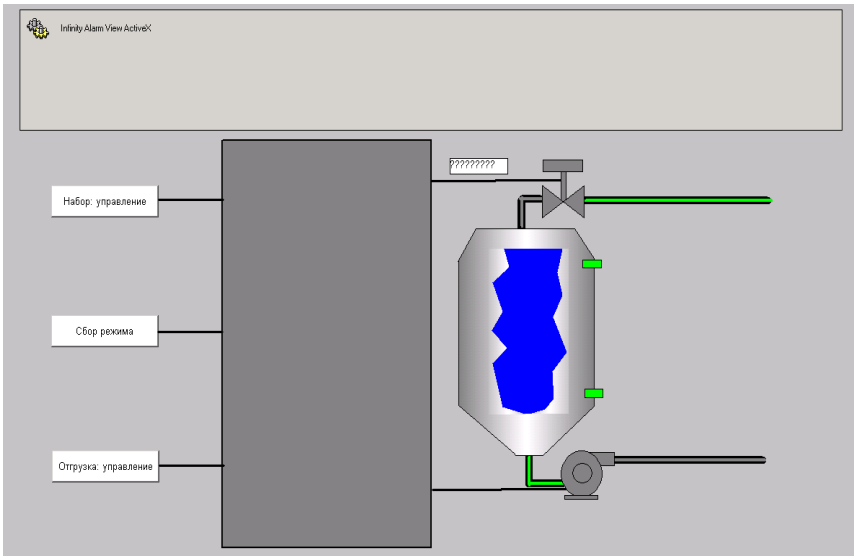


Рис. 7.2. Вид мнемосхемы со вставленным ActiveX компонентом

5. Активируйте добавленный Вами элемент управления и при помощи кнопки  на панели VisualBasic (или меню Макрос → Окно свойств) просмотрите окно свойств данного элемента управления. Отключите опцию AutoActivateAlarms, задайте имена файлов конфигурации и фильтра, как показано на Рис. . Имена файлов лучше задавать с абсолютными путями, но Вы также можете указать относительный путь. Однако, в этом случае текущим путем будет путь, указанный при старте InfinityHMI (если Вы не измените этот путь в дальнейшем). Прочие свойства лучше не трогать.

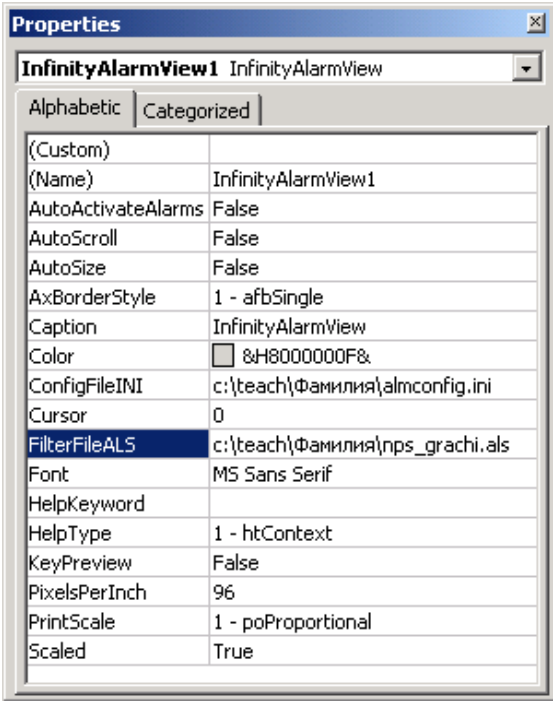


Рис. 7.3. Настройка свойств компонента InfinityAlarmView

6. В событии PostRuntimeStart модуля ThisDisplay (в окне VisualBasic) внесем одну строку: InfinityAlarmView1.ActivateAlarms.
7. Запустите мнемосхему, убедитесь, что алармы работают.
8. Свойства ConfigFileINI и FilterFileALS равно как и все прочие, которые Вы можете видеть в инспекторе свойств можно менять и в процессе работы мнемосхемы при помощи VBA.

Упражнение 2. *Встраивание InfinityTrendsView компонента*

Встроим в мнемосхему графики при помощи InfinityTrendView компоненты.

9. Сохраните файл **Фамилия12.xml** под именем **Фамилия13.xml**.
10. Аналогично алармам добавьте на мнемосхему компонент InfinityTrendView.
11. Подробный список свойств и методов для данного компонента можно посмотреть в главе 7 руководства пользователя.

Упражнение 3. *Работа со свойствами ActiveX компонента с помощью VBA*

В InfinityHMI существует возможность изменения файла конфигурации и фильтра для компоненты InfinityAlarmView при помощи VBA непосредственно в процессе работы мнемосхемы. Фильтр создается в модуле Infinity Alarms.

Предполагается, что два или более файлов с фильтрами алармов уже существуют. Если файлы отсутствуют, необходимо запустить Infinity Alarms, создать несколько фильтров и сохранить их как **Фамилия1.als**, **Фамилия2.als** и т.д.

12. Откройте среду разработки InfinityHMI, сохраните файл как **Фамилия14.xml**.
13. Установите на экранную форму ActiveX компонент InfinityAlarmView. Для этого нажмите кнопку  на панели инструментов или выберите пункт OLE в меню Рисование и выберите прямоугольную область на экране. После этого появится диалог, как показано на Рис. .

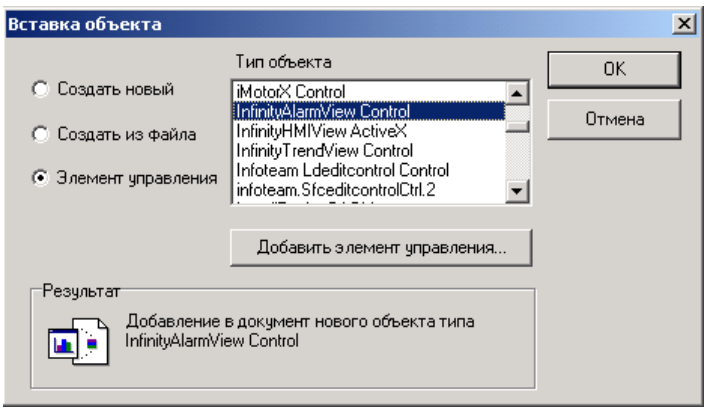


Рис. 7.4. Вставка ActiveX компонента

14. Выберите в нем элемент управления «InfinityAlarmView Control» и нажмите OK. После этого в выбранном Вами прямоугольнике появится элемент управления аналогично тому, как показано на Рис. .

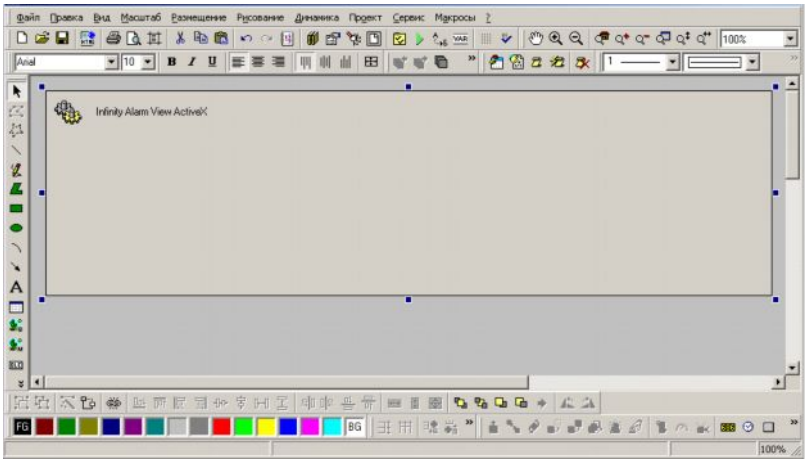


Рис. 7.5. Вид главного окна InfinityHMI со встроенным ActiveX компонентом

15. Выделив появившийся элемент управления, вызовите правым щелчок мыши контекстное меню и выберете в нем пункт «Объект InfinityAlarmView Control», затем «Properties». В результате появится окно «Свойства», как показано на Рис. . Файл фильтра и конфигурации можно не задавать. Также следует выключить опцию «Автоматическая активация».

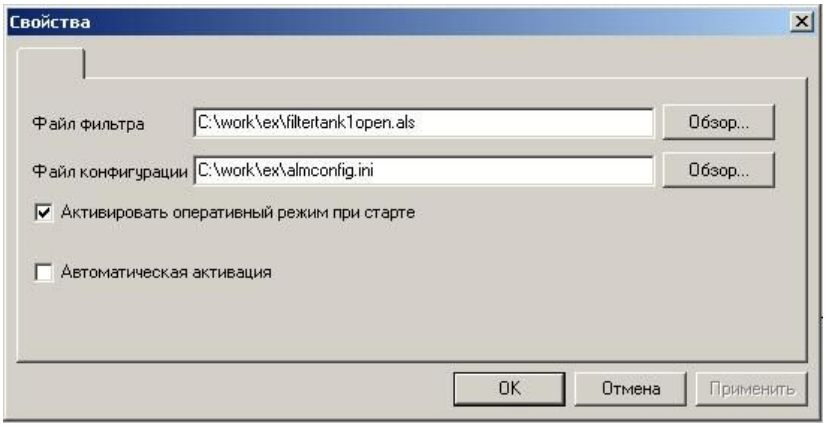


Рис. 7.6. Окно свойств компонента InfinityAlarmsView

16. Откройте Редактор Visual Basic либо нажав кнопку  на панели инструментов, либо выбрав пункт Редактор Visual Basic в меню Макросы.
17. Откройте окно просмотра кода модуля ThisDisplay, в котором описываются те действия, которые будут выполнены при переходе мнемосхемы в режим исполнения. Для этого необходимо дважды щелкнуть по значку ThisDisplay в Project Explorer (дерево проекта), либо, вызвав контекстное меню щелчком правой кнопки мыши по значку ThisDisplay в Project Explorer, выбрать пункт «View Code».
18. Теперь необходимо программно активировать InfinityAlarmView, как показано на Рис. .7.

