

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

А.А. Мезенцев, В.М. Павлов

САПР TRACE MODE 6

*Рекомендовано в качестве учебно-методического пособия
Редакционно-издательским советом
Томского политехнического университета*

Издательство
Томского политехнического университета
2012

УДК 004.4'22:681.5(075-057.88)

М44

Мезенцев А.А.

М44 САПР TRACE MODE 6: учебно-методическое пособие / А.А. Мезенцев, В.М. Павлов; Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2012. – 137 с.

Пособие посвящено вопросам проектирования прикладного программного обеспечения, используемого в автоматизированных и автоматических системах управления, с применением средств автоматизированной разработки CASE. В работе описаны этапы освоения пакета инструментальных программ TRACE MODE 6, а также предложены практические задания, позволяющие закрепить полученные знания. Работа содержит 9 глав, 137 страниц, 3 приложения, 97 иллюстраций и 6 таблиц, 9 источников литературы.

Предназначено для студентов обучающихся по специальности 140801 «Электроника и автоматика физических установок».

УДК 004.4'22:681.5(075-057.88)

Рецензенты

Доктор технических наук, профессор ТГУ
В.Т. Калайда

Кандидат физико-математических наук
научный сотрудник Института сильноточной электроники СО РАН
С.А. Попов

© ФГБОУ ВПО НИ ТПУ, 2012
© Мезенцев А.А., Павлов В.М., 2012
© Оформление. Издательство Томского
политехнического университета, 2012

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
1. ПАКЕТ ИНСТРУМЕНТАЛЬНЫХ ПРОГРАММ TRACE MODE	7
1.1. Основной состав продуктов TRACE MODE	7
1.2. Структура и принципы взаимодействия элементов TRACE MODE.....	8
1.3. Структура и состав элементов Проекта.....	9
1.4. Канал, его структура и назначение.....	16
1.5. Вопросы для самоконтроля к главе №1	18
2. ОСОБЕННОСТИ ПРОЕКТИРОВАНИЯ ПРИКЛАДНОГО ПО В TRACE MODE 6	20
2.1. Общее описание графического интерфейса пользователя ИС TRACE MODE 6.....	20
2.2. Стратегии разработки Проекта в зависимости от структуры и сложности АСУ	22
2.3. Принципы работы в интегрированной среде разработки Про- ектов TRACE MODE 6.....	26
2.4. Переменные в TRACE MODE 6.....	28
2.5. Редактор-навигатор переменных (аргументов).....	30
2.6. Подготовка Проекта к загрузке в контроллер, запуск МРВ на исполнение.....	33
2.7. Вопросы для самоконтроля к главе №2.....	34
3. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ОБРАБОТКИ ДАННЫХ В TRACE MODE 6.....	36
3.1. Редактор программ.....	36
3.2. Язык программирования «Техно FBD».....	38
3.3. Язык программирования «Техно IL».....	40
3.4. Язык программирования «Техно ST».....	43
3.5. Язык программирования «Техно LD».....	45
3.6. Создание пользовательских функциональных блоков.....	46
3.7. Вопросы для самоконтроля к главе №3.....	47
4. РАЗРАБОТКА ПРОЕКТА ПРИКЛАДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В TRACE MODE 6.....	49

4.1. Правила чтения функциональных схем автоматизации.....	50
4.2. Алгоритм проектирования прикладного ПО.....	52
5. УПРАВЛЕНИЕ ПРАВАМИ ДОСТУПА ПЕРСОНАЛА, ПАРОЛЛИРОВАНИЕ.....	54
5.1. Вопросы для самоконтроля к главе №5.....	58
6. АРХИВИРОВАНИЕ ТЕХНОЛОГИЧЕСКОЙ ИНФОРМАЦИИ..	59
6.1. Использование промышленных архивов данных в TRACE MODE 6.....	59
6.2. Использование системных переменных для управления дан- ными в архивах.....	64
6.3. Вопросы для самоконтроля к главе №6.....	66
7. ЗАПУСК НА ВЫПОЛНЕНИЕ ГОТОВЫХ ПРОЕКТОВ.....	67
8. ВЗАИМОДЕЙСТВИЕ С УСО В ОС MS WINDOWS И DOS.....	68
8.1. Принципы разработки и структура Проекта создаваемого в TRACE MODE 6 для микропроцессорного контроллера с пла- тами УСО.....	69
8.2. Настройка драйвера в составе Проекта.....	71
8.3. УСО в распределённых АСУ.....	77
8.4. Элементы индикации и управления, используемые в составе интерфейса пользователя при работе с УСО.....	80
8.5. Описание лабораторного стенда.....	88
8.6. Вопросы для самоконтроля к главе №8.....	91
9. МЕЖПРОГРАММНОЕ ВЗАИМОДЕЙСТВИЕ В ОС MS WINDOWS С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМА DDE, ВЗАИМОДЕЙСТВИЕ С РЕЛЯЦИОННОЙ БАЗОЙ ДАННЫХ ПОД УПРАВЛЕНИЕМ СУБД MS ACCESS.....	92
9.1. Data dynamic exchange или Механизм динамического обмена данными между приложениями ОС MS Windows.....	92
9.2. Взаимодействие TRACE MODE 6 с СУБД.....	97
9.3. Вопросы для самоконтроля к главе №9.....	100
СПИСОК ЛИТЕРАТУРЫ	102
ПРИЛОЖЕНИЕ А. Практические задания к главе №3.....	103
ПРИЛОЖЕНИЕ Б. Практические задания к главе №4.....	108
ПРИЛОЖЕНИЕ В. Практические задания к главе №8.....	130
ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ И ОПРЕДЕЛЕНИЯ.....	133

ВВЕДЕНИЕ

Современные АСУТП – это сложные распределённые в пространстве установки, необходимые для управления технологическим оборудованием на производствах с широким спектром назначения. Одним из основных критериев управления производством является гарантированное качество регулирования и беспбойная работа в течение длительного времени. Количественные критерии оценки качества работы АСУ варьируются в зависимости от важности или опасности производства. Они выше в атомной, химической отрасли и на экспериментальных установках, где каждый результат эксперимента может стоить десятки и сотни тысяч долларов или процесс производства опасен для человека и окружающей среды. Они ниже в отрасли народного хозяйства или производства продуктов бытового назначения.

При проектировании современных АСУТП используются специализированные программные комплексы различного назначения, например, ePLAN, позволяющие автоматизировать данный процесс и с гарантированным качеством конечного продукта (сметной спецификации или электрической схемы подключений) проводить разработку в короткие сроки. При проектировании уникальных элементов АСУ, как правило, пользуются более специальными программными пакетами, такими как P-CAD, AutoCAD или MS Visio.

Для разработки прикладного программного обеспечения АСУ в настоящее время широко используют САПР (также применяется понятие SCADA-система). Программные продукты данного класса позволяют автоматизировать процесс программирования алгоритмов сбора, предварительной и постобработки данных, а также алгоритмов управления и регулирования с использованием классических законов регулирования. САПР на сегодняшний день становятся универсальными средствами разработки программного обеспечения в области АСУТП и позволяют использовать широкий спектр механизмов управления данными. На рисунке 1.1 представлена схема взаимодействия САПР с элементами систем управления.

В мире существует множество различных САПР-пакетов. Многие являются коммерческим продуктом, некоторые свободно-распространяемы. Существует множество проектов, которые поддерживает некоторая компания и использует данную САПР исключительно для собственных нужд.

В России наиболее перспективной САПР прикладного ПО АСУ является TRACE MODE¹ (продукт компании «AdAstrA Reserch Group»). TRACE MODE 6 – последняя наиболее совершенная серия продуктов данной компании, позволяющая разрабатывать программное обеспечение как для сложных распределённых в пространстве, так и для малых, локальных АСУ.

В рамках этого учебного пособия будут рассмотрены основные характеристики и возможности данной SCADA-системы – TRACE MODE 6.06.2 (далее по тексту ТМ).

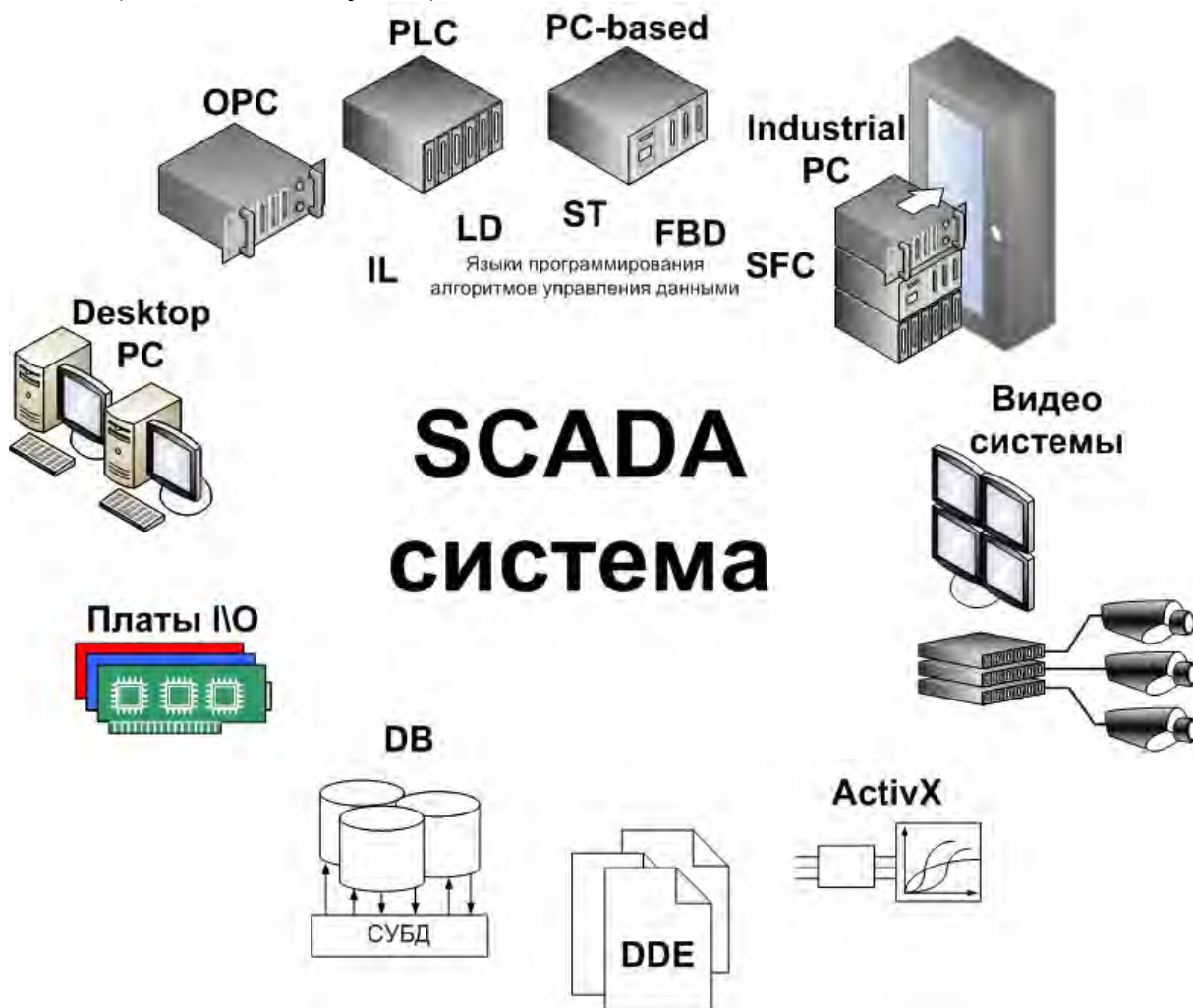


Рисунок 1.1. Элементы АСУ с которыми взаимодействует и в которых работает ТМ

¹ Субъективное мнение автора.

1. ПАКЕТ ИНСТРУМЕНТАЛЬНЫХ ПРОГРАММ TRACE MODE

1.1. Основной состав продуктов TRACE MODE

Компания AdAstrA выпускает несколько основных типов продуктов, это:

- 1) TRACE MODE IDE professional – интегрированная среда разработки прикладных программ, используемых в АСУТП, профессиональная, с физическим ключом защиты.
- 2) TRACE MODE IDE based – интегрированная среда разработки прикладных программ, используемых в АСУТП, учебная, без защиты. Ограничено время исполнения Проекта – 1 ч.
- 3) TRACE MODE RTM – специализированная программа-интерпретатор проектов созданных в ИС, предназначенная для ЭВМ, работающих под управлением ОС Windows. Выполняет роль интерпретатора алгоритмов и программ управления данными, зашифрованных в Проекте, выполняет обращения, посредством драйверов, к УСО, сетевым и периферийным устройствам контроллеров и рабочих станций.
- 4) TRACE MODE MicroRTM – специализированная программа-интерпретатор проектов созданных в ИС, предназначенная для ЭВМ, работающих под управлением ОС DOS и Windows CE. Выполняет роль интерпретатора алгоритмов и программ управления данными, зашифрованных в Проекте, выполняет обращения, посредством драйверов, к УСО, сетевым и периферийным устройствам контроллеров и рабочих станций.

Каждый из перечисленных программных продуктов используется для специальных целей: ИС – для разработки Проекта, МРВ – для чтения сценариев и алгоритмов управления данными в АСУТП из Проектов и исполнения команд в рамках ОС MS Windows. МикроМРВ – выполняет функции аналогичные МРВ, но для DOS или ОС MS Windows CE. Стоит уточнить, что интерпретация Проектов в ИС с использованием отладочного МРВ (интегрированного в ИС – «Профайлер») проводится только для тестирования и отладки программного обеспечения, заказчику поставляются файлы Проекта и МРВ, поскольку поставлять заказчику ИС нецелесообразно. Проекты, созданные в профессиональной версии программы ИС, не могут редактироваться в базовой версии

и наоборот. Проекты, предназначенные для интерпретации в МРВ или МикроМРВ должны создаваться в профессиональной версии ТМ.

Рассмотрим структуру и принципы взаимодействия продуктов ТМ.

1.2. Структура и принципы взаимодействия элементов TRACE MODE

TRACE MODE IDE 6.06.2 – это специализированная программа, используемая для автоматизированного проектирования прикладного программного обеспечения АСУ, которое может применяться в ней при выполнении функций супервизорного контроля значений параметров и управления технологическим оборудованием. Результатом использования данного пакета является прикладная программа «Проект». Проектом, соответственно, называется совокупность интерпретируемых файлов, в которых запрограммированы алгоритмы управления данными в некотором узле АСУ (в котором будут установлены файлы Проекта и МРВ). Стоит отметить, что узлы системы управления могут работать автономно, без использования элементов Проекта. Таким образом, не каждый узел АСУ можно назвать Узлом Проекта. Рассмотрим элементы программного обеспечения узла АСУ на примере рисунка 1.2.

В соответствии со спецификацией представленной ранее, разработка Проекта ведётся с использованием профессиональной (professional) или базовой, учебной (base) версии ИС. Далее Проект интерпретируется МРВ или МикроМРВ в зависимости от операционной системы. Для интерпретации Проектов, созданных в базовой версии ИС, используется Профайлер. МРВ содержит инструменты взаимодействия с УСО и, следовательно, с технологическим оборудованием. Посредством этих инструментов МРВ выполняет технологические операции, запрограммированные в Проекте.

Среда разработки, в свою очередь, содержит инструменты разработки Проекта, его математической, графической, алгоритмической и информационной составляющих. Для выполнения различных операций используются специальные редакторы в составе ИС (рисунок 1.2).

При более глубоком исследовании программного пакета ТМ следует описать его основные определения и элементы Проекта.

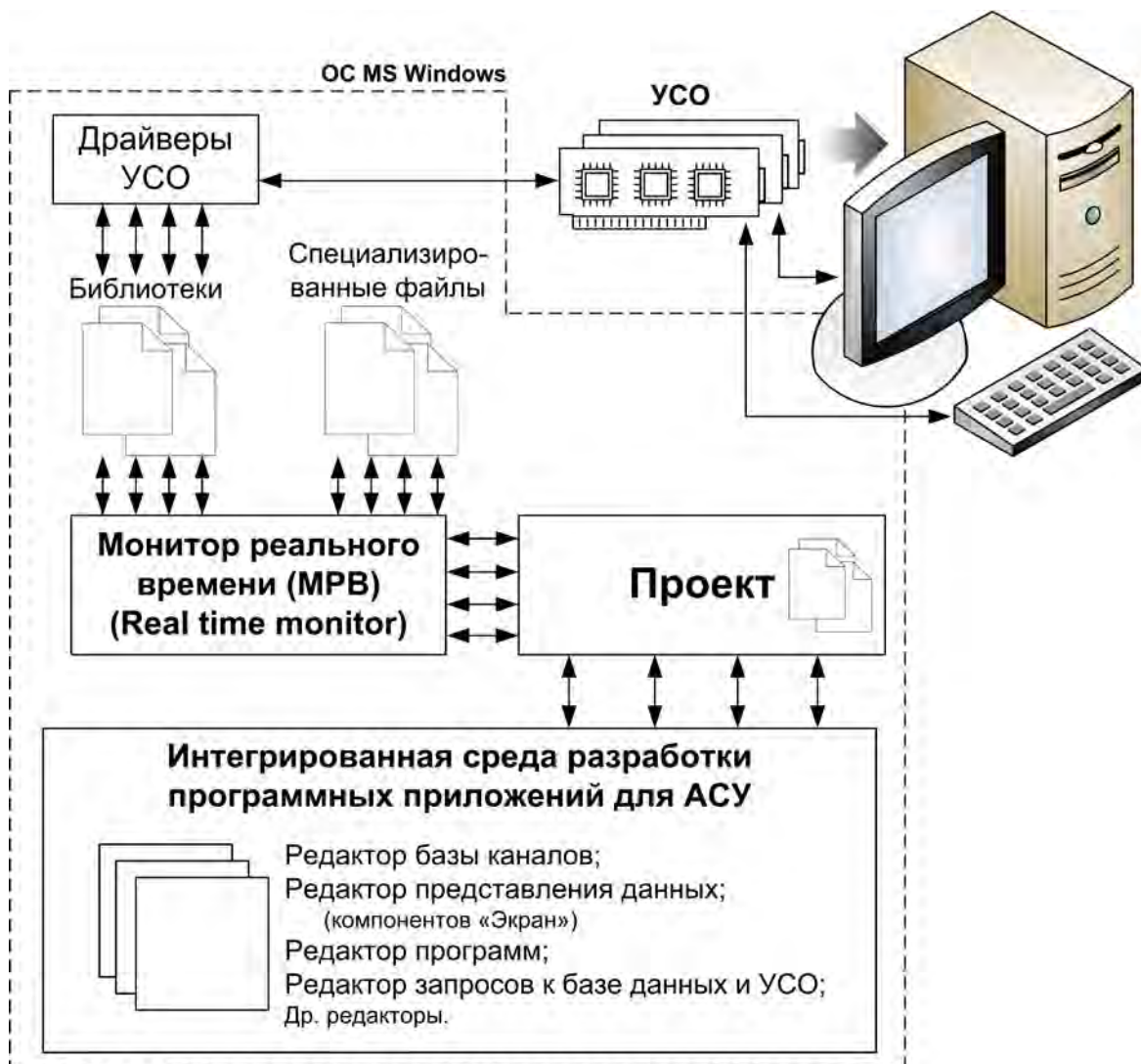


Рисунок 1.2. Структурная схема программного обеспечения узла АСУ

1.3. Структура и состав элементов Проекта

Рассмотрим структуру программного обеспечения узла АСУ в общем виде.

Пусть мы имеем некоторую АСУ, структура которой представлена на рисунке 1.3. Она состоит из верхнего и нижнего уровня системы автоматизации. На верхнем уровне расположены пульта операторов, в состав которых входят ПЭВМ, дисплеи и периферийные элементы ввода/вывода данных. На нижнем уровне расположены промышленные электротехнические шкафы с различной степенью защиты, в которые встраиваются системы управления электропитанием, различные преобразователи, контроллеры и промышленные ЭВМ. Взаимодействие между уровнями АСУ может осуществляться с помощью информационных

сетей различного типа, например Ethernet, RS-485 или коммутируемые линии связи.

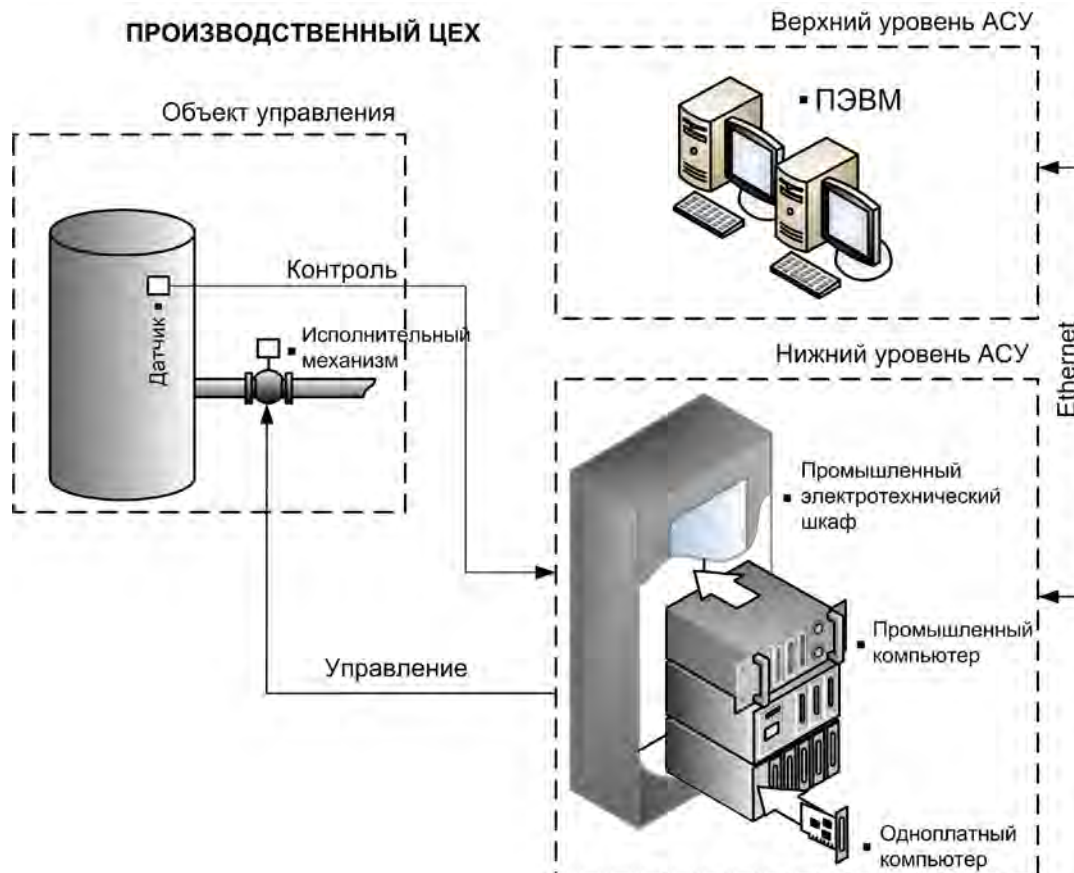


Рисунок 1.3. Состав и структура АСУ производством

Объект или группа объектов, тем или иным способом выделенные из АСУ (например, функционально) называются узлом этой системы. Если в составе узла АСУ запускается часть разработанного Проекта, то эту часть тоже называют Узлом (частью Проекта). Узлы являются самым крупным элементом структуры основного слоя Проекта – «Система» (рисунок 1.4). Система – это совокупность программного обеспечения АСУ разработанного в ТМ, а также слой Проекта (основополагающий Узел в структуре Проекта).

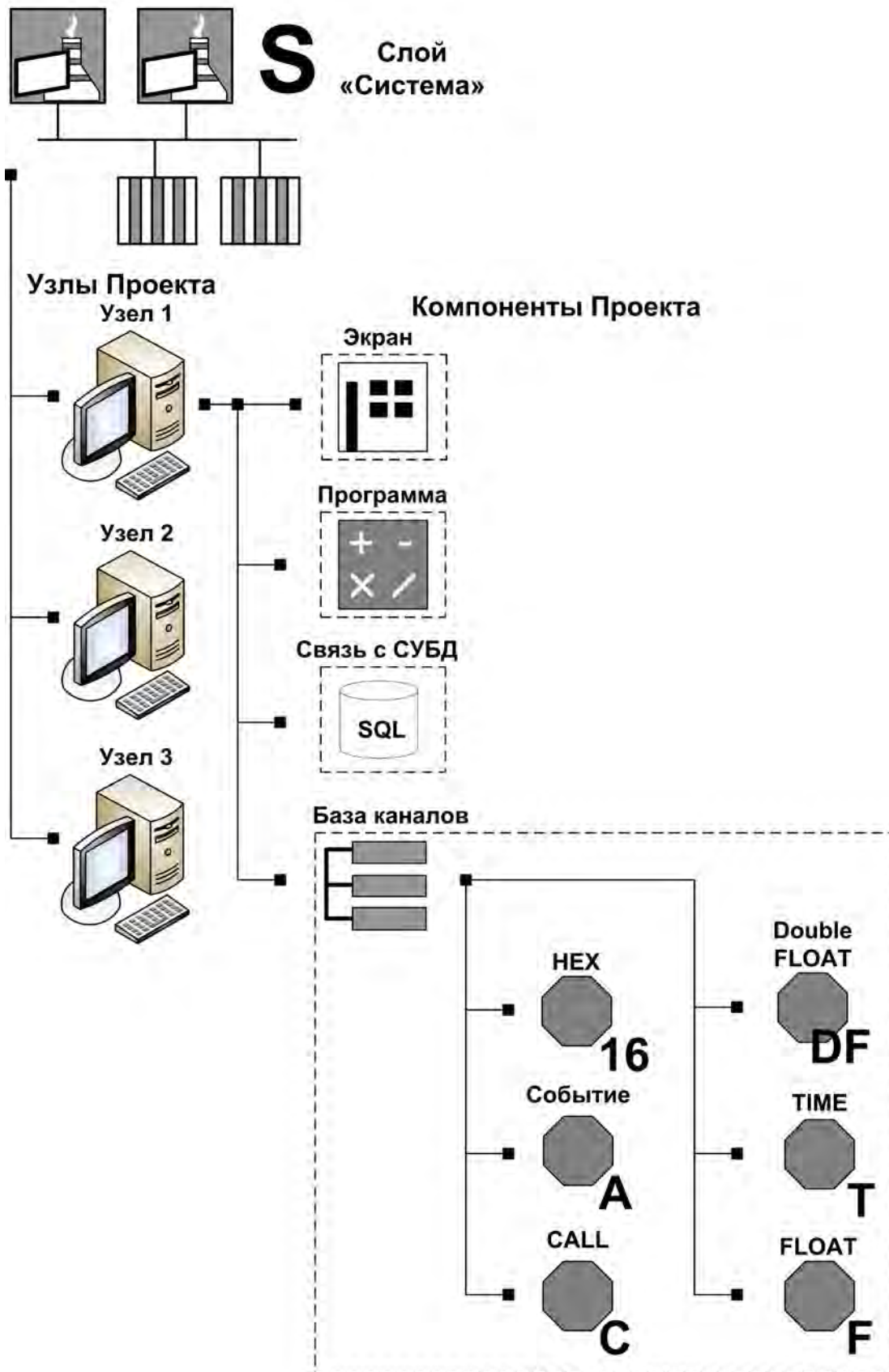


Рисунок 1.4. Структура и состав элементов Проекта

Как правило, структура слоя «Система» с достаточной точностью повторяет структуру АСУ, поскольку для каждого вычислительного узла АСУ разрабатывается часть Проекта, т.е. Узел. По окончании разработки Проекта, выполняется процедура «Сохранить для МРВ» для каждого из Узлов (конвертирование файла <name.prj> в группу файлов с основным файлом <name>.dbb). Полученные после выполнения процедуры файлы Узла Проекта загружаются в соответствующий узел АСУ (например, в контроллер или рабочую станцию оператора) и интерпретируются МРВ.

В составе каждого Узла Проекта можно выделить множество компонентов, которые используются для описания алгоритмов управления данными и информацией. Рассмотрим их более детально.

Канал – это базовое понятие ТМ, которое включает в свой состав совокупность математических алгоритмов, функций и переменных, необходимых для передачи и предварительной обработки технологических данных. Дополнительные функции глубокой постобработки данных программируются в компонентах «Программа», а затем включаются в состав измерительного канала или канала управления как функция «Трансляция».

База каналов – совокупность каналов Проекта объединённых по некоторому признаку. Для группировки нескольких каналов имеющих общие характеристики или назначение в базе каналов могут быть организованы подгруппы (например, относящиеся к одному УСО).

Экран – компонент Проекта, необходимый для отображения информации в АСУ. Инструментальная среда разработки ТМ предполагает использовать в составе Экрана мнемосхему и элементы графического интерфейса оператора. Более детально процесс разработки Экранов будет рассмотрен в следующих главах.

Программа – компонент Проекта, необходимый для управления и специальной обработки данных в АСУ. Программирование алгоритмов обработки данных предполагается проводить с использованием языков программирования стандарта ИЕС(МЭК) 61131-3. Программы, как правило, используются в тех случаях, когда функций предварительной обработки данных доступных в настройках каналов не достаточно или при необхо-

димости программирования сложных законов автоматического управления.

Связь с СУБД – компонент Проекта, необходимый для взаимодействия с СУБД. При использовании данного компонента инструментальная среда разработки позволяет запрограммировать обращение к СУБД на основе SQL-запросов. Данный механизм, более детально, будет представлен далее.

Документ – компонент Проекта, необходимый для создания документов с функцией автоматического чтения значений каналов Проекта в выделенных ячейках. Данный компонент широко используется при формировании отчётных материалов.

Пользователь – компонент Проекта, необходимый для учёта количества, типов и характеристик пользователей, имеющих доступ к управлению производством в АСУ, а также для назначения прав доступа к управлению производством.

Единица оборудования – компонент Проекта, необходимый для учёта количества и характеристик технологического оборудования используемого в АСУ. В ТМ различные характеристики компонента Проекта «Единица оборудования» позволяют отдельным службам предприятия автоматически рассчитывать выработанный технологическим оборудованием ресурс, количество дней простоя по причине отсутствия нагрузки или по причине выхода из строя.

Стоит отметить, что каналы Проекта в зависимости от назначения и типа передаваемых данных делятся на несколько типов: FLOAT, HEX(8, 16), DoubleFLOAT, Событие, CALL, TIME. Более детально каналы будут рассмотрены ниже.

Для практической демонстрации назначения рассмотренных компонентов Проекта выделим из рисунка 1.3 узлы и рассмотрим их более подробно. Структура элементов Проекта представлена на рисунке 1.5.

Пусть в составе электротехнического шкафа имеется промышленный компьютер, в котором на пассивную объединительную плату установлены: одноплатная ЭВМ и платы ввода\вывода сигналов. Пусть на одноплатной ЭВМ установлена дисковая операционная система, МикроМРВ и файлы Проекта, сохранённые для МикроМРВ из среды разработки. В рамках Проекта выделенная промышленная ЭВМ будет являться Узлом. Назовём её «Узел №1».

В составе Проекта запрограммированы: совокупность математических алгоритмов обработки данных (рисунок 1.5(3)), настройки УСО (рисунок 1.5(1)), переменные и каналы связи между ними (рисунок 1.5(2)). Каналы, как было описано ранее, выполняют роль среды передачи информации, а также предоставляют функции обработки данных. Взаимодействие ЭВМ с верхним уровнем АСУ выполняется посредством сетевого пакетного драйвера и сетевой платы. Таким образом, на нижнем уровне АСУ выполняется сбор и предварительная обработка данных, которые затем передаются на верхний уровень АСУ. Визуализация информации может быть выполнена как на нижнем, так и на верхнем уровне автоматизации, однако на нижнем уровне она ограничена количеством функций отображения. Возможно отображение только текстовых данных (используется DOS).

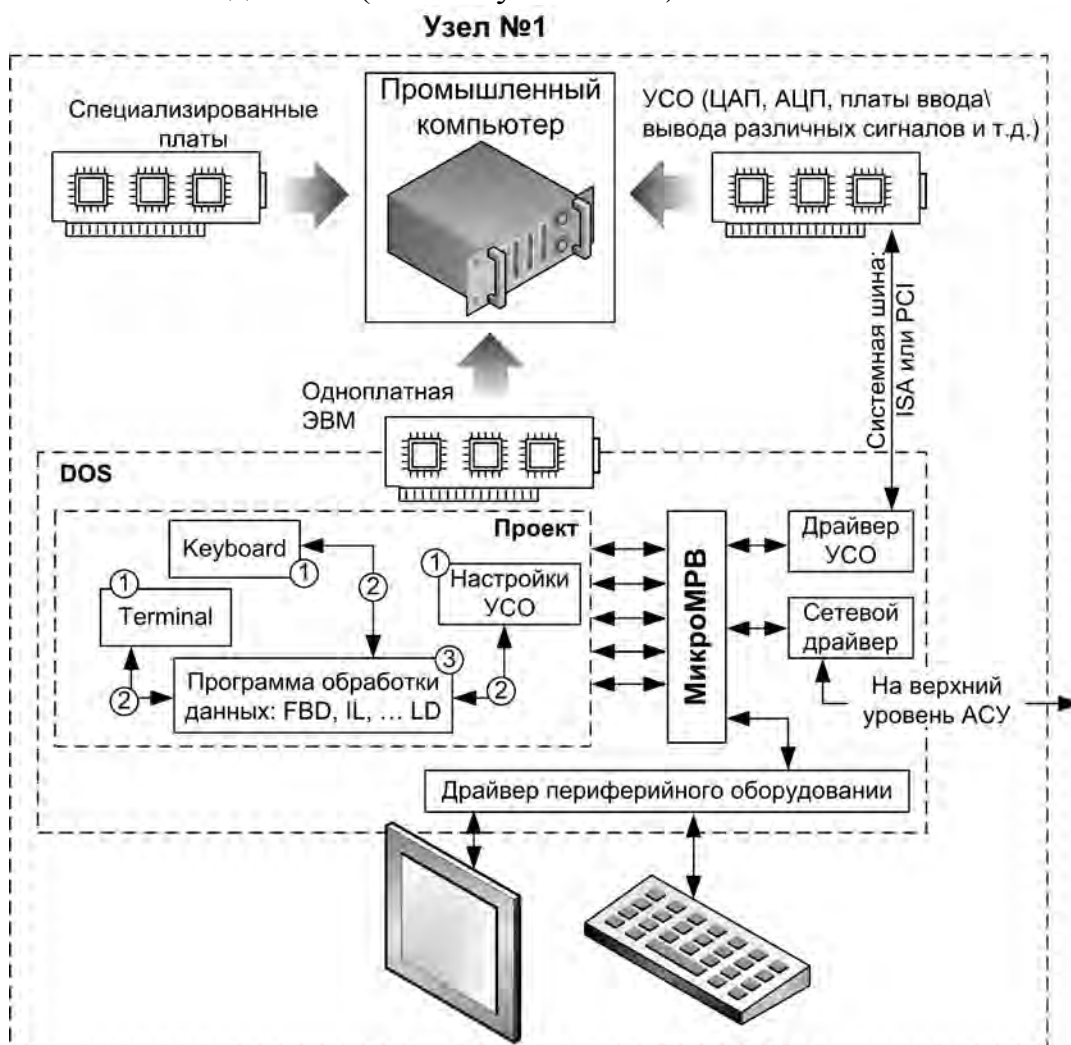


Рисунок 1.5. Структура элементов Проекта в рамках DOS: 1 – настройки устройств используемых в ЭВМ, с которыми работает Проект посредством драйверов; 2 – каналы связи элементов Проекта; 3 – компонент «Программа»

Далее рассмотрим структуру элементов Проекта в рамках ОС MS Windows (рисунок 1.6). Пусть эта часть Проекта называется «Узел №2».