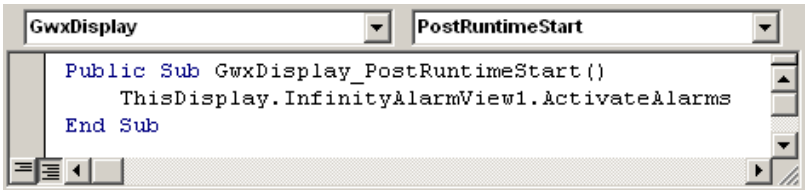


Рис. 7.7. Исходный текст активации ActiveX компонента

19. Создайте новый модуль в данном проекте. Для этого выберете в контекстном меню проекта пункт «Insert», затем «Module», либо пункт «Module» в меню «Insert». В новом модуле создайте процедуру, например, f1, для изменения свойства объекта InfinityAlarmView FilterFileALS, как показано на Рис. . В данной процедуре при четном или нулевом нажатии свойство FilterFileALS устанавливается в значение *"C:\teach\Фамилия\Фамилия1.als"*, иначе в значение *"C:\teach\Фамилия\Фамилия2.als"*.

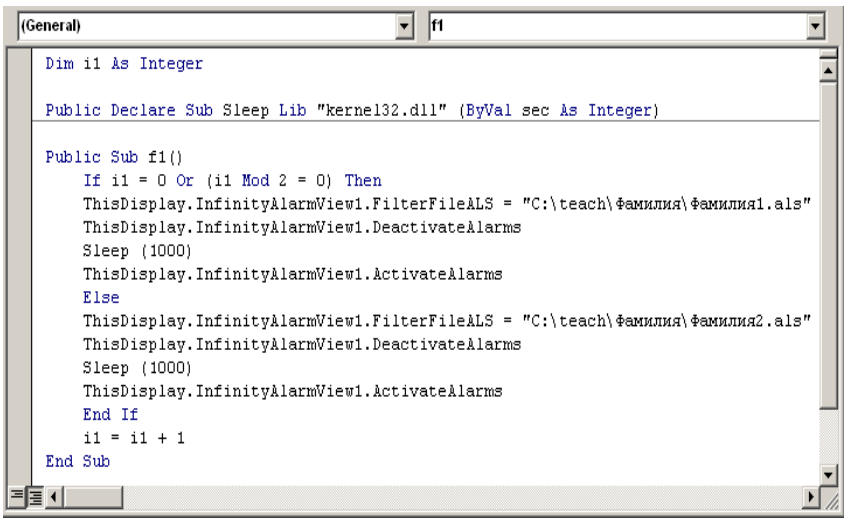


Рис. 7.8. Исходный текст процедуры изменения свойств объекта

20. Рассмотрим подробнее данную процедуру. Для корректного закрытия и открытия фильтра объекта InfinityAlarmView организована задержка. Для этого используется функция Sleep. Ее использование сводится к обращению к соответствующей функции из динамической библиотеки Kernel32. В качестве параметра передается целое число, обозначающее время задержки в миллисекундах. Вообще говоря, использование процедур типа Sleep в мнемосхемах не приветствуется (во всяком случае, не стоит делать длительные задержки) так как это может привести к тому, что какая либо другая логика в мнемосхеме также привязанная ко времени может не успеть отработать. Но в данном случае мы делаем задержку на одну секунду, поскольку мы точно знаем, что это не повлияет серьезно на диспетчерский контроль. Длительность задержки, которую нужно указать зависит от множества различных факторов и, в простейшем подбирается экспериментально.
21. Расположите на экранной форме среды разработки Infinity HMI динамический объект «Кнопка». Настройте самостоятельно цвет заливки объекта, назовите объект «Изменить фильтр». Добавьте динамику «Действие» и настройте ее как показано на Рис. 7.9. В результате при нажатии на кнопку будет выполняться процедура f1, описанная в модуле Module1.

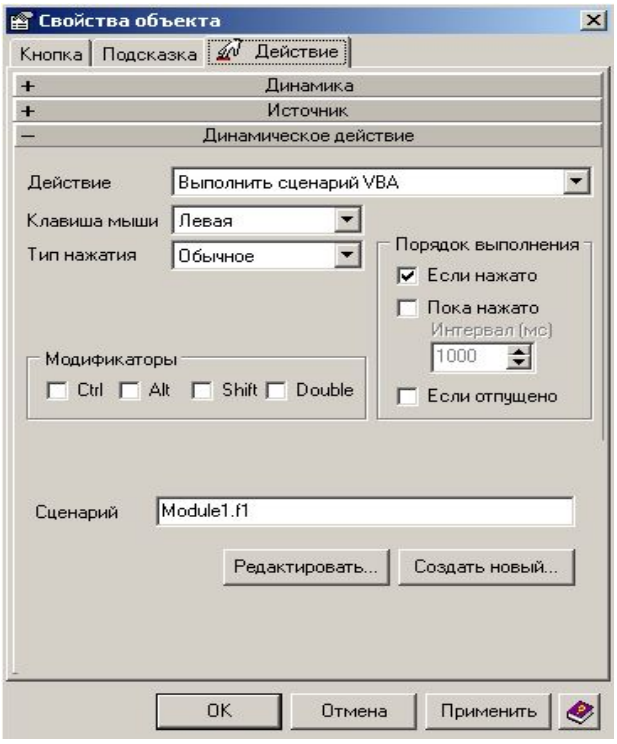


Рис. 7.9. Настройка вызова процедуры на нажатие кнопки

22. Для проверки корректности работы построенной Вами мнемосхемы запустите демо-проект «Общая схема РДП «Грачи». Затем в Вашей среде разработки Infinity HMI перейдите в режим исполнения. Осуществите какие-нибудь действия в демо-проекте, например, откройте/закройте задвижку, включите/отключите набор какого-либо резервуара. Убедитесь, что события отображаются в InfinityAlarmView на мнемосхеме, как показано на Рис. 7.10.

Стоит сохранять проект перед входом в режим исполнения!

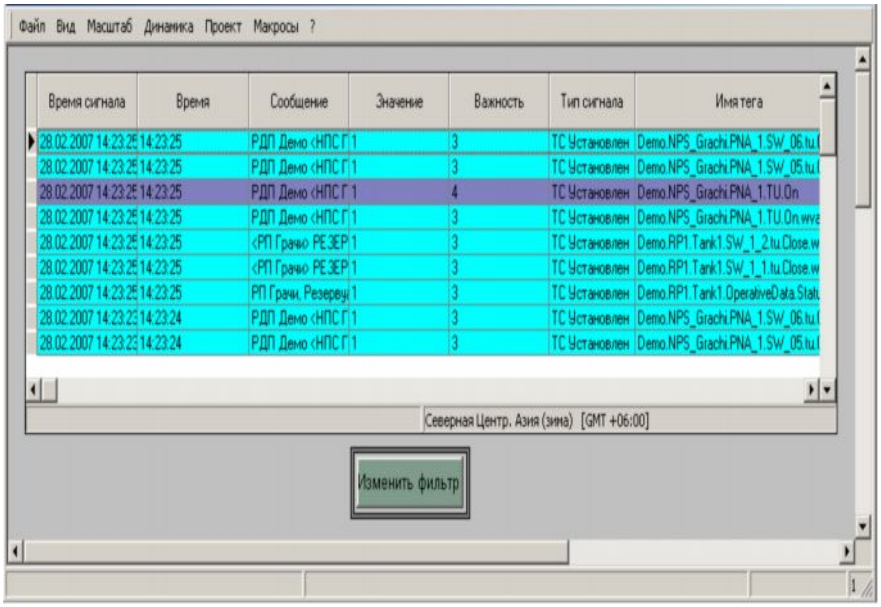


Рис. 7.10. Мнемосхема со встроенным компонентом InfinityAlarmView

23. Работа со свойством ConfigFileIni объекта InfinityAlarmView осуществляется аналогичным образом.

На этом лабораторная работа закончена

Контрольные вопросы

1. Что такое Infinity Alarms?
2. Как на экранную форму ActiveX установить компонент InfinityAlarmView?
3. Какая функция используется для задержки закрытия-открытия фильтра объекта InfinityAlarmView?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе

Лабораторная работа №8. Манипуляция объектами мнемосхемы

Цель работы:

1) Приобретение практических навыков по использованию скриптов VBA при обращении с помощью мышки к графическим объектам, расположенным на мнемосхеме.