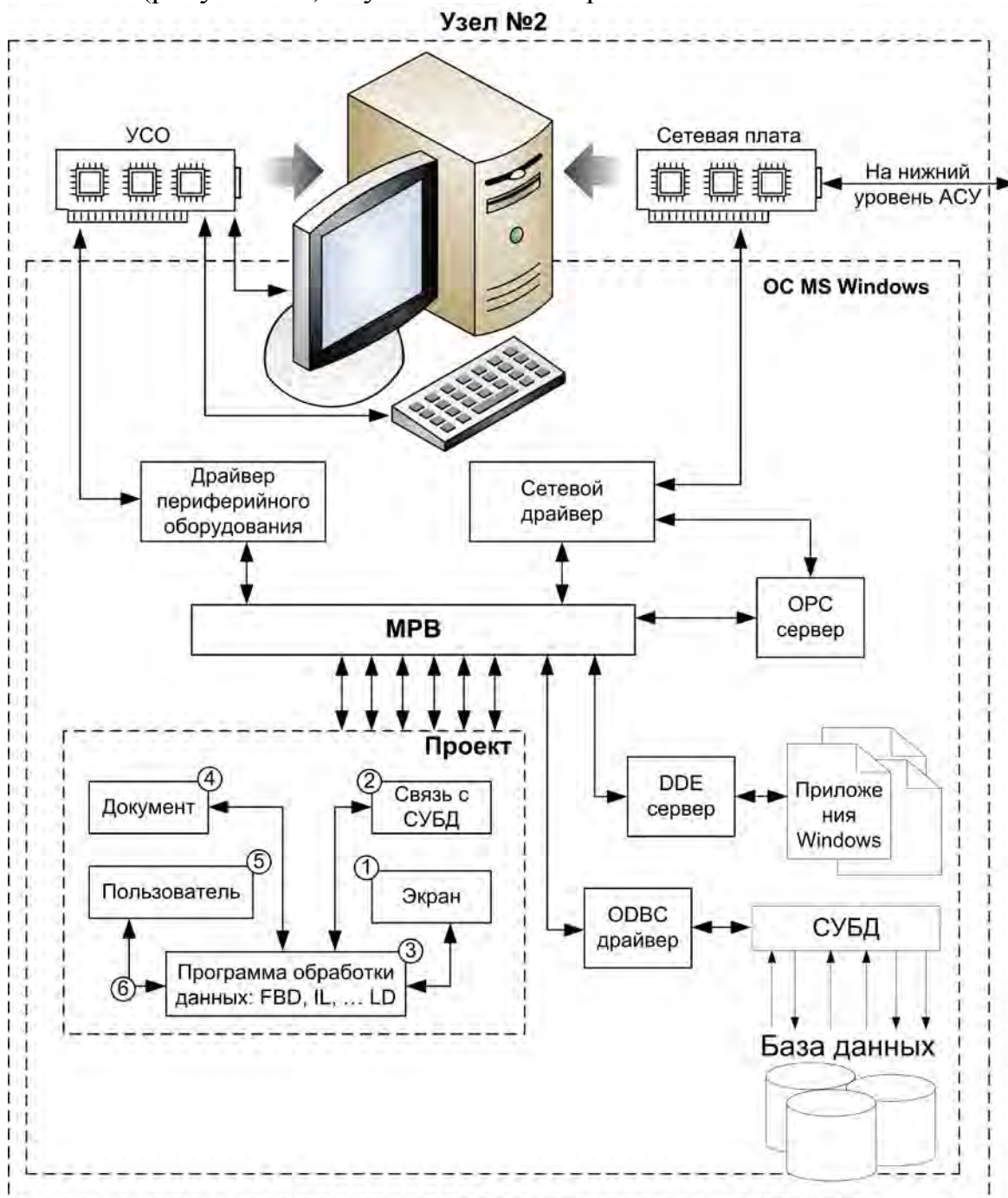


Рисунок 1.6. Структура и состав элементов части Проекта загружаемой на верхнем уровне АСУ: 1 – компонент «Экран»; 2 – компонент «Связь с СУБД»; 3 – компонент «Программа»; 4 – компонент «Документ»; 5 – компонент «Пользователь» или «Оборудование»; 6 – каналы

Сравнивая рисунок 1.5 и рисунок 1.6, можно отметить, что набор функций в части Проекта для ОС MS Windows значительно шире. На

практике этот факт оправдан, поскольку вычислительные возможности программной и аппаратной части нижнего уровня АСУ, как правило, значительно ниже, чем у аналогичных элементов верхнего уровня. При необходимости использования на нижнем уровне АСУ более мощных алгоритмов обработки или визуализации данных и информации используют ПЭВМ с большими значениями характеристик и ОС MS Windows (или другие ОС при разработке Проектов в других SCADA-системах).

1.4. Канал, его структура и назначение

Ранее было сказано о том, что канал в составе Проекта представляет собой среду передачи и преобразования (обработки) технологических данных. Канал, таким образом, позволяет взаимодействовать различным элементам Проекта, например, компоненту «Программа» и «Экран», драйверу УСО и компоненту «Программа». Однако в версии ТМ 6 каналы применяются только для ввода/вывода данных, а межкомпонентная передача данных выполняется через переменные. Исследование канала как объекта необходимо начинать с деления каналов на типы по некоторым признакам:

- по направлению движения информации в канале: INPUT, OUTPUT;
- по типу передаваемой информации: HEX16, HEX32, FLOAT, DOUBLE FLOAT, TIME, Событие, Пользователь, Единица оборудования, Персонал, М-ресурс, D-ресурс, CALL.

В зависимости от направления передачи данных каналы используются для передачи данных от УСО и ввода информации в ЭВМ (INPUT) и наоборот (OUTPUT).

В соответствии с описанными типами данных передаваемых в канале, они используются:

Группа 1. Каналы, используемые для работы с данными

- HEX16 – для работы с 2-байтовыми целыми числами;
- HEX32 – для работы с 4-байтовыми целыми числами;
- FLOAT – для работы с 4-байтовыми вещественными числами (существуют две разновидности канала этого класса – с обработкой и без обработки в канале);
- DOUBLE FLOAT – для работы с 8-байтовыми вещественными числами;
- TIME – для работы со значениями времени (дата и время);

Группа 2. Каналы, используемые для мониторинга

- Событие – для мониторинга объекта с целью фиксирования возникновения/исчезновения на этом объекте некоторого события или ситуации (например, аварии). Канал хранит историю события и допускает его квитирование;

Группа 3. Каналы, регламентирующие права пользователей

- Пользователь – для задания прав пользователя на разработку и/или интерпретацию Проекта;

Группа 4. Каналы класса T-FACTORY

- Единица оборудования – для учета характеристик единицы оборудования;
- Персонал – для учета характеристик работника, а также планирования и мониторинга его участия в техобслуживании оборудования;
- М-ресурс – для учета складских ресурсов;
- D-ресурс – для планирование и мониторинга процесса техобслуживания оборудования и ряда других задач;

Группа 5. Каналы многофункционального назначения

- CALL – для вызова Шаблонов* – компонентов Проекта.

Группа каналов 1, которые называют числовыми каналами, является основной, и используются для обработки информации. Рассмотрим структуру канала класса FLOAT на примере рисунка 1.7. Данный тип каналов широко используется для работы с аналоговыми линиями связи и содержит соответствующие процедуры обработки данных.

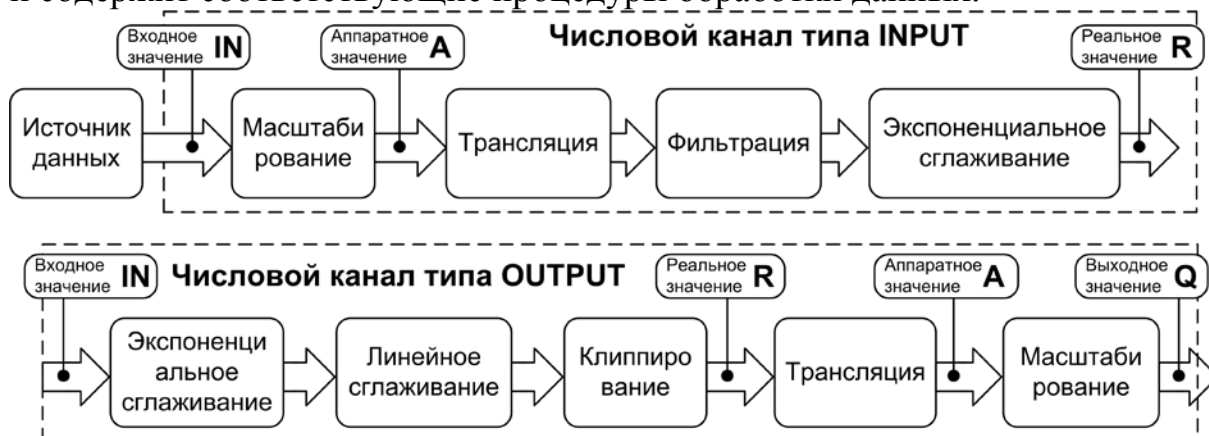


Рисунок 1.7. Структура канала класса FLOAT

* Шаблон – компонент проекта, понятие которого детально будет рассмотрено в главе №2

После выполнения группы операций над данными, значению канала присваивается один из статусов: входное (I), аппаратное (A), реальное (R) или выходное значение (Q). Использование такой конструкции позволяет конфигурировать функции обработки в соответствии с необходимым алгоритмом обработки данных и использовать значения обработанных данных после каждого этапа обработки. Так на мнемосхеме может быть отображено аппаратное или реальное значение канала. Структуру канала класса HEX16 необходимо рассмотреть самостоятельно.

1.5. Вопросы для самоконтроля к главе №1

1. Назначение пакета программ TRACE MODE?
2. Какова структура TRACE MODE 6.06.2 и связи этого пакета с внешними программами и технологическим оборудованием АСУ?
3. Дайте определение понятию Проект (в рамках ТМ), каково его назначение?
4. Дайте определение понятию канал (в рамках ТМ), каково его назначение, функции и характеристики?
5. Дайте определение понятию монитор реального времени, каково его назначение и функции?
6. Изобразите структуру канала класса FLOAT и его состав (в рамках ТМ)?
7. В рамках каких операционных систем может работать TRACE MODE IDE 6.06.2?
8. Какие САПР, кроме TRACE MODE Вы знаете?
9. Какие виды обработки данных предусмотрены в каналах Проекта? Поясните на примере канала класса FLOAT.
10. Назовите основные типы каналов используемых в TRACE MODE?
11. Дайте определение понятию «Система» (в рамках ТМ), какова её структура и состав?
12. Дайте определение понятию Узел (в рамках ТМ)?
13. Назовите основные языки программирования алгоритмов управления данными стандарта МЭК 61131-3?
14. Назовите типы MPB, используемых TRACE MODE 6.06.2?
15. Назовите основное отличие базовой версии TRACE MODE IDE от профессиональной?
16. Возможно ли использование по назначению Проектов TRACE MODE созданных в базовой версии ИС для DOS и почему?

17. Возможно ли редактирование в базовой версии ИС Проектов созданных в профессиональной версии ИС и почему?
18. Каково назначение МикроМРВ, назовите его функции?
19. Назовите отличия в структуре или составе Проектов созданных для МикроМРВ и МРВ? Отличаются ли они?
20. Что такое квитиование?

2. ОСОБЕННОСТИ ПРОЕКТИРОВАНИЯ ПРИКЛАДНОГО ПО В TRACE MODE 6

В первой главе были рассмотрены вопросы, связанные с общими характеристиками САПР TRACE MODE 6.06.2. Описанные элементы и структура интегрированной среды разработки и Проекта показывают её достаточную универсальность. Понятие «Интегрированная среда разработки» подразумевает объединение всех редакторов компонентов Проекта в один пакет программ. Достоинством такого подхода является возможность разработки Проекта, начиная с любой его части (математической, графической части, с настройки драйверов связи с УСО и т.д.). Возможности и принципы разработки Проектов в ТМ будут рассмотрены в данной главе пособия.

2.1. Общее описание графического интерфейса пользователя ИС TRACE MODE 6

Для дальнейшего описания принципов работы с ИС программного обеспечения ТМ необходимо рассмотреть основные графические элементы окна данной программы. Графический интерфейс оператора ИС ТМ представлен на рисунке 2.1.

В составе графического интерфейса пользователя ИС содержится Навигатор проекта, позволяющий перемещаться по его структуре, изменять состав и структуру Узлов Проекта, создавать компоненты Проекта. Также он предназначен для запуска редактора свойств каждого компонента, а затем настройки взаимодействия с драйверами, сетевого взаимодействия между Узлами и т.д. Специальные пиктограммы Навигатора проекта, расположенные в правой части области пиктограмм, позволяют делить область Навигатора проекта на две или более одинаковых части. Это свойство значительно облегчает процесс проектирования в тех случаях, когда необходимо работать одновременно с несколькими частями Проекта. Путь к текущему объекту, свойства которого редактируются, указан в заголовке окна программы и в нижней части окна Навигатора проекта. Следует отметить, что при изменении любых свойств элементов Проекта, название файла Проекта в заголовке окна программы помечается знаком «*».

Внимание, учитывая возможные нестабильности в работе ОС и ИС, рекомендуется периодически выполнять сохранение Проекта на жёсткий диск ПЭВМ.

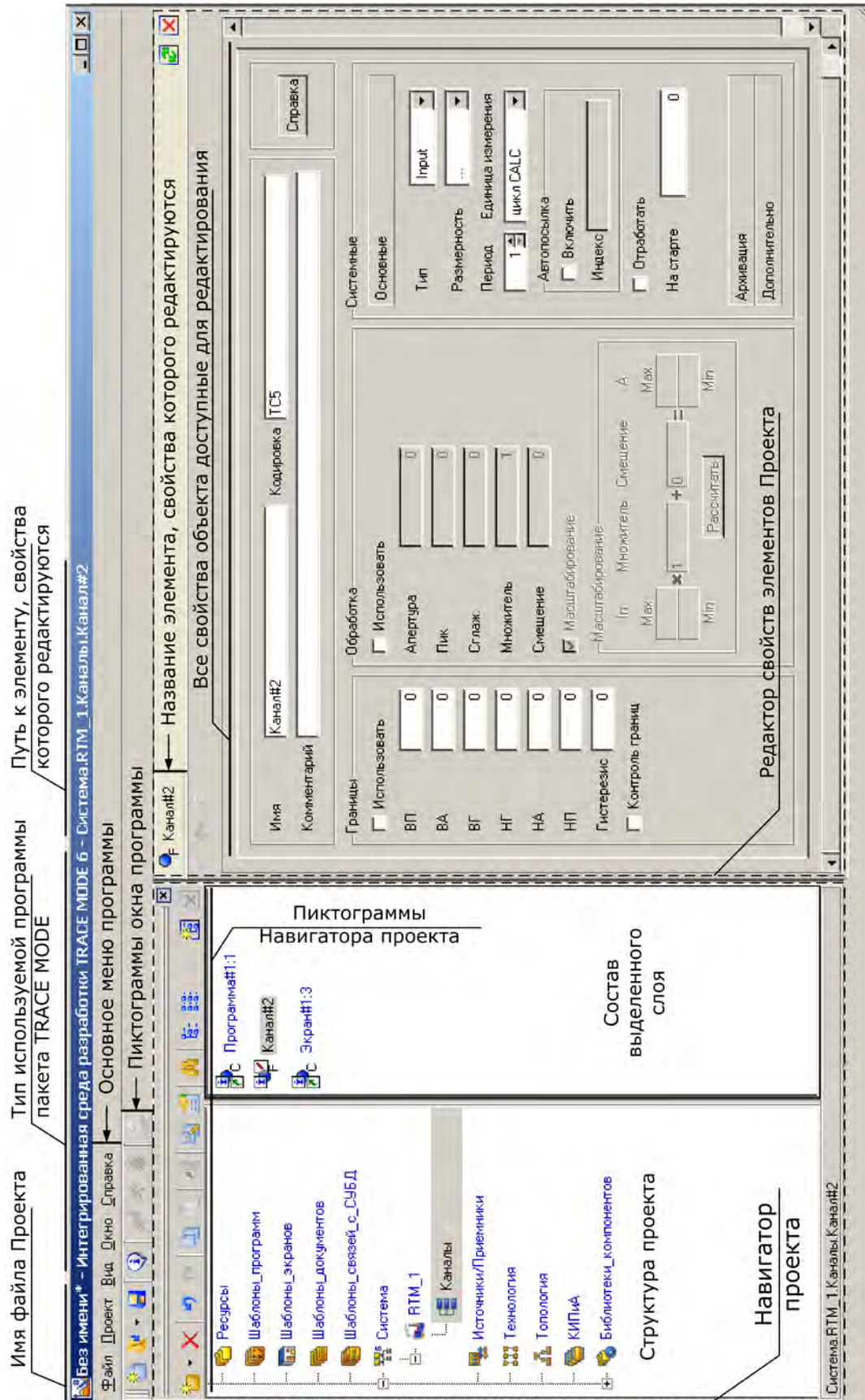


Рисунок 2.1. Вид окна программы ИС ТМ

В правой части окна программы открываются свойства выделенного компонента, это могут быть каналы, Программа, Экран и т.д. Система временного разделения пространства выделенного в окне программы ИС для редактора свойств позволяет наиболее эффективно работать с данным редактором, открывать для работы несколько редакторов одновременно и работать с ними поочередно. В разделе «Системные» редактора свойств (рисунок 2.1), также предусмотрены несколько вкладок, которые группируют свойства объекта по функциональному признаку. Для оперативного доступа к разделу справки – руководству пользователя, необходимо воспользоваться кнопкой «Справка» в окне редактора свойств объекта либо в основном меню программы ИС в заголовке её окна.

2.2. Стратегии разработки Проекта в зависимости от структуры и сложности АСУ

О значительной универсальности интегрированной среды разработки пакета программ ТМ свидетельствует наличие нескольких стратегий разработки Проектов. В зависимости от сложности АСУ, для которой производится разработка прикладного программного обеспечения, среда разработки ТМ может быть настроена индивидуально. При первичной инициализации операции создания Проекта в ИС разработчики ТМ предлагают выбрать один из 3-х уровней первичной настройки среды разработки, однако, в редакторе параметров настройки ИС присутствуют элементы точной настройки необходимых групп компонентов Проекта – «Слоёв».

Разработка Проекта в ИС может выполняться с учётом структуры предприятия (топологии) с её делением на: объект, здание, этаж и помещение. В этом случае учитывается местоположение узлов АСУ в структуре предприятия, внутри зданий и помещений. В дальнейшем структура узлов АСУ переносится в структуру Узлов Проекта. Также разработка Проекта может осуществляться в зависимости от технологии производства с учётом количества и качества агрегатов и элементов производства, объединяемых по функциональному признаку. Далее, как и в первом случае, структура узлов АСУ переносится в структуру Узлов Проекта. Подобные конструкции применяются при проектировании программного обеспечения для АСУ с высокой распределённостью элементов и собственной многоуровневой структурой.

В общем случае при разработке Проектов в качестве головного узла используется слой «Система». Рассмотрим более детально Навигатор проекта в ИС и вкладку с инструментами редактирования состава эле-

ментов Проекта в зависимости от мощности АСУ на примере серии рисунков (2.2, 2.3, 2.4, 2.5).

При первичной инициализации Проекта в ИС, как было сказано ранее, необходимо определить сложность структуры и полноту состава элементов Проекта. Выполнить это действие можно при помощи дополнительного окна программы, представленного на рисунке 2.2. В дальнейшем изменить эту настройку можно в окне программы настройки специальных параметров интегрированной среды разработки Проектов, которая представлена на рисунке 2.3. Следует выделить функцию детальной настройки именно тех слоёв, которые необходимы в конкретном случае. Для этого используется группа инструментов «Видимость слоёв» также представленная на рисунке 2.3. Вкладка Навигатора проекта, с указанными на рисунке 2.3 в разделе «Видимость слоёв» типами компонентов, представлена на рисунках 2.4, 2.5 для простой и комплексной АСУ.