Теоретическая часть

Основным инструментом работы диспетчера в процессе автоматизированного управления технологическим процессов является мышка. С ее помощью графические объекты экранных форм «кликуются» и тем самым активизируется выполнение VBA скриптов. Это в свою очередь ведет к активизации действий исполнительных подсистем (подсистем технологических переключений, запуска исполнительных органов, экранных форм и др.) Задачей лабораторной работы является управление графическим объектом на мнемосхеме. Лабораторная работа будет выполняться в 3 этапа.

1. Необходимо заставить двигаться объект, созданный вручную, по траектории эллипса.
2. Необходимо приобрести практические навыки самостоятельно, создавать объекты на мнемосхеме из VBA скрипта.
3. Необходимо освоить практические приемы при обращении к свойствам объектов мнемосхемы и научится использовать их для организации логики в скрипте.

Упражнение 1. *Движение объектов по эллиптической орбите*

24. Откройте среду разработки InfinityHMI, сохраните новый файл как Фамилия20.xml. Расположите на экранной форме объект «Эллипс», назовите его LabSun. Для этого в окне «Свойства объекта» укажите имя объекта как LabSun (Рис.). Задайте цвет заливки желтый, ширину и высоту – 90, остальные параметры произвольно. Расположите еще один объект «Эллипс», назовите его LabPlanet, задайте ширину и высоту 40, остальные параметры произвольно. Расположите третий объект «Эллипс», назовите его LabOrbita, задайте ширину и высоту 320 и 150 соответственно, цвет заливки – пустой, остальные параметры произвольно.

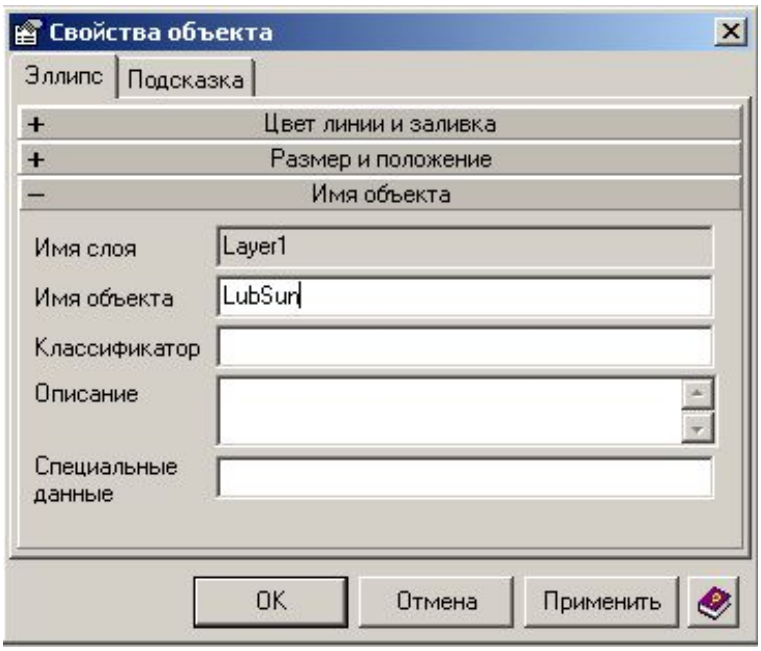


Рис. 8.1. Задание имени объекта в диалоге свойств объекта

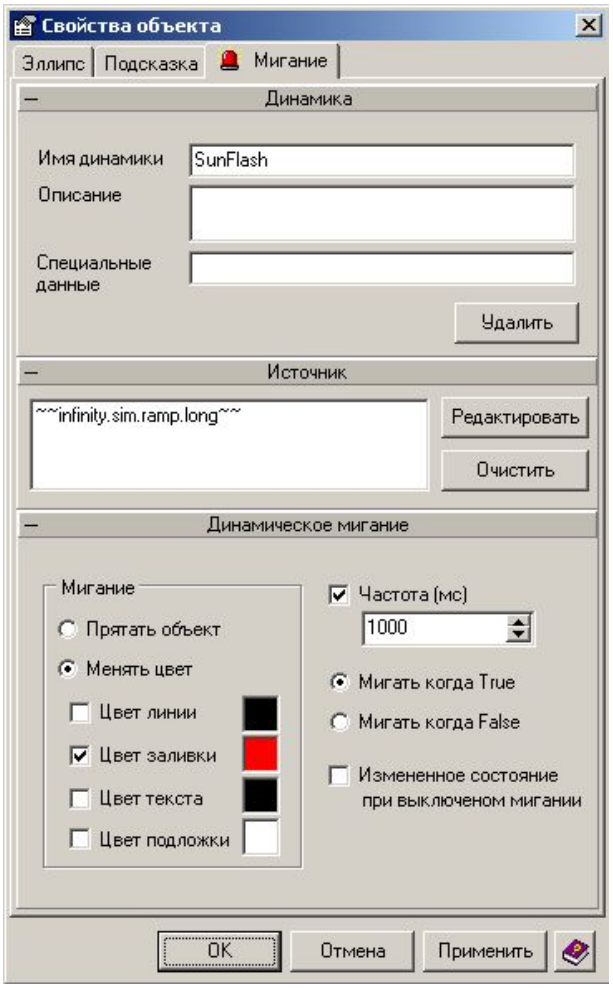


Рис. 8.2. Настройка динамики «Мигание» для объекта «LabSun»

25. Расположите созданные Вами объекты таким образом, чтобы объект LabSun был в центре объекта LabOrbit, а объект LabPlanet находился на линии объекта LabOrbit.

26. Задайте динамику «Мигание» для объекта LabSun, установите параметры динамики, как показано на Рис. Задайте динамику «Вращение» для объекта LabPlanet, установите параметры динамики, как показано на Рис. .

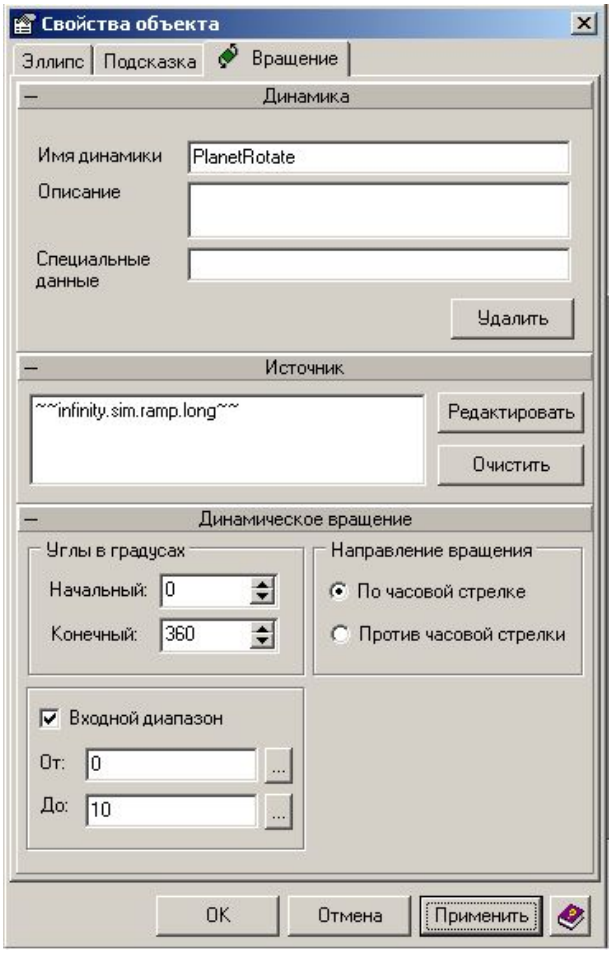


Рис. 8.3. Динамика «Вращение» для объекта LabPlanet

27. Реализуйте вращение объекта LabPlanet по линии объекта LabOrbita с помощью VBA. Для этого откройте редактор Visual Basic и создайте процедуру PlanetMovement в модуле Module1, в которой реализуйте алгоритм изменения координат x и y объекта Planet по закону синуса и косинуса, как показано на Рис. .
28. Переменные x1, y1 (начальные координаты), xc, yc (координаты центра) описаны как глобальные переменные и инициализированы в процедуре Create(), которая находится также в Module1 и выглядит как показано на Рис. . Значение xc определяется как положение X объекта LabSun плюс 45 (половина ширины объекта), yc - положение Y объекта LabSun плюс 45 (половина высоты объекта).