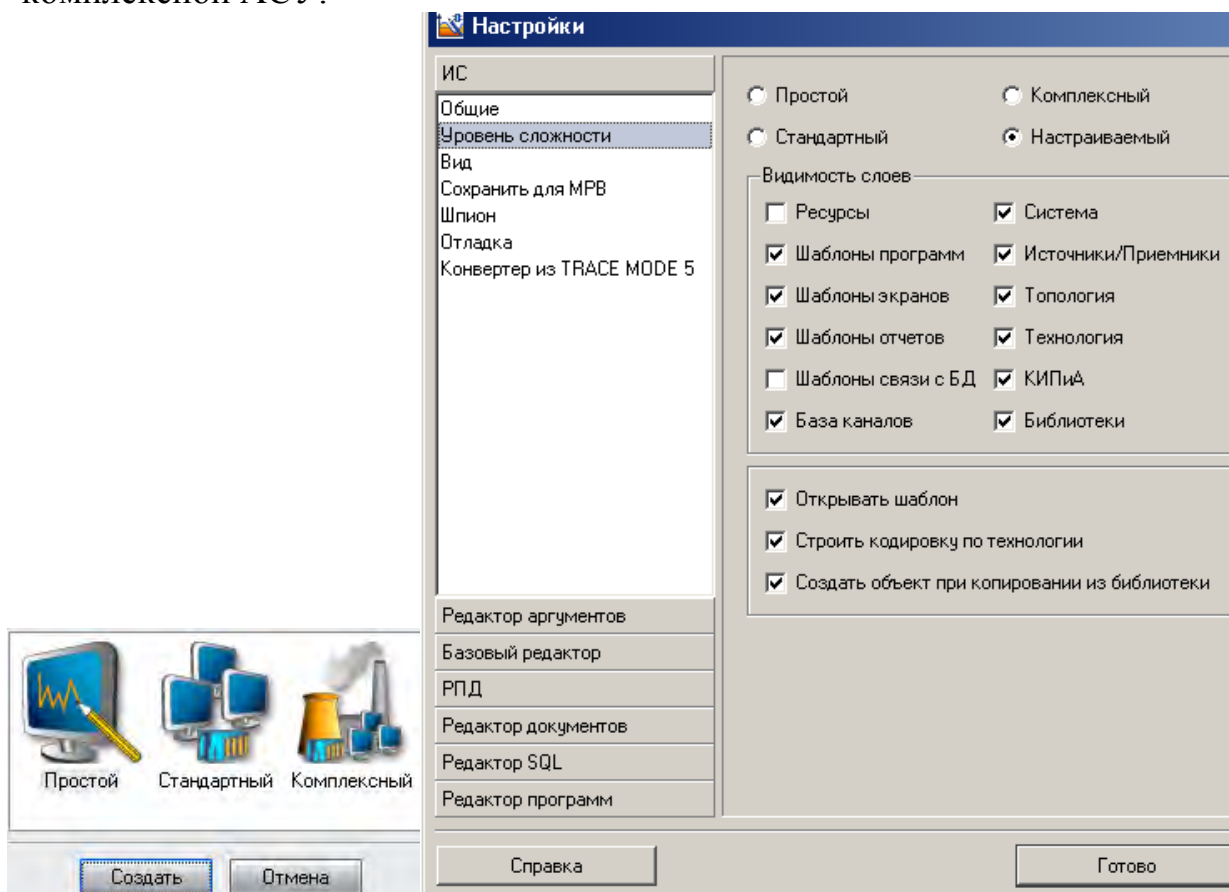


Рисунок 2.2. Интерфейс окна программы первичной инициализации Проекта

Рисунок 2.3. Вид окна программы настройки специальных параметров интегрированной среды разработки ТМ

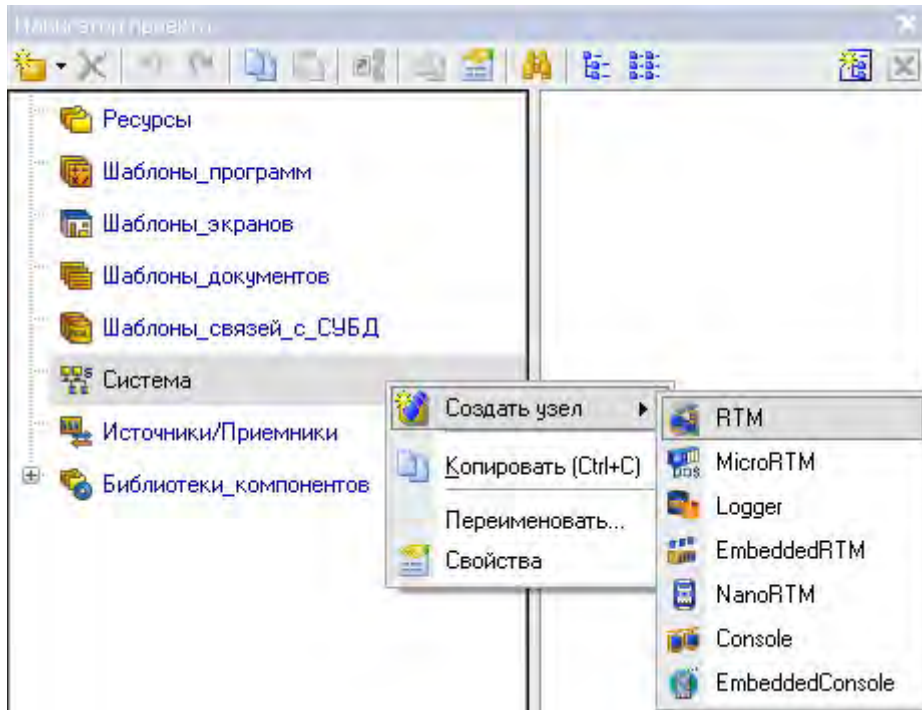


Рисунок 2.4. Вид вкладки «Навигатор проекта» для режима работы ИС «Простой»

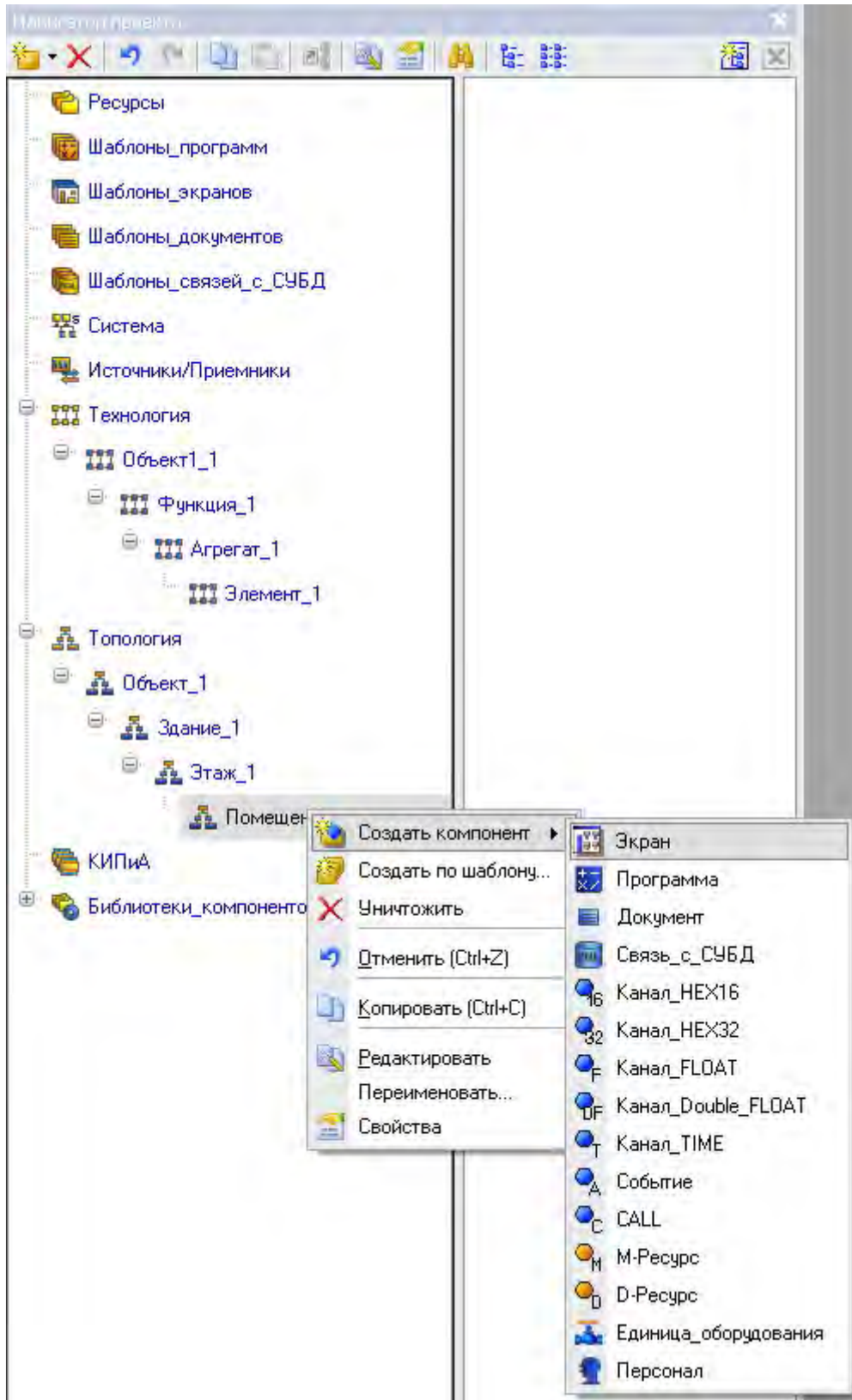


Рисунок 2.5. Вид вкладки «Навигатор проекта» для режима работы ИС «Комплексный»

2.3. Принципы работы в интегрированной среде разработки Проектов TRACE MODE 6

В соответствии с описанной структурой и составом элементов Проекта, а также стратегией проектирования рассмотрим основные правила создания Проекта. Детальная информация о работе с ИС ТМ представлена в руководстве пользователя на данный программный продукт. Пошаговые инструкции по разработке Проекта представлены в разделе «Быстрый старт» того же руководства пользователя.

На первой стадии разработки необходимо определить количество и типы Узлов, которые будут созданы в Проекте. Рассмотрим пример на

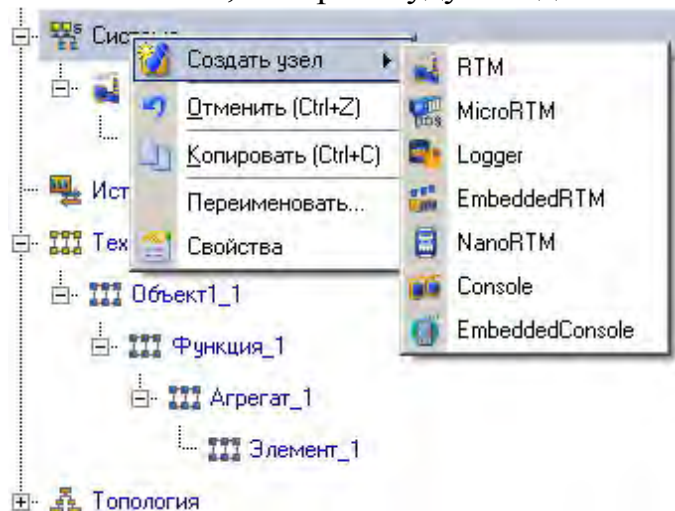


Рисунок 2.6. Вид вкладки «Навигатор проекта», типы Узлов

основе Проекта, в котором существует головной элемент структуры – «Система», поскольку только в слое «Система» могут быть созданы Узлы (режим редактирования «Простой»). Во всплывающем меню данного слоя, представленного на рисунке 2.6, указаны основные действия, которые можно выполнить над ним. Главным действием является «Создать узел».

В зависимости от операционной системы и типа контроллера, в котором будут установлены файлы Проекта, выбирается один из следующих типов Узлов: RTM, MicroRTM, Logger, EmbeddedRTM, NanoRTM, Console, EmbeddedConsole. После создания необходимого количества Узлов необходимо сформировать их структуру и состав. При разработке Проектов в режиме «Комплексный» необходимо перенести созданные компоненты Проекта из слоя «Топология» или «Технология» методом «Drag and Drop» с одновременным удерживанием клавиш «Ctrl» и «Shift». Метод «Drag and Drop» также используется для автопривязки элементов Проекта (канал, «Экран» и т.д.) без фиксации вышеуказанных контрольных клавиш универсальной клавиатуры.

Создадим Узел типа RTM для работы в рамках ОС MS Windows на автоматизированном рабочем месте (АРМ) оператора. Далее необходимо создать компонент «Экран», используемый для формирования гра-

фического интерфейса оператора. Важной особенностью ИС ТМ на этом этапе проектирования ПО является то, что в ней присутствует понятие шаблон. Шаблон – это компонент Проекта, который в дальнейшем вызывается при помощи специального канала класса CALL с необходимыми атрибутами в том или ином Узле Проекта. В составе Проекта могут быть созданы шаблоны Экранов, Программ, электронных документов (рисунок 2.5). В дальнейшем шаблон может быть вызван один или более раз в разных Узлах. Шаблон может быть создан автоматически при создании компонента «Экран», «Программа» или «Связь с СУБД» в одном из Узлов Проекта. Каждый вновь создаваемый канал также приводит к автоматическому созданию шаблона канала в соответствующем слое «База каналов». Компонент может быть удалён из Узла, но шаблон этого компонента с уникальным порядковым номером будет храниться в слое шаблонов компонентов данного вида.

Вид-пример Навигатора проекта представлен на рисунке 2.7.

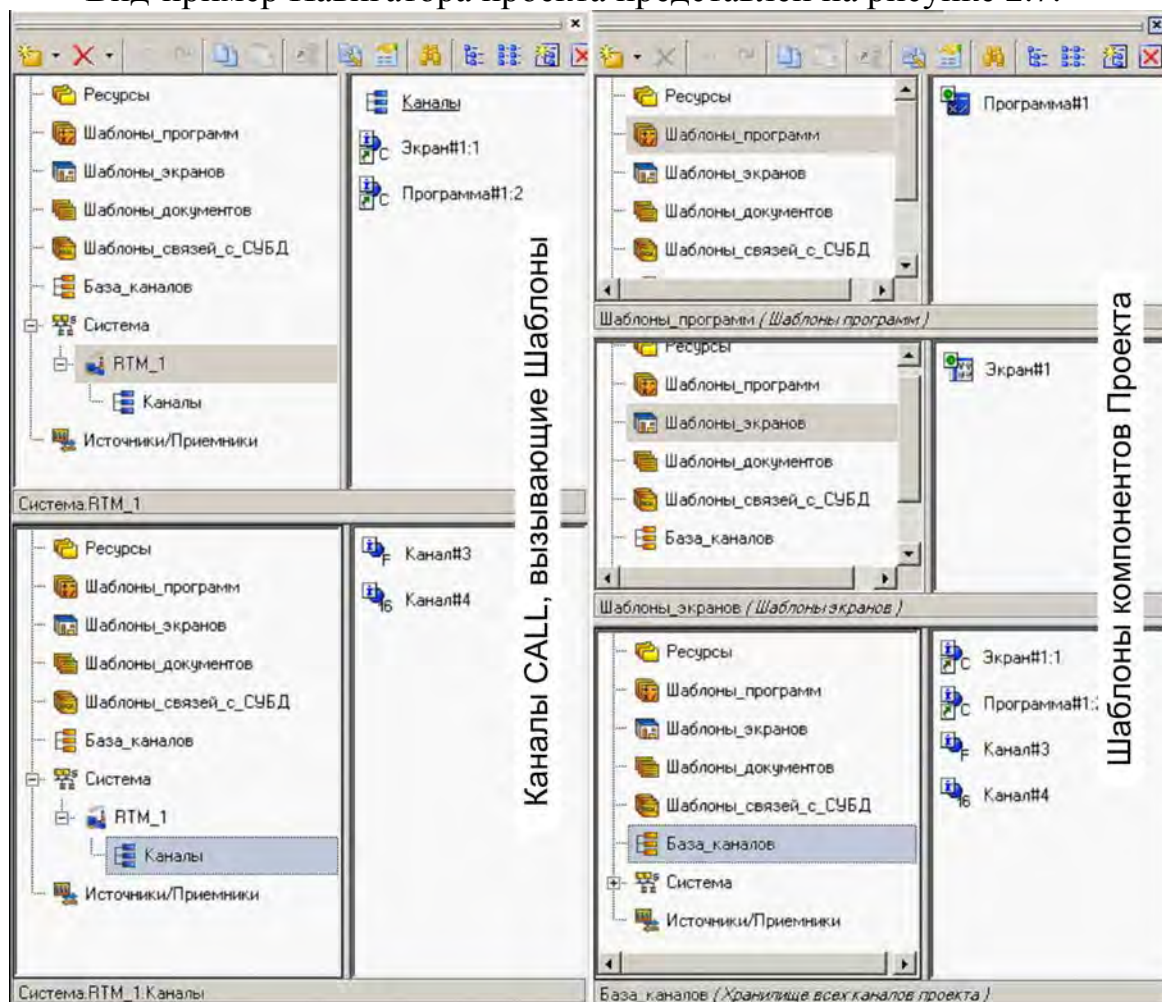


Рисунок 2.7. Вид окна Навигатора проекта (шаблоны компонентов показаны справа, каналы класса CALL вызывающие шаблоны показаны слева)

На рисунке 2.7 слева представлены компоненты: «Экран», «Программа» и два канала разных типов (FLOAT и HEX16). Справа на рисунке видно, что в результате добавления каналов в структуру Проекта, в его составе появились шаблоны компонентов «Программа», «Экран» и 4-х каналов. Очевидно, что в слое «База каналов» кроме каналов вызывающих шаблоны каналов класса FLOAT и HEX16 были созданы каналы класса CALL, которые вызывают шаблоны компонентов «Программа» и «Экран». Этот факт подтверждает ранее описанную особенность проектирования ПО в ИС. Чёткий порядок нумерации каналов показывает, в какой последовательности они были созданы.

2.4. Переменные в TRACE MODE 6

В TRACE MODE 5 каналы были универсальной и единственной единицей-носителем информации в рамках Проекта. Компоненты графического интерфейса оператора и элементы мнемосхемы для динамизации могли быть привязаны только к его выводам («Аппаратное», «Реальное», «Входное», «Выходное»). В отличие от 5-й версии TRACE MODE в шестой появилось полноценное понятие переменной, которыми обладают все компоненты Проекта.

Внимание, при создании шаблона компонента «Экран» в нём обязательно должен быть создан хотя бы один аргумент, поскольку в противном случае он не может быть вызван при работе Проекта в контроллере.

Рассмотрим на примере способ динамизации (изменения свойств или характеристик в течение времени в зависимости от какого-либо критерия) графического элемента мнемосхемы с использованием переменных компонента Проекта «Экран» и «Генератор». Для этого необходимо:

- 1) создать Проект;
- 2) создать Узел типа RTM;
- 3) создать компонент «Экран» и привязать его к каналу класса CALL, который будет вызывать его из слоя «Шаблоны экранов»;
- 4) создать математическую модель генератора унифицированного сигнала (например, синусоиды);
- 5) создать в рабочем поле компонента «Экран» графическую форму отображения текстовой информации;
- 6) привязать текстовую графическую форму к выходу математической модели генератора унифицированного сигнала.

Для выполнения некоторых пунктов необходимо учесть следующие рекомендации:

- добавление в состав элементов Проекта компонента «Экран» может быть выполнено непосредственно в слое «Система.RTM» либо в слое «Шаблоны экранов», а затем необходимо методом «Drag and Drop» выполнить перенос шаблона компонента «Экран» в Узел RTM;
- математическая модель генератора может быть создана в слое «Источники/Приёмники» либо в виде компонента «Программа»;
- для индикации выходного значения математической модели генератора можно использовать графический объект «Текст», при этом используемый вид индикации – «Значение» (рисунок 2.8);
- привязка значения графического элемента «Текст» должна быть выполнена к атрибуту «Значение» математической модели генератора (рисунок 2.9).

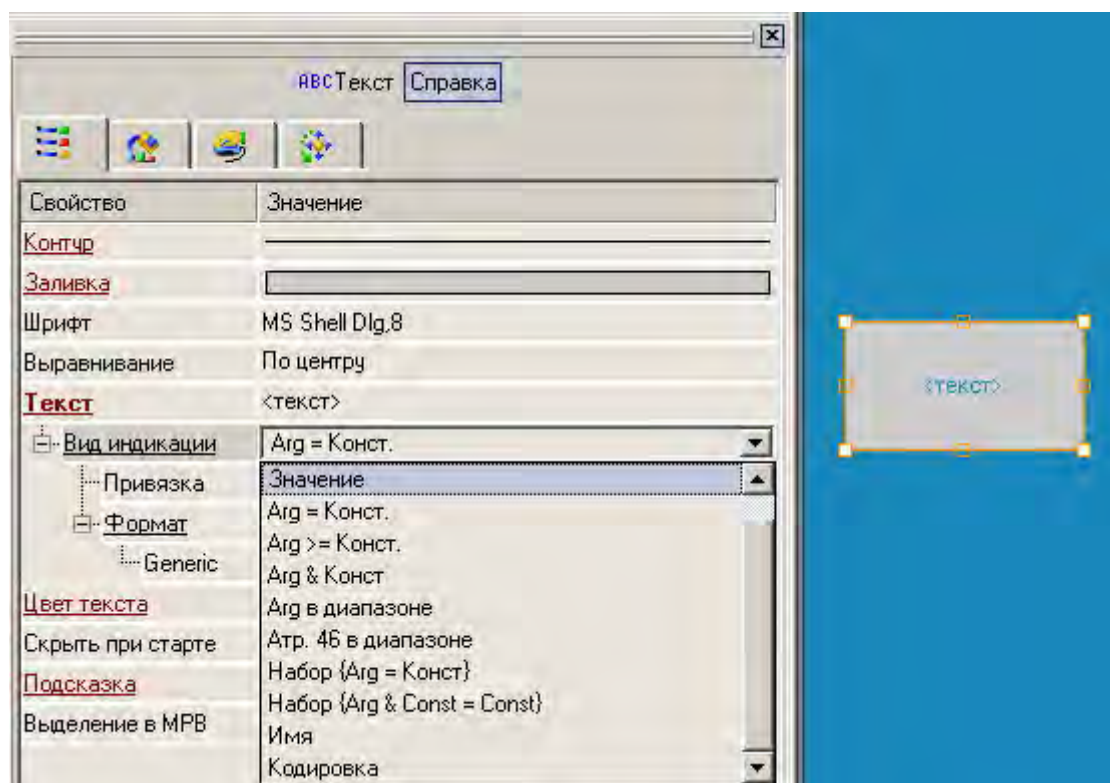


Рисунок 2.8. Вид редактора свойств графических элементов и рабочей области редактора представления данных программы ИС. Графический элемент «Текст» расположен справа, свойства компонента – слева

В описанном примере привязка математической модели генератора сигнала выполняется напрямую к переменной компонента «Экран» и данный пример является работоспособным.

Важно отметить, что привязка (рисунок 2.9) может быть выполнена как к одному из атрибутов компонента, так и к любому из аргументов данного компонента.

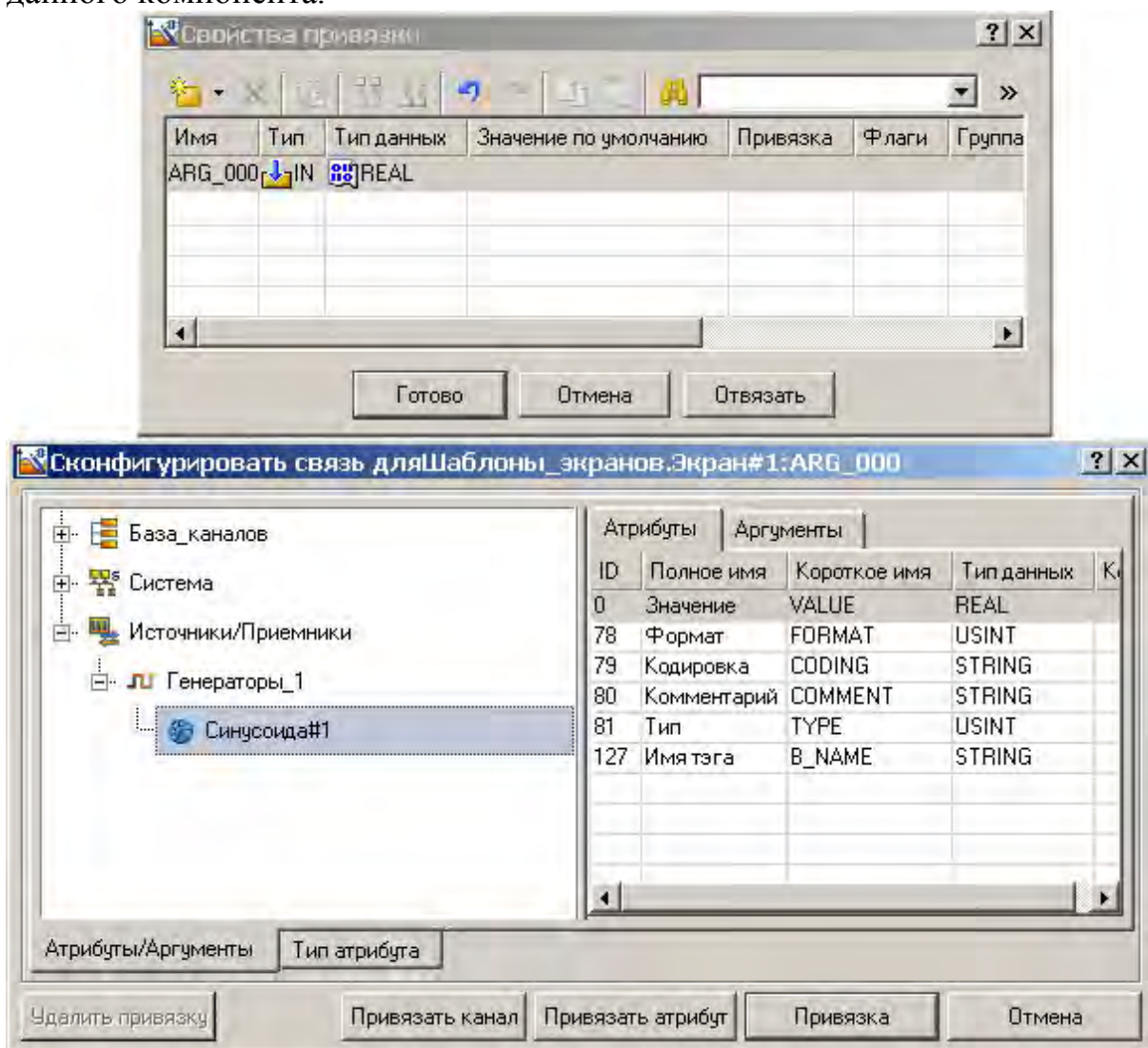


Рисунок 2.9. Вид редактора настройки свойств привязки графического элемента «Текст»

2.5. Редактор-навигатор переменных (аргументов)

Очевидно, что переменная в компьютерной программе играет наиболее важную роль. Важно понимать, что в крупных Проектах переменных может быть значительное количество. Для того чтобы эффективно оперировать ими в составе ИС ТМ 6 используется специальный навигатор, вид окна программы навигатора представлен на рисунке 2.10.

Открыть навигатор аргументов можно через меню «Свойства» шаблона компонента. Для эффективного управления аргументами в составе навигатора переменных содержатся следующие основные поля: «Имя», «Тип», «Тип данных», «Значение по умолчанию», «Привязка», «Группа», «Единица измерения» и «Комментарий». Название каждой переменной должно быть уникальным в рамках одного компонента и не может содержать пробелов. Тип переменной указывает на направление передачи информации и выполняет роль фильтра при привязке аргументов. Тип аргумента, таким образом, может быть «IN» или «OUT», а также «IN/OUT». Очень важно использовать поле «Комментарий» и «Единица измерения», поскольку они позволяют более детально описать назначение переменной, повышают эффективность обслуживания данного программного обеспечения другими программистами через длительный промежуток времени после его создания.

При создании нового Проекта следует учитывать одну особенность Редактора аргументов, название нового аргумента автоматически изменяется при его привязке к другому аргументу или каналу. Данная функция может быть выключена в окне программы настройки ИС (рисунок 2.11).

Групповые операции

При работе с аргументами или каналами, над ними можно выполнять групповые операции. Для того, чтобы задать некоторое свойство группе аргументов, выделите данную группу, удерживая клавишу «Ctrl», двойным кликом левой кнопки мыши откройте для редактирования одно из полей редактора. После назначения нового значения в данном поле новое значение будет скопировано в остальные поля столбца редактора характеристик аргументов выделенной группы.

Дополнительную информацию смотрите в разделе руководства пользователя «Поля редактора аргументов».

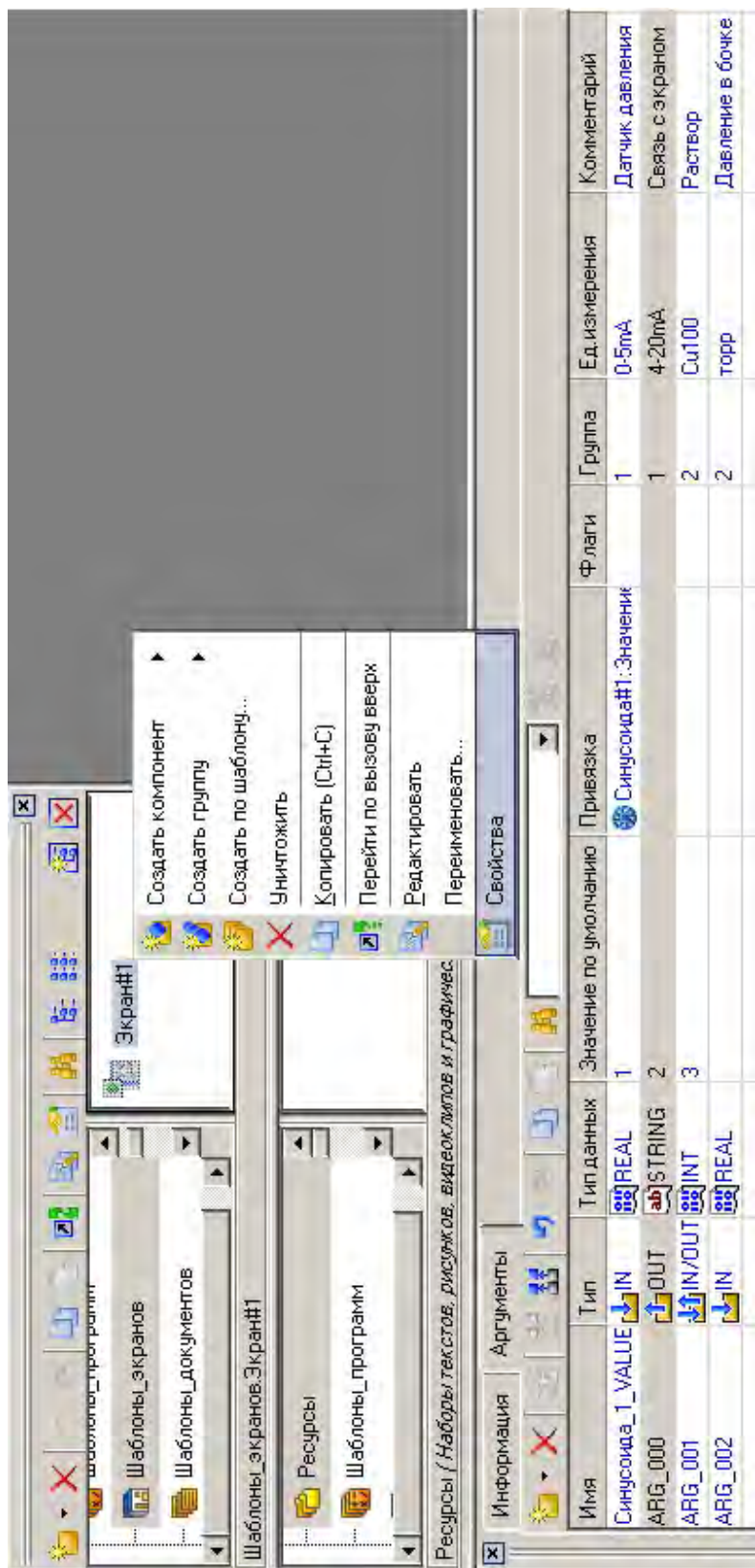


Рисунок 2.10. Вид окна программы «Навигатора аргументов»

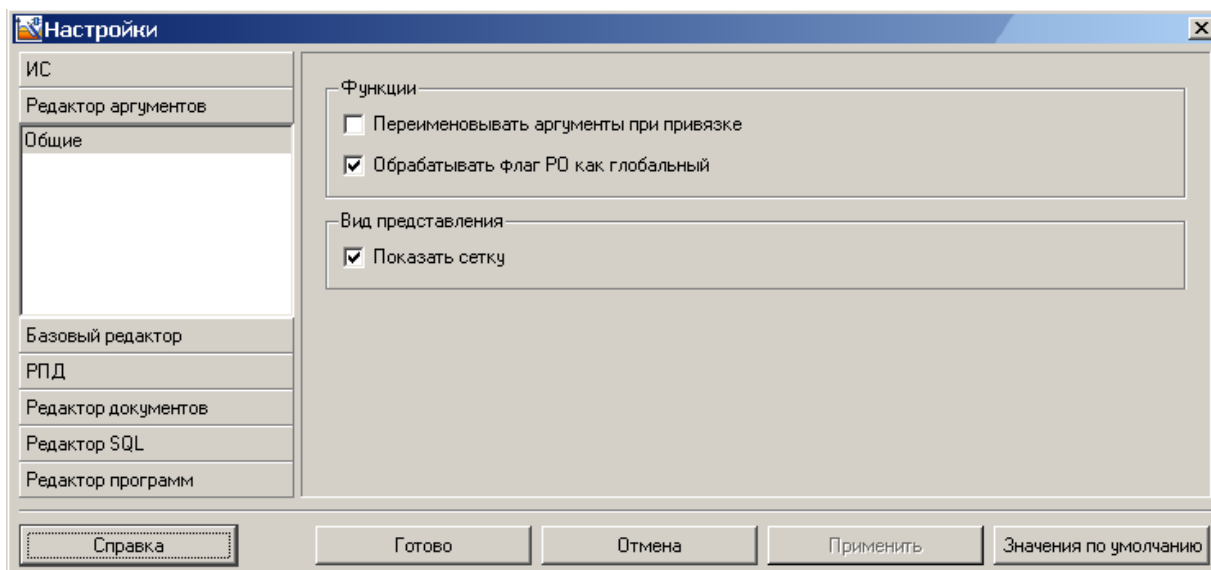


Рисунок 2.11. Вид окна программы настройки ИС ТМ

2.6. Подготовка Проекта к загрузке в контроллер, запуск МРВ на исполнение

Проект, созданный в интегрированной среде разработки, не является конечным файлом, который может быть загружен в контроллер, и участвовать в управлении технологическим оборудованием. Ранее было показано, что выдачу команд управления в рамках операционной системы выполняет МРВ, а алгоритм управления технологическим оборудованием записан в файлах Проекта предварительно интерпретированных для МРВ. Данная команда в ТМ называется «Сохранить для МРВ». Она может быть выполнена из меню окна программы ИС, с использованием специальной пиктограммы или из всплывающего меню Узла. Важно понимать, что Проект может разрабатываться для множества контроллеров АСУТП и в его составе содержатся Узлы для этих контроллеров, следовательно, необходимо интерпретировать каждый Узел Проекта в отдельности.

Рассмотрим типы файлов, создаваемых при экспорте Узлов Проекта. Они имеют следующее назначение:

- **<имя файла prj>_<ordinal>.cnv** – файл текстового формата, содержащий служебную информацию;
- **<имя файла prj>_<ordinal>.dbb** – бинарный файл Узла (используется МРВ);
- **<имя файла prj>_<ordinal>.dbx** – бинарный файл Узла (используется для конвертации Узла Проекта в файл для МРВ);

- **<имя файла prj>_<ordinal>.rtp** – вспомогательный текстовый файл Узла (используется MPB);
- **addr.ind** – описание параметров узлов (этот файл создается в папке каждого Узла);
- **<имя программы>.tmsd** – коды компонентов «Программа»;
- **<ordinal + 1>_<порядковый номер в Узле>.res** – файлы, содержащие исполняемый код, используемый MPB, нумерация файлов начинается с нуля.