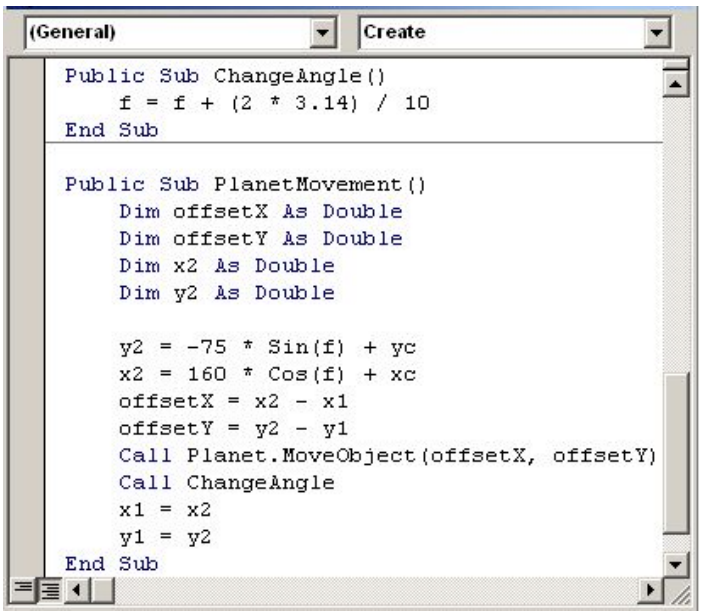


Рис. 8.4. Реализация алгоритма движения объекта по траектории эллипса

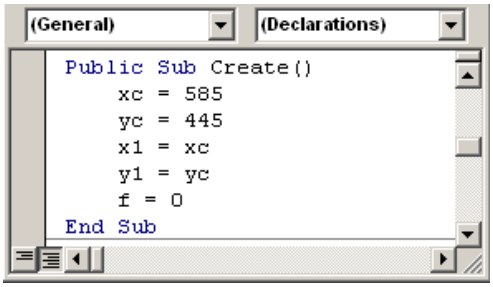


Рис. 8.5. Инициализация глобальных переменных

29. Расположите на Вашей мнемосхеме объект для осуществления движения планеты по орбите. Например, это может быть прямоугольник в качестве панели, на которой расположена построенная Вами модель. Настройте самостоятельно свойства отображения объекта. Добавьте динамику «Действие» и настройте ее таким образом, чтобы при запуске мнемосхемы начала с каким-то периодом (например, 1 раз в 200мс) выполняться процедура PlanetMovement. В результате при запуске мнемосхемы будет осуществляться движение планеты по орбите.
30. Проверьте корректность работы Вашей мнемосхемы. Результат должен выглядеть приблизительно, как показано на Рис. , т.е. в режиме исполнения Ваша мнемосхема должна при нажатии кнопки «Создать» создавать модель вращения планеты по солнечной орбите.

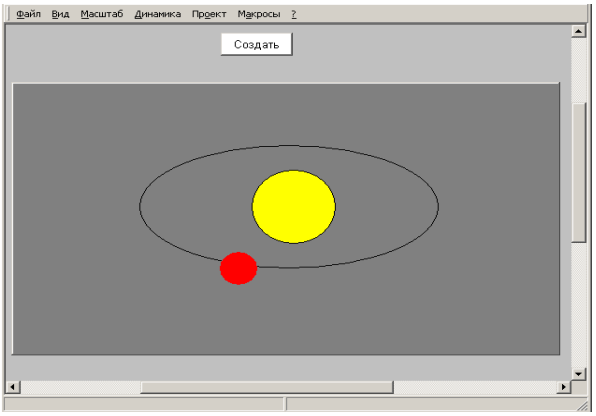


Рис. 8.6. Мнемосхема «Солнечная система» в действии

Упражнение 2. *Создание графических объектов при помощи VBA*

31. Сохраните Ваш файл как Фамилия21.xml.
32. Удалите с Вашей формы видимые объекты LabSun, LabPlanet, LabOrbita и реализуйте функцию создания этих объектов и их динамик с помощью VBA. Для этого откройте Редактор Visual Basic и реализуйте в методе Create() создание объектов LabSun, LabPlanet и LabOrbita. Для создания визуального объекта LabSun опишите объект Sun типа GwxEllipse, который будет связан с визуальным объектом с помощью метода CreateEllipse, в качестве параметров которого указываются расположение объекта на экранной форме, ширину, высоту, наличие заполнения, цвет заполнения, цвет линии, ширину линии, тип линии, наличие тени, цвет тени, тип краев, видимость объекта и имя объекта. Данный метод может быть вызван только (!) в режиме разработки, поэтому сначала необходимо остановить режим исполнения с помощью метода StopRuntime(), а после запустить режим исполнения с помощью команды StartRuntime(), как показано на Рис. 8.6.
33. Создайте еще два объекта типа GwxEllipse, назовите их Orbita и Planet. Аналогичным образом создайте визуальные объекты LabOrbita и LabPlanet с помощью метода CreateEllipse. Для объекта Orbita укажите свойство IsFilled как False, т.е. без заполнения, как показано на Рис. .
34. Расположите на экранной форме среды разработки Infinity HMI динамический объект «Кнопка». Настройте самостоятельно цвет заливки объекта, назовите объект «Создать». Добавьте динамику «Действие» и настройте ее так, что бы при нажатии на эту кнопку вызывалась процедура Create из модуля Module1.
35. Переведите Вашу мнемосхему в режим исполнения и убедитесь, что объекты действительно создаются при нажатии на кнопку. Когда Вы перейдете в режим разработки эти объекты останутся (!) и их нужно будет удалить вручную.

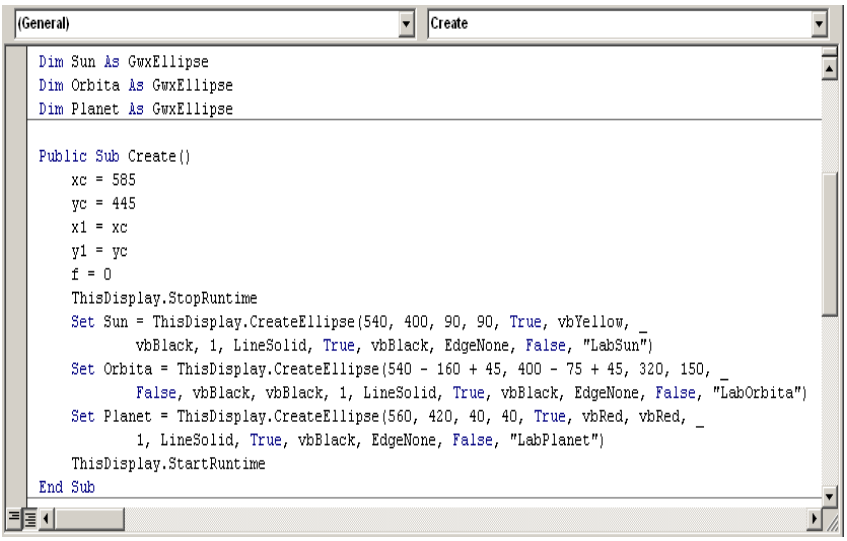


Рис. 8.7. Исходный текст процедуры создания объектов при помощи VBA

36. При каждом запуске Вашей мнемосхемы будут создаваться объекты, описанные в методе Create(). Осуществите автоматическое удаление всех Ваших объектов после выполнения мнемосхемы. Для этого в модуле ThisDisplay создайте метод GwxDisplay_PostRuntimeStop(), в котором удалите объекты с помощью методов DeleteDynamic и DeleteObject.
37. Проверьте корректность работы Вашей мнемосхемы.
38. Создайте динамику «Мигание» для объекта Sun с помощью метода CreateFlashDynamic, в качестве параметров которого укажите имя объекта, к которому привязывается данная динамика, имя динамики, скрывать объект или нет, мигание при установлении источника в значение false или true, наличие альтернативного состояния, изменение цвета заполнения, изменение цвета линии, изменение цвета тени, альтернативный цвет заполнения, линии и тени. Назовите объект динамика «SunFlash».
39. Данный метод используется также только (!) в режиме разработки.
40. Для работы динамики необходимо задать источник. Для этого в свойстве объекта SanFlash dataSource укажите имитационную переменную ~~infinity.sim.ramp.long~~.
41. Проверьте корректность работы Вашей мнемосхемы. Если она работает корректно, создайте динамику «Вращение» для объекта Planet с помощью функции CreateRotationDynamic, как показано на Рис. .8.
42. Результат выполнения Вашей мнемосхемы должен быть аналогичен результату в упражнении 1, т.е. в режиме исполнения Ваша мнемосхема должна при нажатии кнопки «Создать» создавать модель вращения планеты по солнечной орбите.

Упражнение 3. *Создание динамики при помощи скриптов VBA*

43. Сохраните построенную Вами мнемосхему как Фамилия22.xml.

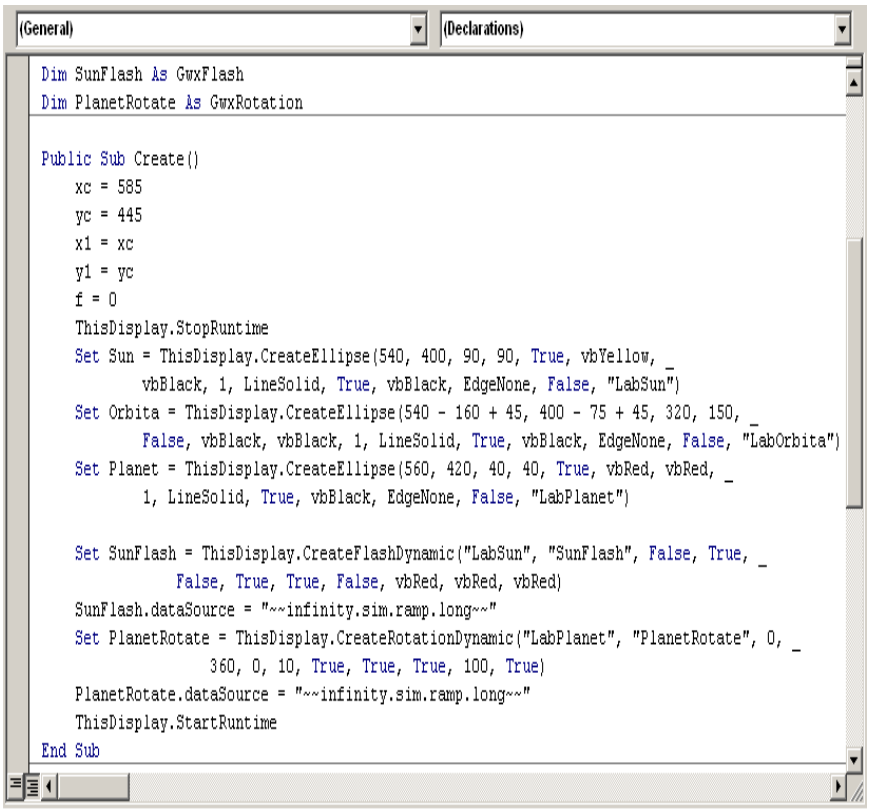


Рис. 8.8. Создание динамики в VBA скрипте

44. В конфигураторе сервера ввода/вывода Infinity создайте сигнал Teach.Inputs.IntRamp20 с амплитудой от 0 до 20 и периодом 20с.
45. Для считывания значения сигнала используется функция GetOPCValue из библиотеки OPCDualSource, как показано на Рис. .

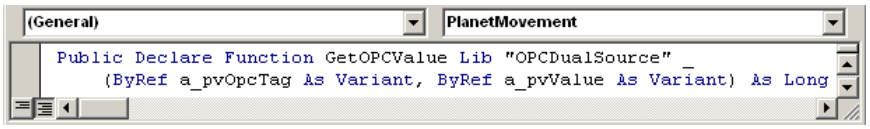


Рис. 8.9. Декларация для вызова внешней процедуры из VBA скрипта

46. Процедура перемещения объекта и изменения угла остаются без изменения, за исключением того, что для изменения угла поворота f используется значение Вашего сигнала (Рис. 8.1).
47. Результат должен быть аналогичен результату предыдущего упражнения, т.е. в режиме исполнения Ваша мнемосхема должна при нажатии кнопки «Создать» создавать модель вращения планеты по солнечной орбите.

Упражнение 4. *Доступ к свойствам объекта из VBA*

48. Сохраните Вашу мнемосхему как Фамилия23.xml.
49. Создайте поля для ввода значений вертикальной и горизонтальных осей орбиты. Для этого расположите на экранной форме Вашей мнемосхемы два динамических объекта «Значение параметра», задайте им уникальные имена AxisA и AxisB. В качестве источника данных задайте локальные переменные ~~planet.axisa~~ и ~~planet.axisb~~. В Module1 опишите два

объекта AxisA и AxisB типа GwxText и свяжите данные объекты с соответствующими визуальными объектами в процедуре Create, как показано на рис.8.11.

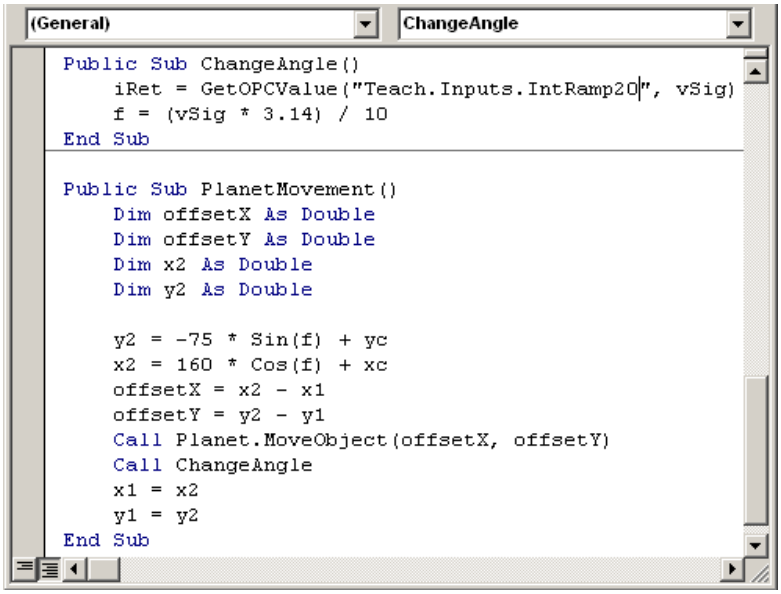


Рис. 8.1. Исходный текст процедуры движения объекта с использованием внешнего сигнала

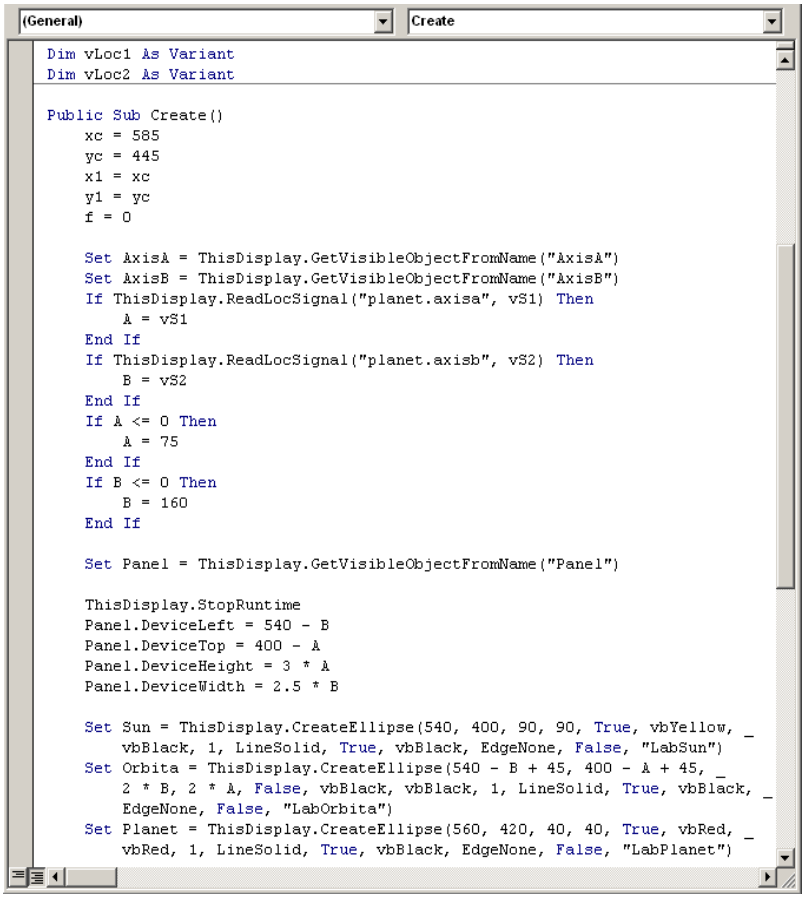


Рис. 8.11. Процедура создания объектов с использованием параметров из объектов мнемосхемы

50. Значения вводимых данных считываются с локальных переменных и записываются в переменные A и B. Если значение не введено, то используется значение по умолчанию (A=75, B=160).
51. При изменении размеров орбиты должны измениться размеры панели, расположение самой орбиты. Для этого описываем объект Panel типа GwxRectangle и связываем его с визуальным объектом «Panel». Задаем размеры панели в соответствии с размерами орбиты. Задаем координаты орбиты относительно солнца и ее размеры.
52. В процедуре PlanetMovement замените коэффициенты перед синусом и косинусом на переменные A и B соответственно.
53. Проверьте корректность работы Вашей мнемосхемы. В режиме исполнения Ваша мнемосхема должна при нажатии кнопки «Создать» создавать модель вращения планеты по солнечной орбите, размеры которой заданы в соответствующих полях ввода (Рис.).

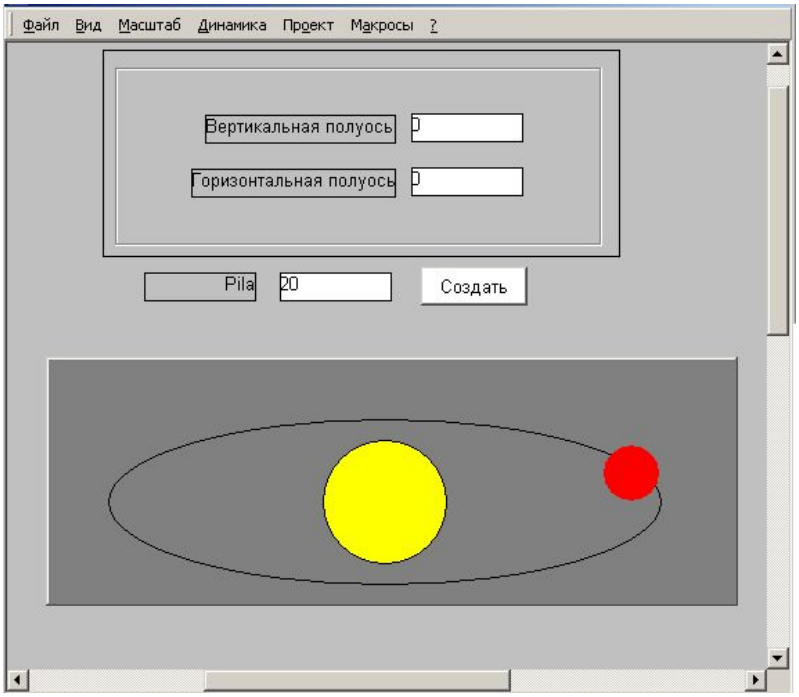


Рис. 8.12. Результирующая мнемосхема с использованием VBA скрипта

54. На этом лабораторная работа закончена

Контрольные вопросы

1. Что из себя представляет VBA скрипт ?
2. Зачем необходима активизация графических объектов на экранных формах?
3. Что означает термин автоматизированное управление?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе.