Каждый комплект файлов для каждого из Узлов Проекта после выполнения операции «Сохранить для MPB» находится в отдельной папке на жёстком диске в корневом каталоге файла Проекта. Далее необходимо выполнить отладку Проекта в Профайлере – отладочном MPB. Запуск на выполнение Профайлера может быть выполнен из главного меню окна программы ИС (меню «Файл : Отладка») или с помощью кнопки с пиктограммой. Файл Профайлера также может быть запущен на выполнение самостоятельно, он расположен по следующему адресу: «<Local Disk>:\Program Files\AdAstra Research Group\Trace Mode IDE 6 Base\rtc.exe».

2.7. Вопросы для самоконтроля к главе №2

1. Опишите графический интерфейс пользователя программы IDE TRACE MODE 6.06.2?
2. Редакторы каких компонентов содержатся в программе ИС?
3. Назовите основные этапы Проектирования программного обеспечения в TRACE MODE 6.06.2?
4. Опишите структуру и состав элементов слоя «Топология».
5. Опишите структуру и состав элементов слоя «Технология»
6. Что такое шаблон (в рамках понятий TRACE MODE)?
7. Шаблоны каких элементов Проекта существуют в ТМ?
8. Возможно ли детально настроить состав слоёв в ИС, каким образом?
9. Какие слои могут быть включены или выключены в структуре навигатора Проекта?
10. Что происходит при выполнении функции «Удалить» над компонентом «Канал» в слое «Система»?
11. Могут ли иметь одинаковое название компоненты расположенные в слое «Система» и слое «Шаблоны экранов»?
12. Может ли шаблон компонента «Экран» быть расположен в слое «Система»?

13. Может ли шаблон компонента «Экран» быть создан выполнением некоторой команды в слое «Система»?
14. К какому типу программного обеспечения относится Проект?
15. Может ли Проект быть запущен на выполнение самостоятельно в рамках операционной системы и почему?
16. Назовите основные поля редактора аргументов?
17. Назовите основные особенности настройки редактора аргументов?
18. Какие операции должны быть выполнены обязательно после создания нового компонента «Экран», без выполнения которых данный компонент функционировать не будет?
19. Опишите процедуру подготовки Проекта предшествующую его загрузке в МРВ, запишите расширения файлов, которые создаются при выполнении данной процедуры?
20. Назовите основные типы аргументов используемых программистом в ИС TRACE MODE 6.06.2?
21. Почему при работе с аргументами важно использовать поле «Комментарий»?
22. Для чего используются каналы класса CALL в TRACE MODE?
23. Какие типы Каналов Вы знаете?
24. Что такое динамизация?
25. Возможны ли групповые операции в TRACE MODE? Насколько они эффективны? Над какими элементами Проекта они выполняются?

3. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ОБРАБОТКИ ДАННЫХ В TRACE MODE 6

Программирование – это процесс взаимодействия человека с ЭВМ. Компьютерная программа позволяет программисту заставить техническое средство выполнять заложенные в него функции по необходимому алгоритму. Наличие в техническом устройстве с электронным вычислительным модулем средств исполнения элементарных функций позволяет оператору их группировать в последовательности, тем самым составлять более сложные функции или подпрограммы.

Во многих языках программирования одного класса, группы элементарных функций похожи и являются универсальными, однако существуют языки программирования адаптированные для некоторой области применения. Например, такие как языки программирования стандарта ИЕС(МЭК) 61131-3. К ним относятся: SFC, FBD, LD, ST и IL. Эти языки программирования включают в свой состав значительное количество готовых математических функций обработки данных, широко используемых в АСУТП. К ним, также, относятся законы регулирования ПИД, ПД, ПИ или П.

Рассмотрим более детально принципы программирования функций обработки данных в САПР TRACE MODE 6.06.2.

3.1. Редактор программ

Одним из редакторов компонентов Проекта в ТМ является «Редактор программ» (рисунок 3.1).

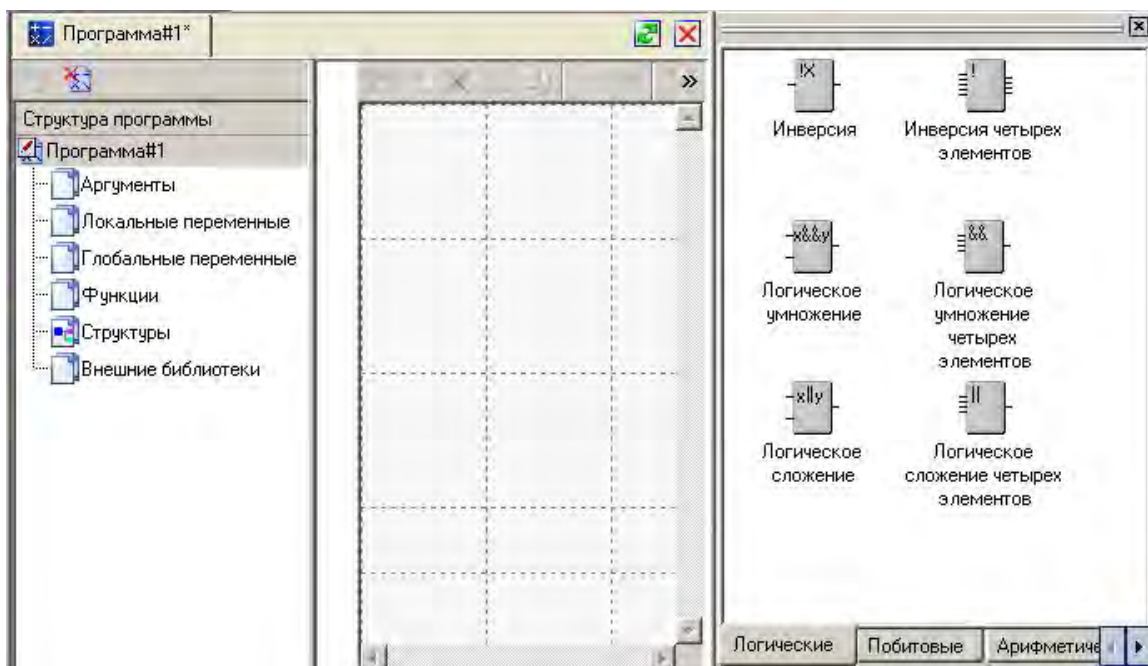


Рисунок 3.1. Вид окна программы «Редактор программ»

Редактор программ представлен навигатором компонента «Программа», рабочим полем текста программы (рисунок 3.1, в центре) и библиотекой функциональных блоков (в случае использования в качестве языка программирования FBD или LD).

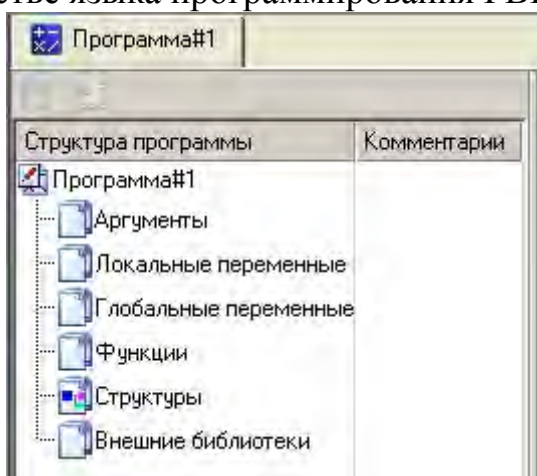


Рисунок 3.2. Вид окна Навигатора компонента «Программа»

Навигатор компонента «Программа» позволяет разработчику взаимодействовать со всеми элементами компонента. Выбор языка программирования выполняется при нажатии левой кнопки мыши (далее по тексту ЛК) на имени компонента «Программа» в вершине иерархии его структуры (рисунок 3.2).

Рассмотрим элементы структуры компонента Программа на примере рисунка 3.2.

- Аргументы компонента «Программа», как и других компонентов Проекта, используются для ввода\вывода данных из\в смежные компоненты. Аргументы могут, также, использоваться для операций обмена данными внутри компонента «Программа» и между ними. Следует отметить, что только аргументы могут служить

входами\выходами пользовательских функциональных блоков при создании таковых внутри компонента «Программа» на языках программирования FBD или LD.

- Локальные переменные используются в рамках объекта (программный код, функция, структура), в котором они определены. Они обнуляются после каждого цикла выполнения программы.
- Глобальные переменные используются в рамках компонента «Программа» и сохраняют свое значение между его вызовами. В частности, глобальными являются переменные FBD и LD-блоков.
- Функции (функции-блоки, подпрограммы) используются для создания собственных блоков-функций, например, FBD-блоков. Программирование данных блоков может осуществляться на любом другом языке программирования, например LD, IL или ST.
- Структуры используются в языке ST для отображения элементов программы.
- Внешние библиотеки позволяют использовать функции библиотек «*.dll» в составе программ Проекта, при этом внешняя библиотека функций должна находиться в выбранной директории.

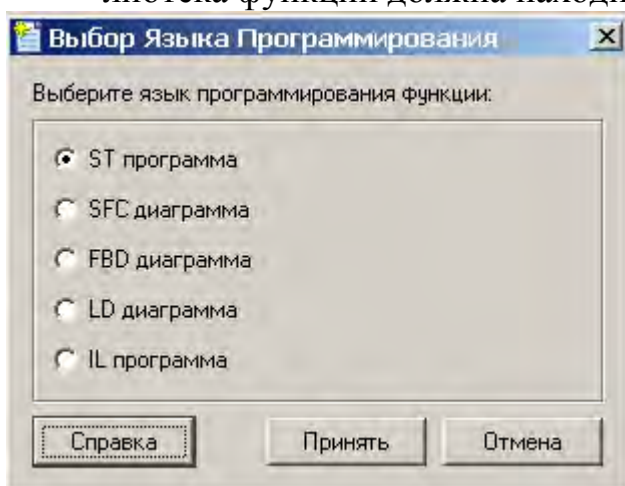


Рисунок 3.3. Вид диалогового окна программы выбора языка программирования

Для смены языка программирования (рисунок 3.3) в действующей программе необходимо использовать пиктограмму «Очистить тело функции» в окне навигатора компонента «Программа». Открыть или закрыть окно библиотеки функциональных блоков можно при помощи пиктограмм в окне Редактора программы, а также в меню «Вид».

Выделим несколько наиболее значимых языков программирования для изучения (FBD, IL, ST), остальные языки программирования, присутствующие в списке на рисунке 3.3 рекомендованы к самостоятельному изучению.

3.2. Язык программирования «Техно FBD»

FBD (Functional block diagram) или язык программирования с использованием функциональных блоков. Он представляет собой высо-

коуровневый объектно-ориентированный язык программирования, широко применяемый в АСУТП для создания алгоритмов управления данными. FBD-программа представляет собой диаграмму последовательно выполняемых функциональных блоков. Функциональные блоки представлены прямоугольными элементами на рабочем поле программы (рисунок 3.4). Каждый блок содержит название, обозначенные входы и выходы, а также уникальный номер. Блок может содержать в себе как одну математическую функцию, так и их набор. Таким образом, подавая определённые значения на входы функционального блока, можно получить на его выходах результат выполнения математической зависимости запрограммированной в нём.

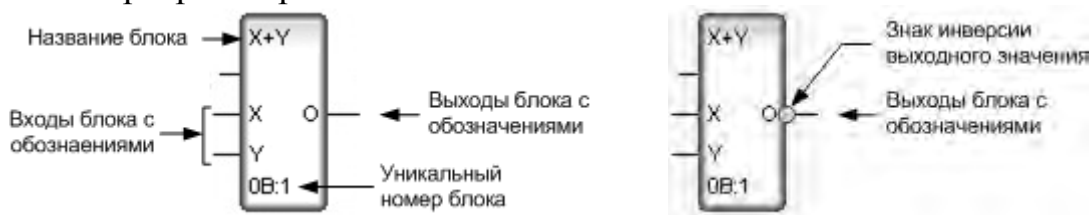


Рисунок 3.4. Вид функционального блока из стандартной библиотеки языка программирования FBD

На рисунке 3.5 показана FBD-программа, в которой запрограммирована зависимость « $Y = (3 + X)^2$ », где X – входное значение, Y – выходное значение.

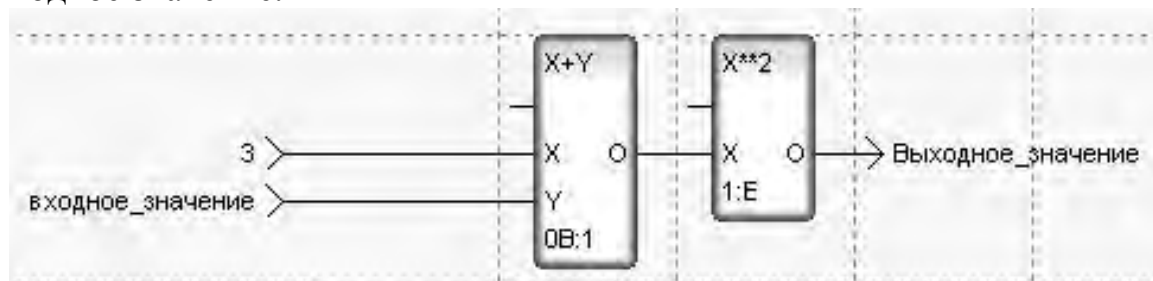


Рисунок 3.5. Вид диаграммы FBD-программы

Точка входа в программу при программировании на FBD, SFC и LD задаётся автоматически.

Процесс программирования, в данном случае, заключается в позиционировании функциональных блоков в рабочей области программы и соединении их входов и выходов. Входы и выходы должны быть подключены к аргументам, в противном случае они не будут доступны для привязки вне программы. Позиционирование функциональных блоков в рабочей области выполняется методом «Drag and Drop».

В библиотеке компонентов представлены блоки, сгруппированные по функциональному признаку. В состав группы входят: логические,

побитовые, арифметические, тригонометрические, алгебраические, блоки сравнения, блоки выборки значений, триггеры и счётчики, генераторы, блоки управления, блоки пересылки значений, блоки регулирования и пользовательские блоки.

Проверка работоспособности программы выполняется после компиляции её кода с использованием кнопки обозначенной пиктограммой в окне программы ИС или с использованием функциональной клавиши «F7». Запуск скомпилированной программы выполняется также с использованием специальной кнопки или с использованием функциональной клавиши «F5». При проверке работоспособности созданной диаграммы FBD-программы, действующие значения отображаются над выводами функциональных блоков (рисунок 3.6). Редактировать значения привязанных к входам блоков аргументов можно при помощи вкладки «Переменные», которая может быть вызвана из меню программы «Вид».

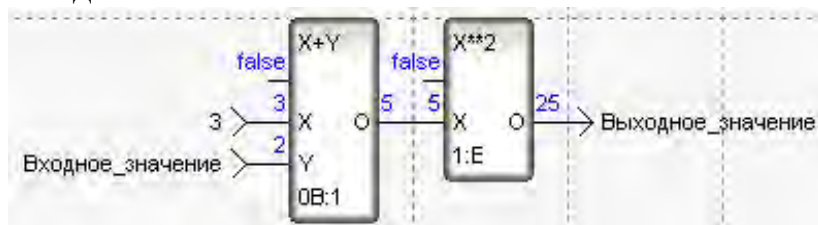


Рисунок 3.6. Вид FBD-программы в режиме проверки работоспособности

В том случае, когда в библиотеке отсутствует необходимой блок, существует возможность программирования собственных функциональных блоков, которые затем помещаются в библиотеку в группу пользовательских блоков. О методах создания пользовательских блоков будет написано далее.

Дополнительная информация содержится в разделе руководства пользователя по адресу: «TRACE MODE 6.06.2 : Программирование алгоритмов : Редактирование FBD-программ».

3.3. Язык программирования «Техно ИЛ»

Язык программирования с использованием инструкций ИЛ (Instruction list) является низкоуровневым ассемблероподобным. Компьютерные программы, написанные на ИЛ, могут выполнять команды в одноадресном и двухадресном режиме и, в большинстве случаев, используются профессионалами для получения специфических уникальных результатов от работы программы. Также данный язык программирования может быть использован для написания пользовательских функциональных блоков.

Синтаксис PL во многом напоминает команды ассемблера. Пример выполнения команд в одноадресном и двухадресном режиме представлен в таблице 3.1.

Таблица 3.1

Текст компьютерной программы, написанный на языке программирования PL (2 режима записи команд)

Функция	Текст программы в одноадресном режиме записи	Текст программы в двухадресном режиме записи
$a = a + b$	<i>LD a // result = a</i> <i>ADD b // result = result + b</i> <i>ST a // a = result</i>	<i>ADD a b</i>

При написании новой программы на PL необходимо учитывать, что точка входа в программу задаётся автоматически и определяется следующей конструкцией:

```
program
  {определение аргументов}
  {список команд}
end_program
```

Поэтому, необходимо при помощи редактора аргументов изначально обозначить переменные и аргументы, которые будут использоваться в программе (переменные могут быть добавлены позже). После того как они будут заданы, в программе автоматически инициализируется каждая из них. Пример программы с инициализированными переменными представлен ниже:

```
PROGRAM
  VAR_INPUT input : REAL; END_VAR
  VAR_OUTPUT output : REAL; END_VAR
  VAR local_var : REAL; END_VAR
  VAR const : REAL := 3; END_VAR

  {список команд}

END_PROGRAM
```

В качестве примера рассмотрим текст программы, в которой запрограммирована функция, показанная на рисунке 3.5. Текст программы представлен ниже:

```

PROGRAM
  VAR_INPUT input : REAL := 0; END_VAR
  VAR_OUTPUT output : REAL := 0; END_VAR
  VAR local_var : INT := 0; END_VAR
  VAR const : INT := 3; END_VAR
  VAR m : INT; END_VAR

  LD input // запись значения пер. input в аккумулятор
  ST local_var // запись значения аккумулятора в пер. local_var
  ADD local_var const // сложение знач. local_var и const, результат в local_var
  LD local_var // запись значения local_var в аккумулятор
  ST m // запись значения аккумулятора в пер. m
  MUL local_var m // умножение значения local_var на знач. m, результат в local_var
  LD local_var
  ST output

END_PROGRAM

```

Переменные, необходимые для программирования, создаются в навигаторе компонента «Программа». В этом примере были использованы два аргумента (Input и Output) для ввода исходных данных в программу и вывода результата, а также три локальных переменных, необходимых для внутреннего обмена данными. Одна из переменных используется в качестве константы.

Операторы LD и ST используются для записи и чтения результата вычисления в\из аккумулятора. Оператор ADD и MUL – для сложения и умножения операндов. В тексте программы также могут быть использованы операторы сравнения и переходов. Некоторые операторы могут быть использованы с модификаторами «С» и «Х». Так, например, инструкции JMP, CAL и RET – выполняются безусловно, а инструкции JMPC, CALC и RETC устанавливаются непосредственно после операторов сравнения и выполняются только при истинном результате их выполнения, в противном случае они игнорируются. Значение модификатора Х уточнить в руководстве пользователя к программе ИС самостоятельно.