Лабораторная работа №9. Управление температурным объектом

Цель работы:

1) управление температурным объектом.

Программа работы:

Лабораторная работа содержит задания в виде упражнений. Все упражнения обязательны для выполнения.

Упражнение 1. *Создание сигналов в конфигурации контроллера*

1. Запустите программу ElsyTMPultPC.
2. Считайте конфигурацию контроллера. В контроллере может быть загружена конфигурация, не удовлетворяющая нашим требованиям. В этом случае нам необходимо будет ее изменить. Конфигурация, которая будет удовлетворять нашим требованиям, показана на Рис. 9.2.

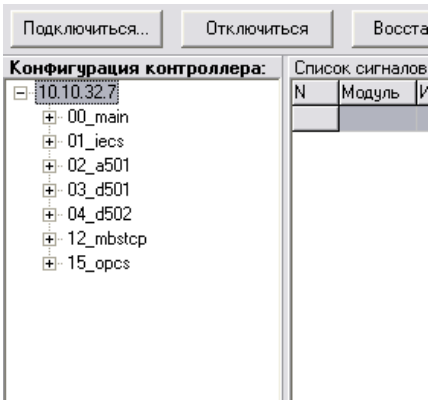


Рис. 9.2 Необходимая конфигурация контроллера

3. Создайте конфигурацию, аналогичную той, которая приведена на Рис. 9.2.
4. Загрузите созданную конфигурацию в контроллер.
5. Откройте конфигурацию в программе ElsyTMManager .
6. Маршрутизируйте сигналы телеизмерения (ТИ), телесигнализации (ТС), телеуправления (ТУ) и телерегулирования (ТР) с соответствующих модулей контроллера в OpenPCS. Для маршрутизации сигнала ТИ откройте закладку модуля ТА-501 «02_a501-> Сигналы->Сигналы выходные». Выделите сигнал AnalIn_2, нажмите на него правой кнопкой мыши и выберите пункт меню «Маршрутизировать в OpenPCS».
7. Проделайте ту же операцию с сигналом ТС (маршрутизируйте выходной сигнал с модуля ТД – 501 - сигнал DigIn_1).
8. Создайте новый сигнал в закладке выходных сигналов модуля 12_mbstcp. Для этого откройте закладку «12_mbstcp->Сигналы->Сигналы выходные». Нажмите на поле сигналов правой кнопкой мыши и выберите в появившемся меню пункт «Добавить сигнал». Опишите сигнал согласно рис.9.2.

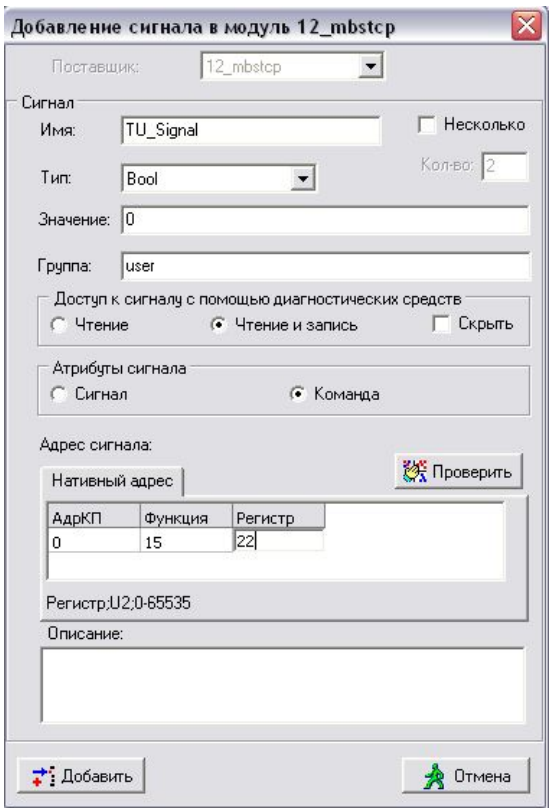


Рис. 9.3 Окно добавления сигнала TU_Signal

Здесь адрес состоит из трех полей:
- поле “АдрКП” – содержит адрес КП, но при создании сигнала всегда равен нулю, так как адрес КП задается в конфигурации процессорного модуля для всех сигналов одновременно;
- поле “Функция” – его назначение пояснено в следующей таблице:

Таблица поле «Функция»

Функция	Код функции Modbus
Чтение битовой ячейки (Coil)	1
Чтение битового входа (Input)	2
Чтение регистра (Holding Register)	3
Чтение входного регистра (Input Register)	4
Запись битовой ячейки (Coil)	15
Запись регистра (Holding Register)	16
Состояние связи с подчинен-ной станцией	255

- поле “Регистр” – адрес регистра хранения сигнала.

9. В этой же закладке добавьте сигнал телерегулирования TR_Signal. Опишите его согласно Рис. 9.4.

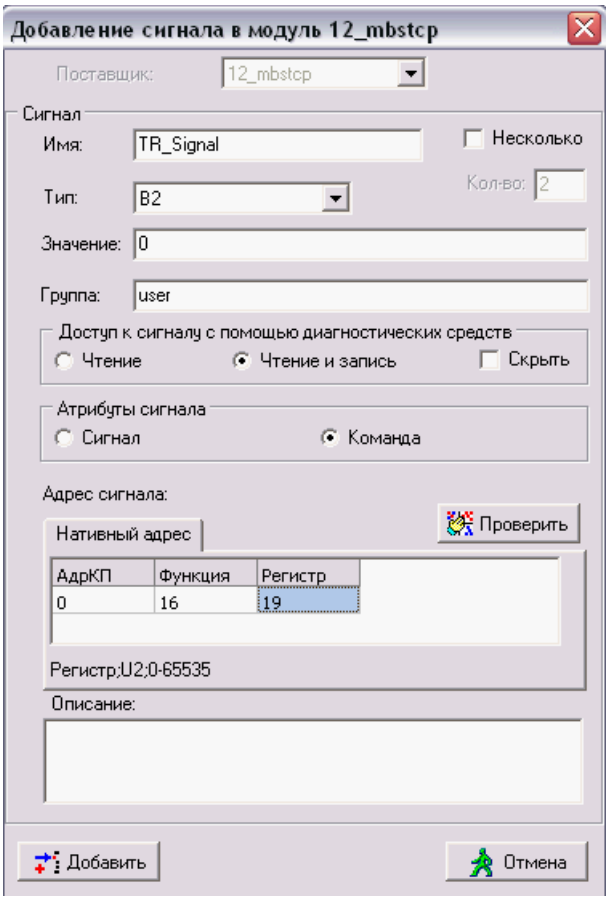


Рис. 9.4 Окно добавления сигнала TR_Signal

10. Маршрутизируйте оба сигнала в модуль OpenPCS.
11. В случае правильного выполнения маршрутизации при открытии закладки модуля OpenPcs: «15_opcs->Сингалы ->Сигналы входные» Вы увидите четыре маршрутизированных в модуль 15_opcs сигнала (Рис. 9.5).

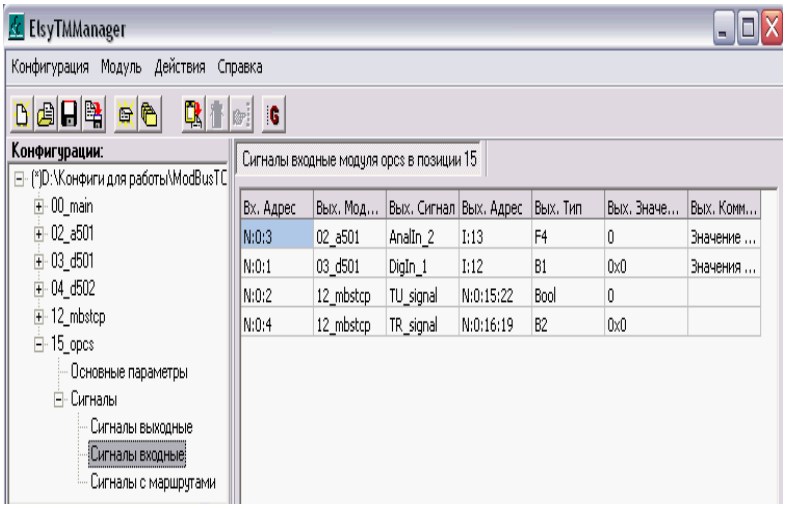


Рис. 9.5. Маршрутизированные в модуль 15_opcs сигналы

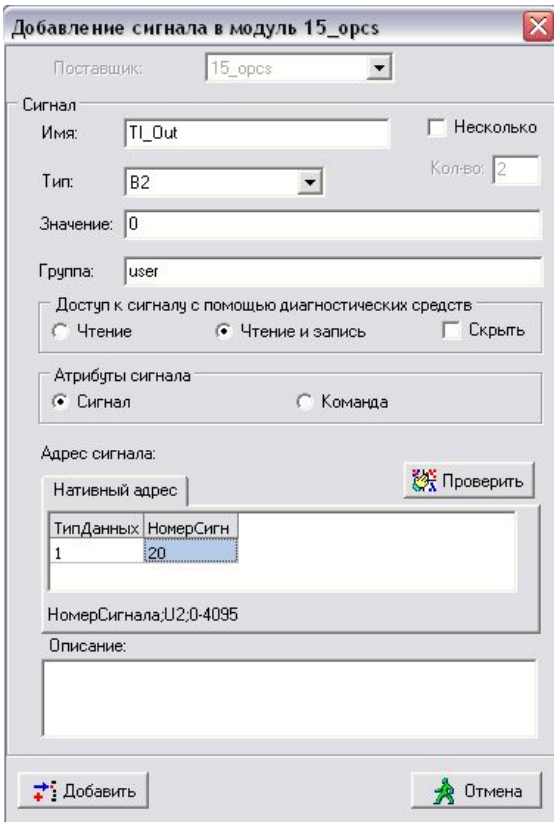


Рис. 9.5. Создание сигнала TI_OUT

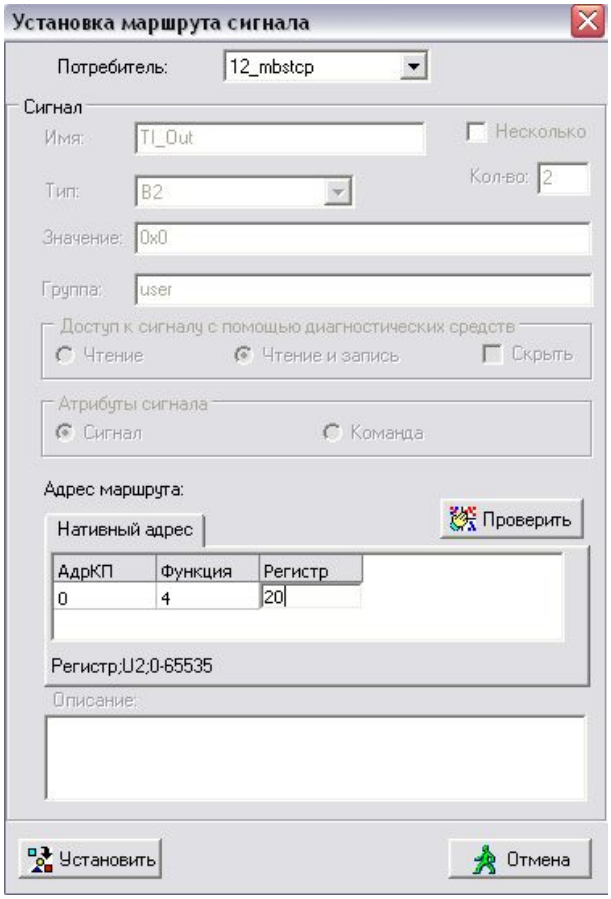


Рис. 9.6. Маршрутизация сигнала TI_Out

12. В редакторе конфигураций создайте выходной сигнал в модуле 15_opcs и настройте его параметры как показано на **Ошибка! Источник ссылки не найден.**.
13. Создайте еще два сигнала:

- TS_Out с номером сигнала 21, типом данных Bool;
- TU_Out и TR_Out с номерами сигналов 22 и 19 соответственно, тип данных – B1.

Маршрутизируйте сигналы TI и TS в модуль 12_mbstcp с параметрами маршрутов, приведенными на Рис. 9.6, Рис. 9.7

Здесь адрес состоит из трех полей:

- поле “АдрКП” - содержит адрес КП, но при создании сигнала всегда равен нулю, так как адрес КП задается в конфигурации процессорного модуля для всех сигналов одновременно;
- поле “Функция” – его назначение пояснено в следующей таблице:
- поле “Регистр” – адрес регистра хранения сигнала.

Таблица поле «Функция»

Функция	Код функции Modbus
Чтение битовой ячейки (Coil)	1
Чтение битового входа (Input)	2
Чтение регистра (Holding Register)	3
Чтение входного регистра (Input Register)	4
Запись битовой ячейки (Coil)	15
Запись регистра (Holding Register)	16
Состояние связи с подчиненной станцией	255

14. Маршрутизируйте сигналы TU_Out и TR_Out в модуль ТД-502. Индексный адрес – 0 и 2 соответственно (это означает, что маршрутизируемый байт будет адресоваться на первые и третьи восемь выходных сигналов соответственно).
15. Откройте “Основные параметры” модуля 12_mbstcp и измените параметры IP_1 и MASK_1 так, как показано на Рис. 9.8.
16. В меню “Действия” выберете пункт “Сгенерировать переменные OPCS в буфер обмена”.
17. Запустите OpenPCS и создайте программу на языке ST.
18. Поместите переменные из буфера обмена в область локальных переменных программы.
19. Опишите переменные и создайте программу так же, как это показано на Рис. 9.9. Программа перекладки данных

20. (комментарии можно пропустить).
21. Загрузите созданную программу в контроллер и включите её выполнение.

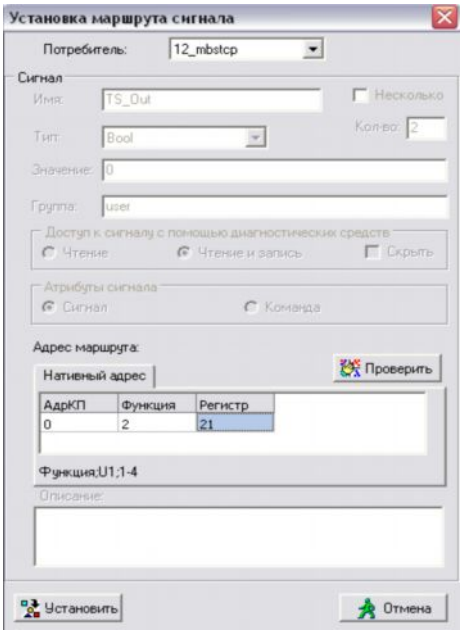


Рис. 9.7. Маршрутизация сигнала TS_Out

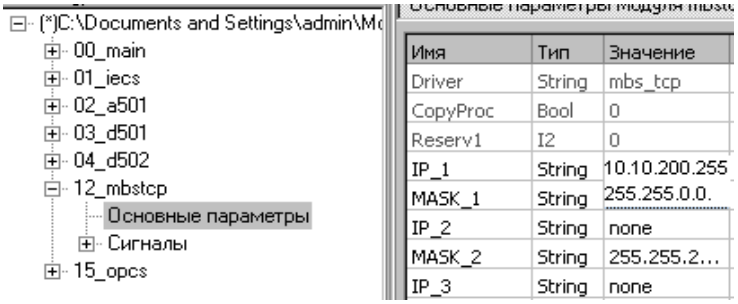


Рис. 9.8 Настройка IP-адресов

```
VAR
(*Выходные сигналы для подачи по модбасу*)
TI_Out          AT %QW0.0   :WORD;   (*N:1:20*)
TS_Out          AT %QX4.0   :BOOL;   (*N:1:21*)
(*Сигнал принимаемый с модбаса для телеуправления (TU) и телерегулирования (TR)*)
TU_signal       AT %IX15.0  :BOOL;   (*N:0:2*)
TR_signal       AT %IW18.0  :WORD;   (*N:0:4*)

(*Сигналы с модулей ввода*)
AnalIn_2        AT %ID0.0   :REAL;   (*N:0:3*)
AnalIn_4        AT %ID6.0   :REAL;   (*N:0:0*)
DigIn_1         AT %IB12.0  :BYTE;   (*N:0:1*)

(*Сигнал, адресумый на модуль дискретного вывода*)
TU_Out          AT %QB7.0   :BYTE;   (*N:1:22*)
TR_Out          AT %QB10.0  :BYTE;   (*N:1:19*)
(*Внутренние переменные для перекладки*)
TiiMB                       :Real;   (*Внутренняя переменная, которая принимает аналоговый сигнал*)
                                     (*В реальных проектах переменная может идти на дополнительную обработку*)
TSiMB                       :Bool;   (*Переменная принимает первый бит (первый сигнал) с модуля дискретного ввода*)
                                     (*В реальных проектах переменная может идти на дополнительную обработку*)
TUoMB                       :Bool;   (*Переменная для перекладки сигнала из модбаса в модуль дискретного вывода*)
                                     (*В реальных проектах переменная может идти на дополнительную обработку*)
TRoMB                       :word;   (*Переменная для перекладки сигнала телерегулирования из модбаса в модуль*)
                                     (*дискретного вывода. В реальных проектах переменная может идти на*)
                                     (*дополнительную обработку*)

END_VAR
```

```
(*Перекладка сигнала с модуля аналогового ввода в переменную, адресуемую в протокол модбас*)

TIiMB:=AnalIn_2;
TIiMB:=TIiMB*1000.0;
TI_Out:=Real_To_Word(TIiMB);     (*Переводим тип данных из Real в тип Word*)

(*Перекладка сигнала телесигнализации из модуля дискретного ввода в переменную, адресуемую в протокол модбас*)

TSiMB:=DigIn_1.0;    (*Перекладывается первый бит (сигнал с первого входа)*)
TS_Out:=TSiMB;

(*Перекладка сигнала телеуправления, приходящего с модбаса на первый выход модуля дискретного вывода*)

TUoMB:=TU_signal;
TU_Out.0:=TUoMB;     (*Сигнал перекладывается на первый бит (на первый выход)*)

(*Программа, которая переводит сигнал телерегулирования в битовую строку и посылать ее на модуль дискретного вывода*)
TRoMB:=TR_signal;
TR_Out:=Word_TO_Byte(TRoMB);
```

Рис. 9.9. Программа перекладки данных

Упражнение 2. *Связь с верхним уровнем*

22. Запустите сервер ввода/вывода. Откройте конфигуратор сервера ввода/вывода Infinity.
23. Добавьте модуль “Модуль ModBus TCP/IP Master”.
24. Выделите добавленный модуль и нажмите на кнопку “Конфигурация модуля…”.
25. В появившемся окне во вкладке “Общие параметры” укажите 0 номер логического канала.
26. Перейдите во вкладку “Маршруты” (Рис. 9.9) и нажмите кнопку “Добавить”.