Для проверки кода программы скомпилируйте и запустите её на выполнение.

Дополнительная информация содержится в разделе руководства пользователя по адресу: «TRACE MODE 6.06.2 : Программирование алгоритмов : Описание языка Техно II».

3.4. Язык программирования «Техно ST»

Язык программирования ST (Structured text) является высокоуровневым текстовым языком. Он используется как для написания самостоятельных программ, так и для программирования FBD-блоков. В составе текста ST-программы могут содержаться обыкновенные функции и функции-блоки.

В составе языка программирования ST могут быть использованы некоторые функции языка программирования Си, такие как: «cos», «sin», «log» или «exp», а так же специальные функции чтения и записи данных в порт. В ST допускается использовать функции, которые содержатся во внешних DLL-библиотеках. При программировании также можно использовать массивы, как одномерные, так и многомерные.

Важным элементов ST-программы является структурная переменная, которая в отличие от массива является совокупностью объектов имеющих различный тип. Переменные-структуры создаются в редакторе структуры компонента «Программа» в слое «Структуры».

Как и в случае использования языка программирования Техно-II в ST все аргументы и переменные необходимо инициализировать в соответствующем редакторе. После инициализации, они автоматически будут вписаны в текст программы. В Техно-ST поддерживаются следующие типы переменных:

- BOOL (bool) – булево значение размерностью 1 байт (true (1) или false (0));
- SINT (__int8) – целое со знаком размерностью 1 байт (-128 ... 127);
- USINT (unsigned __int8) – целое без знака размерностью 1 байт (0 ... 255);
- INT (short) – целое со знаком размерностью 2 байта (-32768 ... 32767);
- UINT (unsigned short) – целое без знака размерностью 2 байта (0 ... 65535);
- DINT (long) – целое со знаком (4 байта) (-2147483648 ... 2147483647);
- UDINT (unsigned long) – целое без знака (4 байта) (0 ... 4294967295);

- TIME, DATE, TIME_OF_DAY, DATE_AND_TIME – соответствуют DINT;
- REAL (float) – вещественное число (4 байта) (максимальное значение 3.402823466e+38);
- LREAL (double) – вещественное число (8 байт) (максимальное значение 1.7976931348623158e+308);
- STRING (char []) – 256 символов в кодировке UTF-8 (512 байт);

Текст программы строится на основе последовательно-расположенных операторов. В конце оператора с аргументами обязательно ставится символ «;».

Операторы языка программирования Техно-ST могут быть символьными (« == », « < », « > », « =< » и т.д.) и строковые. Поддерживаются все виды операторов-условий, -циклов. Пример:

Оператор IF

```
IF VAR_002 < 2 THEN
    VAR_000 = 200;
ELSIF VAR_002 < 1 THEN
    VAR_000 = 500;
ELSE
    VAR_000 = 300;
END_IF;
```

Оператор CASE

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

Оператор WHILE

```
WHILE VAR_000 > 2 DO
    VAR_000 = VAR_000 - 1;
    VAR_001 = VAR_001 + 2;
END_WHILE;
```

Оператор FOR

```
FOR VAR_000 = 10 TO 20 DO
    VAR_001 = VAR_001 + 2;
END_FOR;
```

В качестве локальной переменной может использоваться массив, одномерный или многомерный. Для инициализации одномерного массива, необходимо в редакторе локальных переменных в поле «[]» поставить цифру, соответствующую количеству элементов массива. Для инициализации многомерного массива, количество элементов (размерность) указывается цифрами через запятую. Пример:

Использование одномерного массива

```
FOR a = 0 TO 3 DO // Цикл
    FOR i = 0 TO 3 DO // Цикл
        IF Xi[i] > Xi[i+1] THEN // Условие
            var1 = Xi[i];
            Xi[i] = Xi[i+1];
            Xi[i+1] = var1;
        END_IF;
    END_FOR;
END_FOR;
```

Дополнительная информация содержится в руководстве пользователя ИС по адресу: «TRACE MODE 6.06.2 : Программирование алгоритмов : Описание языка Техно ST».

3.5. Язык программирования «Техно LD»

Язык программирования LD (Ladder diagram) – язык построения лестничных диаграмм. Он исторически образовался при необходимости программирования работы релейной автоматики. LD-программа состоит из последовательно выполняемых функциональных блоков. Изначально библиотеки LD-блоков формировались контактами (разомкнутый, замкнутый и т.д.). Модифицированный язык программирования «Техно LD» содержит в собственной библиотеке также функциональные LD-блоки.

Размещение LD-блоков выполняется в рабочей области, каждый из них подключается выводами к шине. В составе LD-программ используются две основные шины и вспомогательные. На основные шины замыкаются все блоки LD-программы, на вспомогательные шины могут замыкаться выводы блоков расположенные один над другим. Размещение блоков в рабочей области выполняется указанным ранее методом «Drag and Drop» из библиотеки блоков.

Отладка и запуск на выполнение LD-программ выполняется аналогичным другим программам образом.

Дополнительная информация содержится в руководстве пользователя ИС по адресу: «TRACE MODE 6.06.2 : Программирование алгоритмов : Редактирование LD-программ».

3.6. Создание пользовательских функциональных блоков

При отсутствии необходимого функционального блока требуется создать новую функцию. Эта процедура выполняется в навигаторе компонента «Программа». Функции будет присвоено название «FUNCTION» с текущим порядковым номером. Её можно переименовать, выполнив двойное нажатие ЛК мыши на названии. Название функции будет в дальнейшем присвоено функциональному блоку в пользовательской библиотеке.

Для того чтобы новый функциональный блок имел входы и выходы, необходимо задать аргументы функции. Аргументы типа Input определяют входы блока, а аргументы типа Output – его выходы. Количество аргументов типа Input и Output определяет количество входов и выходов функционального блока.

Тело функции может быть запрограммировано на любом из предлагаемых языков программирования (рисунок 3.3). Вызов диалогового окна для выбора языка программирования выполняется нажатие ЛК мыши на названии новой функции. Сменить язык программирования можно посредством команды «Очистить тело функции», как это было описано ранее.

Пусть в качестве нового функционального блока будет выполнена программа, описанная в предыдущем параграфе, которая несёт в себе функцию « $Y = (X + 3)^2$ ». Описанный пример должен выглядеть в ИС следующим образом (рисунок 3.7). В структуре компонента «Программа» представлена функция, аргументы и переменные данной функции. Названия аргументов и функции отображаются на FBD-блоке. Блок также представлен в библиотеке FBD-блоков в группе «Пользовательские». Позиционирование нового блока выполняется указанным ранее методом «Drag and Drop». Стоит отметить, что созданные таким образом функциональные блоки могут быть также использованы в LD-программах.

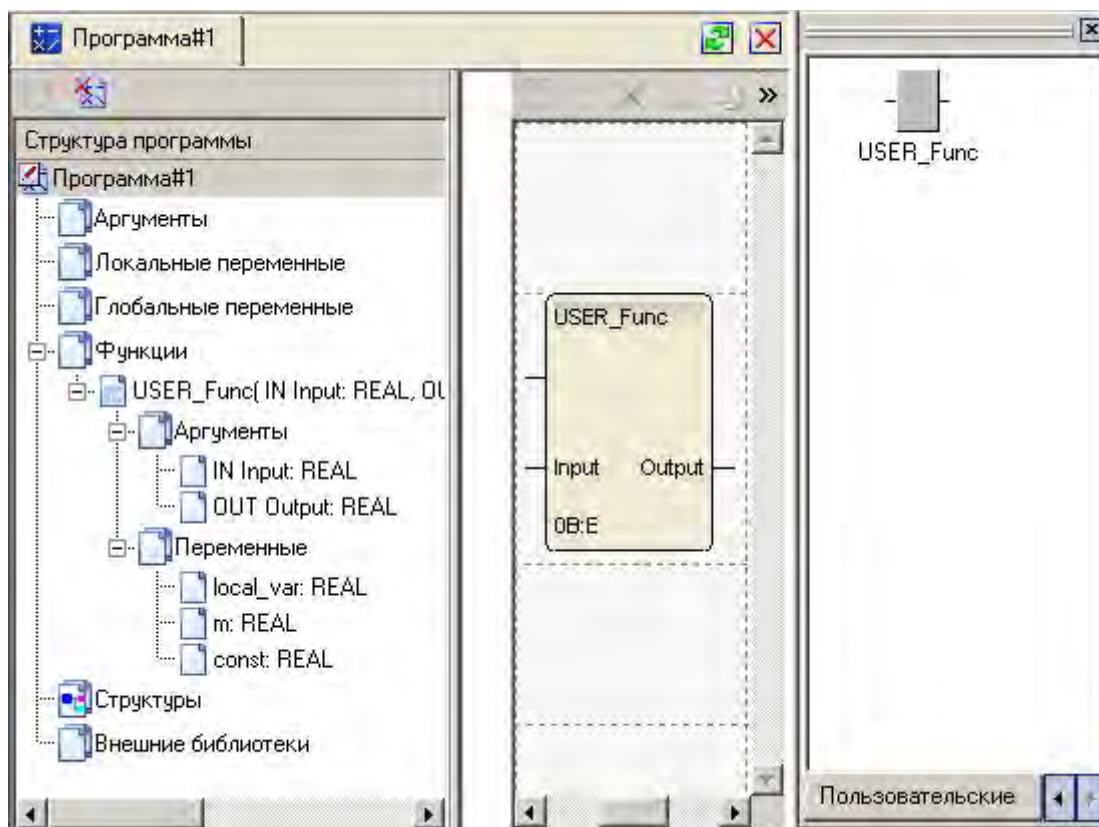


Рисунок 3.7. Вид окна Редактора программ

В ST-программах такие функции используются как блок-функции в тексте программы.

Для закрепления материала выполните практические задания (Приложение А).

3.7. Вопросы для самоконтроля к главе №3

1. Что такое компьютерная программа?
2. Какие языки программирования принадлежат стандарту IEC(MЭК) 61131-3 и используются в TRACE MODE IDE 6.06.2?
3. Что такое «Техно IL»?
4. Что такое «Техно ST»?
5. Что такое «Техно LD»?
6. Что такое «Техно FBD»?
7. Что такое «Техно SFC»?
8. Какова скорость выполнения простейшей IL-программы (использовать любые единицы измерения времени, обосновать ответ)?
9. Какие языки программирования в ТМ используют функциональные блоки в качестве функций?

10. Назовите оператор цикла, используемый в языке программирования «Техно ПЛ»?
11. Назовите операторы работы с аккумулятором в языке программирования «Техно ПЛ»?
12. Назовите основные элементы SFC-диаграммы?
13. Назовите все модификаторы оператора JMP, объясните их назначение?
14. Напишите на языке программирования «Техно ПЛ» одноадресную операцию сложения?
15. Напишите на языке программирования «Техно ПЛ» двухадресную операцию сложения?
16. Напишите код тела программы FBD-блока с одним входом и одним выходом?
17. Возможно ли использовать в составе программы в ТМ функции внешних DLL библиотек?
18. Опишите структуру компонента «Программа»?
19. Сколько шин в языке программирования «Техно LD» являются основными?
20. Каким образом может быть выполнена операция создания пользовательского функционального блока?
21. Каким образом можно запрограммировать название функционального блока?
22. Каким образом можно запрограммировать название входа функционального блока?
23. Может ли какой либо из языков программирования в ТМ использовать в качестве переменной двумерную матрицу, если да, то какой?
24. Опишите способ инициализации многомерной матрицы в качестве переменной ST-программы?
25. Назовите отличие аргумента от локальной переменной?
26. Каково назначение глобальной переменной?
27. Существует ли функция группировки в редакторе программ «Техно FBD», можно ли назначить метку группе блоков?
28. Назовите основные компоненты библиотеки функциональных блоков «Арифметические»?
29. Каково минимальное количество входов и выходов функционального блока?

4. РАЗРАБОТКА ПРОЕКТА ПРИКЛАДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В TRACE MODE 6

Прикладное программное обеспечение в составе АСУ выполняет задачи анализа параметров объекта управления и технологического оборудования (косвенно или напрямую) и выдачу команд управления. Пусть прикладное ПО располагается в составе регулятора, функции которого выполняет, например, микропроцессорный контроллер (рисунок 4.1).

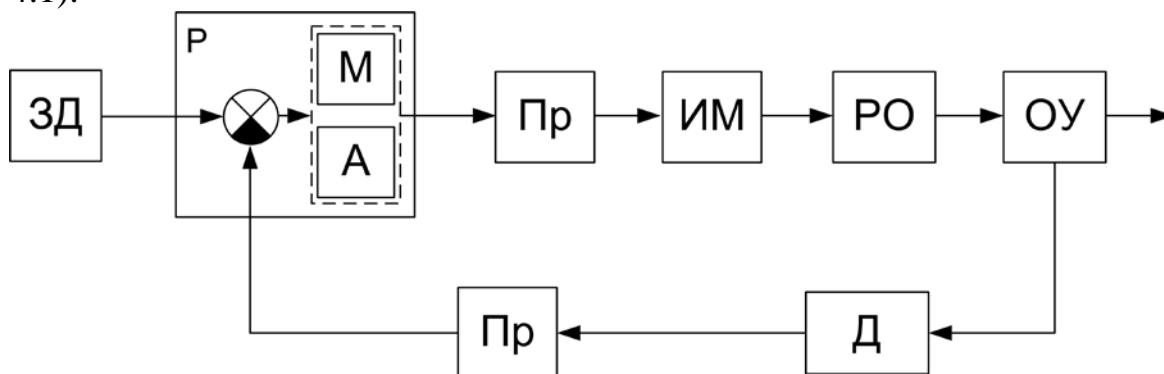


Рисунок 4.1. Структурная схема контура управления АСУ: ОУ – объект управления, Д – датчик, Пр – преобразователь характеристик сигнала, ЗД – задатчик уставки, Р – регулятор, ИМ – исполнительный механизм, РО – регулирующий орган, М – математические функции (в том числе законы регулирования), А – алгоритмы обработки данных и управления

Одной из задач, выполняемых при проектировании САУ, САР или АСУ является разработка алгоритмического и математического обеспечения, а затем программирование функций регулятора. Функции контроллера определяются технологическим процессом и описываются на функциональных схемах автоматизации. Данный тип схем в процессе автоматизации является отправной точкой, так как на них показывают функции, которые должно выполнять технологическое оборудование и узлы АСУ в процессе производства конечного продукта.

Следует отметить, что для выявления некоторых характеристик сложных или дорогостоящих систем управления часто применяют математическое моделирование, как метод представления некоторого объекта с ограниченным и достаточным количеством его характеристик. Используя принципы математического моделирования, выполняют проекты различных лабораторных стендов, необходимых для обучения сотрудников предприятий или оперативного определения некоторых свойств ОУ или его частей.

В данном разделе пособия рассмотрим некоторые химические или иные технологические производства в качестве объекта исследования и

автоматизации. Поскольку реального оборудования АСУ и ОУ не существует, воспользуемся средствами математического аппарата среды разработки ТМ для их создания. Использование математических моделей элементов АСУ позволит потренироваться в настройке каналов управления и технологического контроля в составе Проекта, программировать функции управления и алгоритмы обработки данных. Разработка виртуальных АСУ позволит закрепить полученные в предыдущих главах знания и систематизировать их.

4.1. Правила чтения функциональных схем автоматизации

Функциональные схемы автоматизации (ФСА) применяются в АСУ для представления функции, которые должно выполнять технологическое оборудование для производства конечного продукта, а также место приложения этих функций. Рассмотрим пример выполнения ФСА (рис. 4.2).

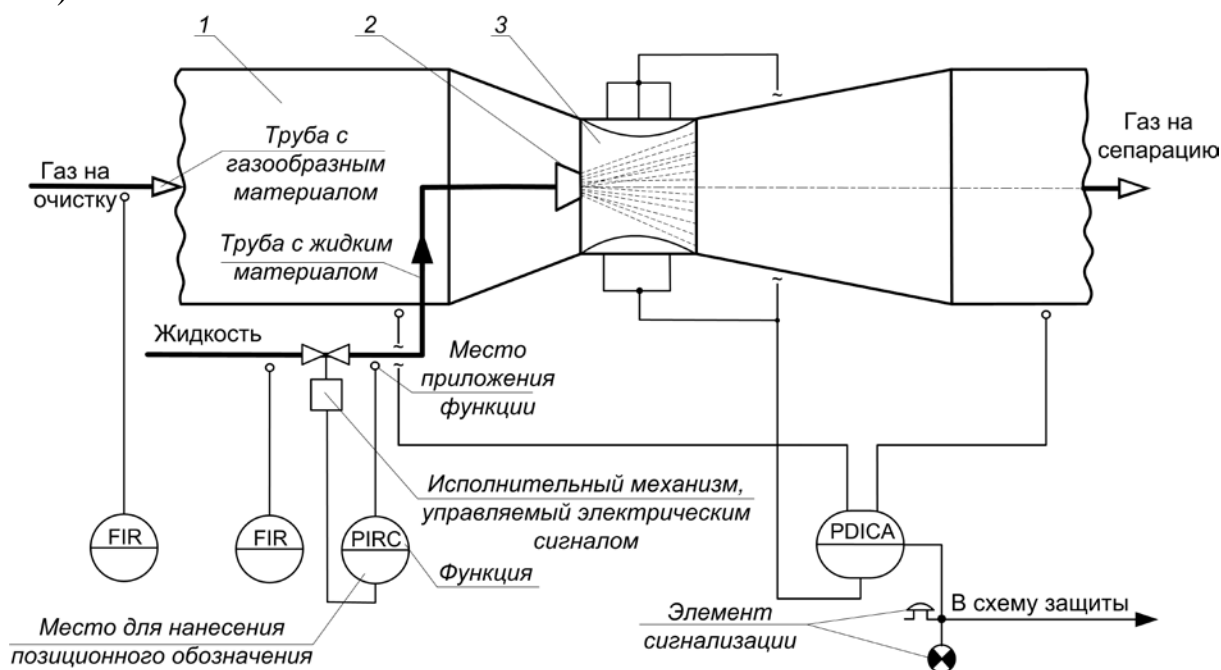


Рисунок 4.2. ФСА установки жидкостной очистки газов:

- 1 – корпус трубы Вентури;
- 2 – форсунки;
- 3 – горловина трубы Вентури с регулируемым диаметром

В соответствии с ГОСТ 21.404-85 «Автоматизация технологических процессов» обозначения на ФСА читаются слева направо. Первым ставится маркер основного обозначения измеряемой величины (например, P – давление). Вторым – дополнительное значение измеряемой величины (например, D – перепад давлений). Третьим – обозначение

функционального признака прибора (например: I – индикация, R – регистрация, C – управление, A – сигнализация). Некоторые обозначения, используемые на ФСА, представлены в таблицах 4.1- 4.3. Дополнительную информацию см. в вышеуказанном ГОСТ.