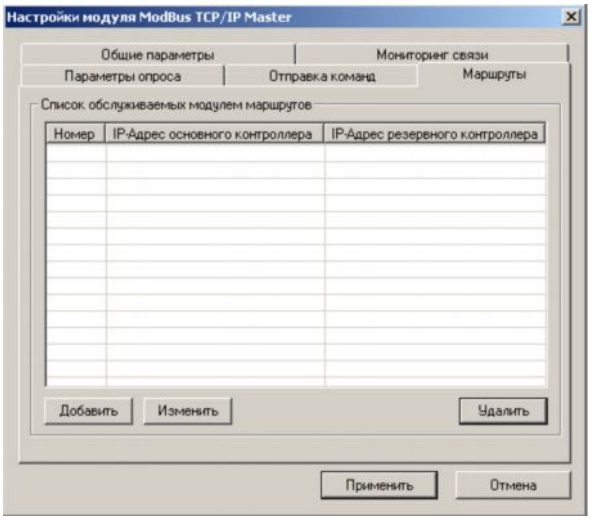


Рис. 9.9 Окно редактирования маршрутов

27. В открывшемся окне (Рис. 9.10) в качестве номера маршрута укажите 1-й маршрут, введите IP-адрес контроллера, с которым Вы работаете. Укажите номер подчиненной станции основного контроллера – 1. Также здесь возможно задействовать резервный контроллер.

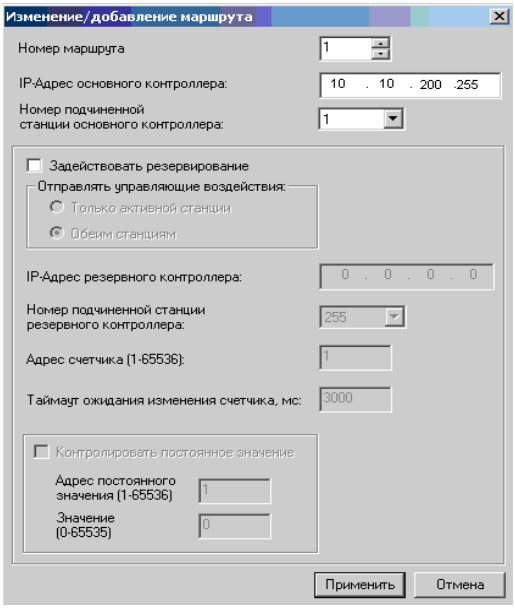


Рис. 9.10 Добавление маршрута

28. В конфигураторе сервера ввода/вывода создайте новый сигнал в формате Boolean. Для этого нажмите на папке “Дерево сигналов” правой клавишей мыши и из появившегося меню выберите “Boolean”. Назовите сигнал “TS_signal”. Откройте вкладку “Редактор адреса”, расположенную с правой части окна конфигуратора.
29. В поле “Доступные протоколы” выберите “ModBus”. В поле “Устройство” выберите “Канал ModBus 0”. В качестве протокольного типа укажите “Телесигнализация”. Номер маршрута – 1. Тип сегмента – Телесигнализация ТС-1х.
30. Аналогичным образом добавьте еще два сигнала представленных в таблице сигналов

Таблица сигналов

Имя сигнала	Тип данных	Протокольный тип	Адрес сигнала	Номер маршрута	Тип сегмента
TU_signal	Boolean	Телеуправление	23	1	Телеуправление ТУ-0х
TI_signal	Word(B2)	Телеизмерение ТИ2	21	1	Телеизмерения ТИ-3х
TR_signal	Word(B2)	Телерегулирование ТР2	20	1	Телерегулирование ТР-4х

31. После добавления сигналов сохраните созданную конфигурацию. Это делается в меню “Файл -> Сохранить конфигурацию”.
32. Перезапустите все модули в конфигураторе для старта обновления статусов сигналов.
33. Откройте Infinity HMI. Для этого последовательно нажмите “Пуск -> Программы -> InfinitySuite -> Infinity HMI -> Infinity HMI”, либо запустите яр-

лык, находящийся на рабочем столе.Создайте мнемосхему как показано на рис.9.12.

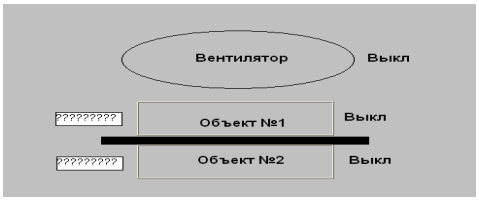


Рис.9.12

34. Настройте объекты “Значение параметра” (кнопка 686, находящаяся в правой нижней части окна редактора) как показано на Рис. 9.113 и Рис. 4.

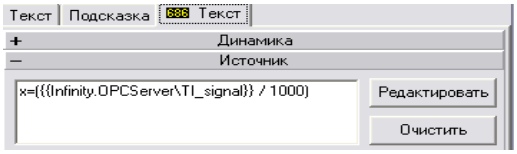


Рис. 9.113 Настройка отображения сигнала телеизмерения

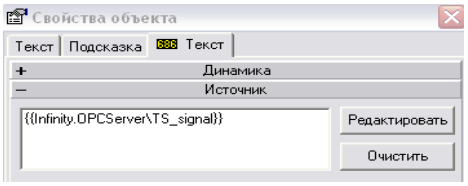


Рис. 9.14 Настройка отображения сигнала телесигнализации

35. Настройте кнопки ВКЛ-ВЫКЛ для объетов 1 и 2: сигналы TU_signal_1 и TU_signal_2 соответственно.

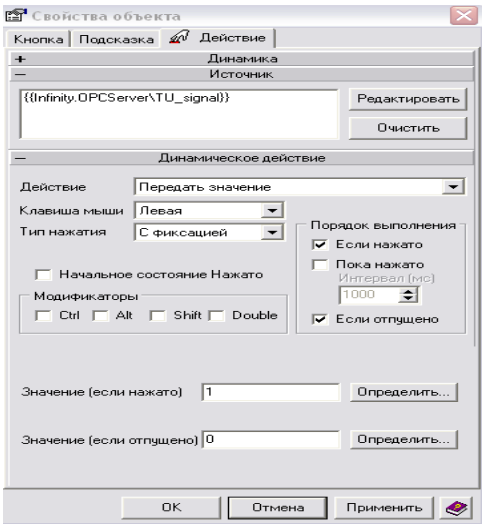


Рис. 9.125 Настройка задания сигнала телеуправления.

36. Добавьте тревоги на объектах, Hi=90, HiHi=120.
37. Запустите созданный проект. Для этого нажмите клавишу F9, либо кнопку “Запустить проект” , расположенную в верхней части окна Infinity HMI. Проверьте работоспособность добавленных сигналов. На этом лабораторная работа окончена.

Контрольные вопросы

1. Каково значение функций в протоколе ModBus?
2. Какой модуль в конфигураторе сервера ввода-вывода отвечает за соединение с сигналами контроллера?
3. Какие сигналы отвечают за отображение характеристик температурного объекта?

Требования по содержанию отчета

В отчете студент должен перечислить цели лабораторной работы, описать ход работы, ответить на контрольные вопросы, сделать вывод о проделанной работе.

ЛАБОРАТОРНЫЙ ПРАКТИКУМ ПО ИНТЕГРИРОВАННЫМ КОМПЬЮТЕРНЫМ СИСТЕМАМ УПРАВЛЕНИЯ

методические указания по выполнению лабораторных работ
по курсам:
проектирование мехатронных систем, интегрированные компьютерные системы проектирования и управления

Составители: Громаков Евгений Иванович
Рудницкий Владислав Александрович

Подписано к печати . Формат 60x84/16. Бумага «Классика».
Печать RISO. Усл.печ.л. . Уч.-изд.л. .
Заказ . Тираж экз.

Томский политехнический университет
Система менеджмента качества
Томского политехнического университета сертифицирована
NATIONAL QUALITY ASSURANCE по стандарту ISO 9001:2000

ИЗДАТЕЛЬСТВО ТПУ. 634050, г. Томск, пр. Ленина, 30.