Таблица 4.1

Таблица измеряемых параметров и их обозначений на ФСА

<i>№</i>	<i>Измеряемый параметр</i>	<i>Обозначение</i>
1	Плотность	D
2	Некоторая электрическая величина	E
3	Расход	F
4	Размер, положение, перемещение	G
5	Уровень	L
6	Влажность	M
7	Давление, вакуум	P
8	Состав, концентрация и т.д.	Q
9	Радиоактивность	R
10	Скорость, частота	S
11	Температура	T
12	Несколько разнородных измеряемых величин	U
13	Вязкость	V
14	Масса	W

Таблица 4.2

Таблица уточняющих параметров и их обозначений на ФСА

<i>№</i>	<i>Уточняющий параметр</i>	<i>Обозначение</i>
1	Разность, перепад	D
2	Соотношение, доля, дробь	F
3	Автоматическое переключение	J
4	Интегрирование, суммирование	Q

Таблица 4.3

Таблица функций, выполняемых приборами

<i>№</i>	<i>Функция</i>	<i>Обозначение</i>
1	Сигнализация	A
2	Показание	I
3	Регистрация	R
4	Регулирование	C
5	Включение, выключение, переключение	S
6	Ручное управление	H

4.2. Алгоритм проектирования прикладного ПО

Для проектирования программного обеспечения в ТМ в учебных целях необходимо придерживаться следующего алгоритма действий:

1. необходимо прочитать ФСА, используя правила чтения таких схем;
2. разработать упрощенную структурную схему АСУ;
3. разработать математическое описание элементов АСУ;
4. разработать функциональную схему АСУ;
5. разработать структурную схему нового прикладного ПО, предварительно определив его место в составе описанной выше АСУ;
6. разработать структуру Проекта и создать необходимые элементы в его составе;
7. запрограммировать функции автоматического или автоматизированного управления и контроля параметров тех. процесса (в соответствии с ФСА).
8. разработать графический интерфейс оператора и запрограммировать его в составе компонентов «Экран» Проекта;
9. разработать алгоритмы проверки работоспособности полученного программного продукта;
10. выполнить проверку программного продукта на работоспособность и на соответствие описанным в индивидуальном задании функциям.

Для получения практических навыков комплексной разработки прикладного ПО в TRACE MODE 6.06.2 используйте задания из Приложения Б. Задание содержит ФСА некоторого технологического процесса, описание этого процесса и рекомендации к выполнению работы. Используйте полученные в предыдущих главах знания, для выполнения заданий. С целью изучения принципов программирования в TRACE MODE при моделировании элементов АСУ (датчиков, клапанов, объектов управления) максимально упрощайте математическое описание модели, поскольку эти задачи не являются приоритетными в данной работе. Например, для математического описания клапана, можно использовать полупериод синусоиды или линейную зависимость (рисунок 4.3).



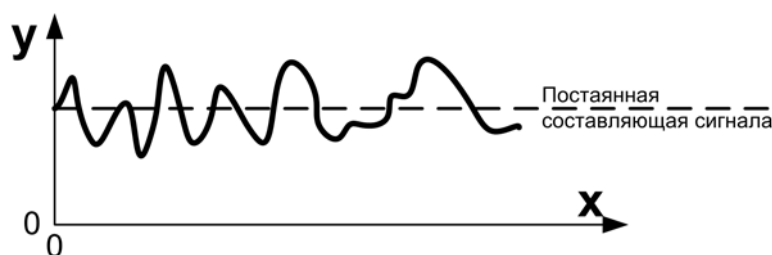
Математическое описание клапана:

$$y = \sin(x) \text{ [степень откр.]};$$

$$x = 0 \dots 90 \text{ [град.]}$$

Рисунок 4.3. Переходная характеристика моделируемого клапана

Датчик некоторого физического параметра исследуемого вещества можно описать системой уравнений, состоящей из статической и динамической составляющей:



Математическое описание датчика:

$$y = T \pm 10\% \text{ [K]};$$

$$x = 0 \dots \text{inf} \text{ [c]};$$

$$T = 10 \text{ [K]};$$

Рисунок 4.4. Временная характеристика датчика

В случае косвенного формирования сигналов датчика, дополнительный выход должна иметь математическая модель объекта управления, например бака или нагревателя (бойлера). Пусть существует система нагрева воды в баке, в которой, датчиком, определяется температура воды. При условии, что потерь тепла нет и весь объём жидкости нагревается равномерно, мощность нагревателя всегда постоянна, а также учитывая то, что 1 кВт энергии нагревает за 1 час 860 литров воды на 1 К, можно описать систему нагрева воды следующей системой уравнений:

$$\begin{cases} T = \frac{W \cdot t + k \cdot V \cdot T_0}{k \cdot V}; & \text{где:} \\ W = 1[\text{кВт}]; & T - \text{температура нагреваемой воды;} \\ V = 100[\text{л}]; & T_0 - \text{начальная температура воды;} \\ k = 0.00117; & V - \text{объём воды в баке;} \\ T_0 = 293[\text{K}]. & W - \text{мощность нагревателя.} \end{cases}$$

При изменении мощности нагревателя можно увеличивать скорость нагрева воды, при отводе тепла продуктом – продукта.

5. УПРАВЛЕНИЕ ПРАВАМИ ДОСТУПА ПЕРСОНАЛА, ПАРОЛЛИРОВАНИЕ

Системы контроля и управления доступом (СКУД) широко используются на предприятиях. Как правило, это программно-технические комплексы, выполняющие физическую защиту элементов производства. Поскольку разработанное в САПР программное обеспечение имеет непосредственные прямые каналы выдачи управляющих сигналов оборудованию АСУ, необходимо использовать специальные функции СКУД персонала (операторов, системных программистов, супервизоров). В TRACE MODE такие инструменты существуют.

Поскольку при разработке СКУД необходимо четко разграничить права операторов, ограничение доступа неквалифицированного персонала к элементам управления также является важной задачей. В TRACE MODE для этого используется специальный информационный канал класса «Пользователь». Данный тип каналов располагается в группе «Пользователи_ТМ» Узла Проекта в слое «Система». Рассмотрим на примере правила настройки прав доступа персонала к графическим элементам управления и компонентам «Экран».

На рисунке 5.1 представлены настройки канала класса «Пользователь» созданного для пользователя «Начальник». В составе полей редактора присутствуют:

- группы полей настройки правил авторизации пользователя в программе;
- группа полей системных настроек;
- группа полей настройки прав управления записями пользователей в Проекте в процессе эксплуатации АСУ;
- группа полей настройки персональных данных пользователя.

Для авторизации пользователя при запуске программы ему необходимо использовать логин и пароль. Пусть для пользователя «Начальник» будет задан логин «bos», пароль не назначен. Для запуска на выполнение МРВ и Проекта, его остановка и изменения положения графических элементов управления необходимо выставить флаги в соответствующих графических формах группы «Система» (рисунок 5.1). В группе графических форм «Экраны» и «Формы» представлены флаги, каждый из которых соответствует биту в кодовом слове (двоичный формат слова). Номер бита соответствует коду доступа к графической форме (десятичный формат) на графическом экране (рисунке 5.2).

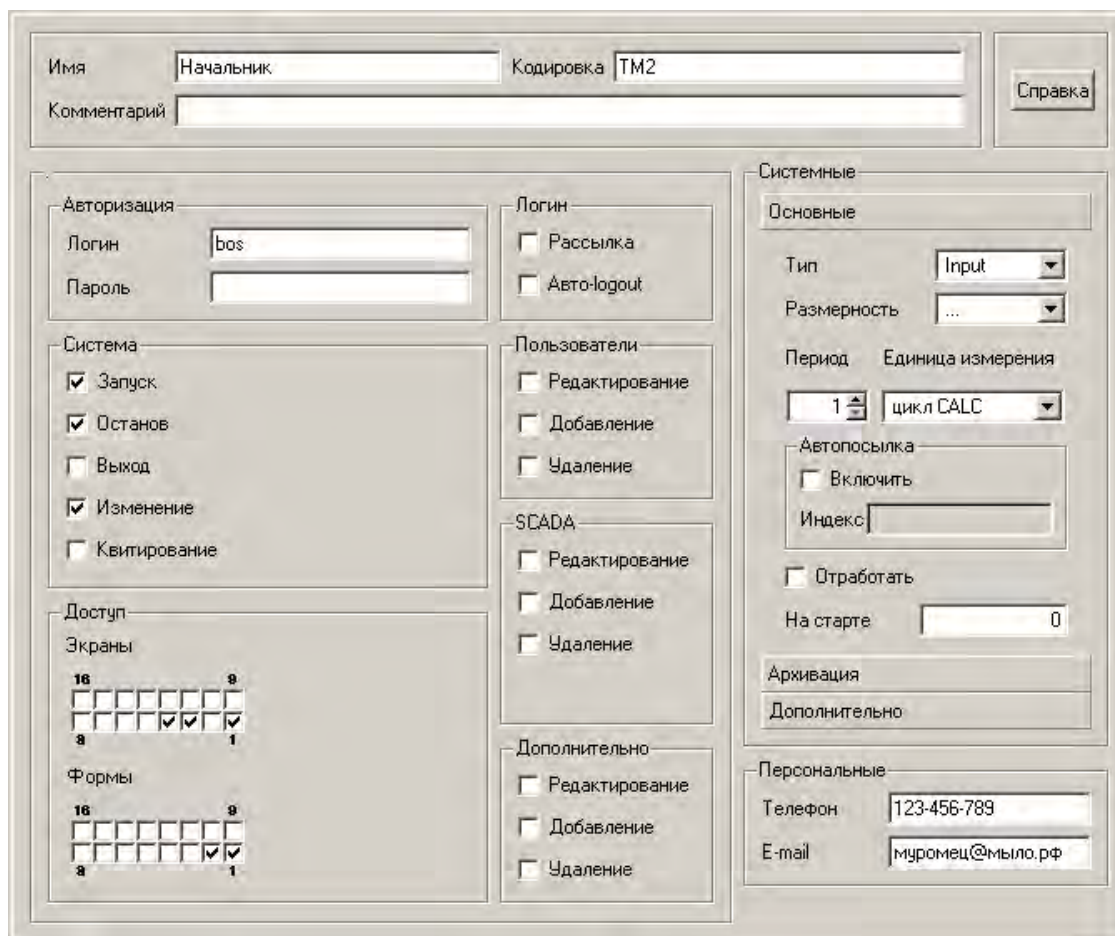


Рисунок 5.1. Вид редактор параметров настройки канала класса «Пользователь»



Рисунок 5.2. Графический интерфейс пользователя программы-примера (основное окно)

В соответствии с настройками, представленными на рисунке 5.1, пользователю «Начальник» разрешён запуск, останов МРВ и изменение состояния динамических объектов формирующих команды управления. Также разрешён доступ на первый, третий и четвёртый графические «Экраны». Их структура представлена на рисунке 5.3. Для формирования команд управления пользователь «Начальник» может использовать графические формы с десятичным кодом доступа «1» и «2» (рисунок 5.2, 5.4, 5.5).

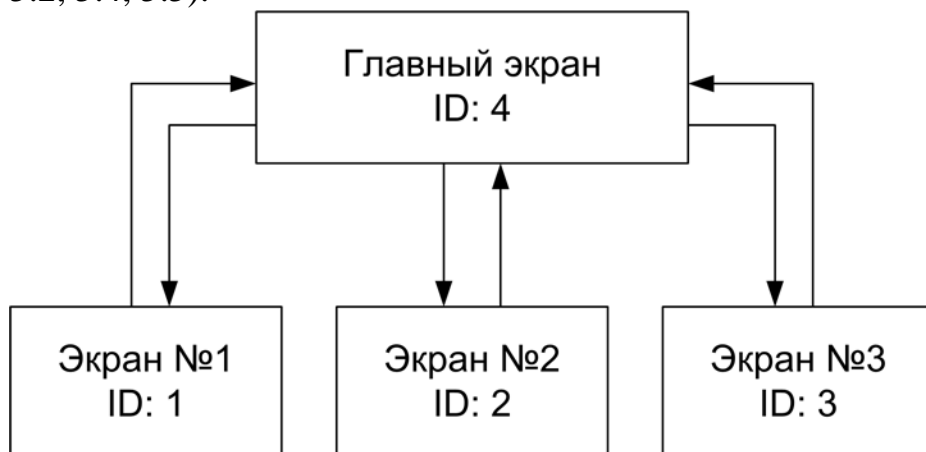


Рисунок 5.3. Структура интерфейса пользователя программы-примера

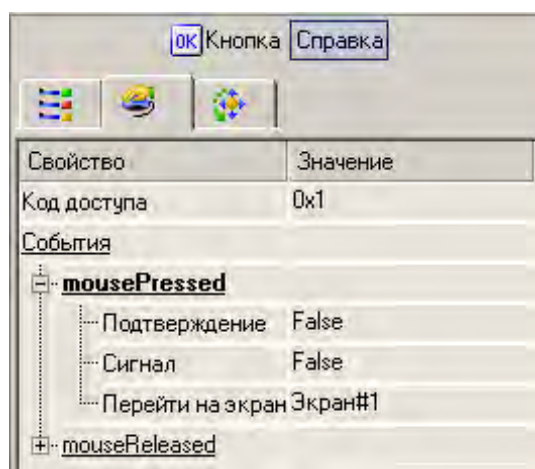


Рисунок 5.4. Вид редактор параметров настройки графического элемента «Кнопка»

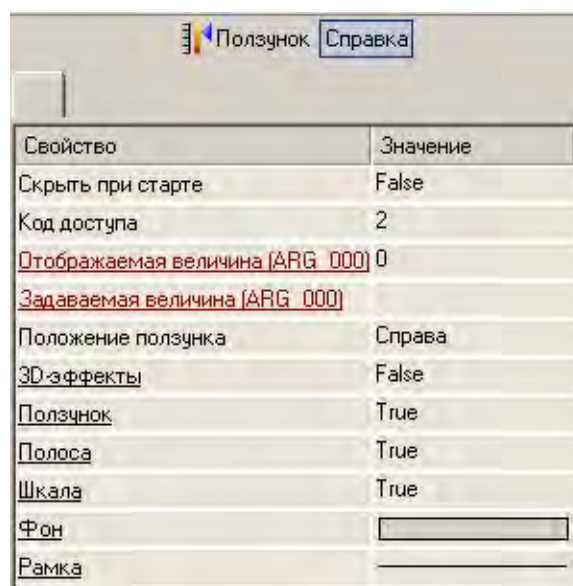


Рисунок 5.5. Вид редактор параметров настройки графического элемента «Ползунок»

В программе также создан профиль второго пользователя – «Инженер». Параметры настройки данного профиля представлены на рисунке 5.6.

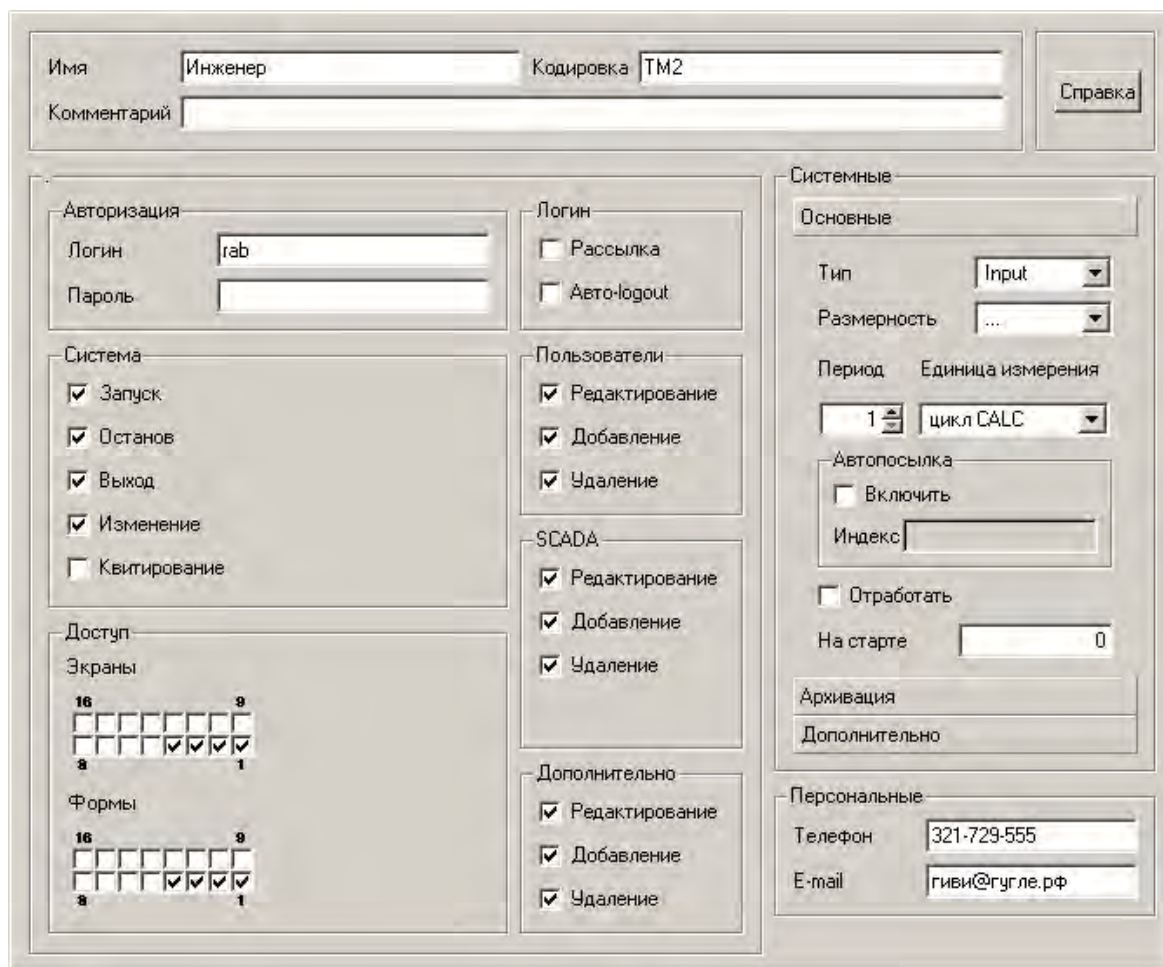


Рисунок 5.6. Вид редактор параметров настройки канала класса «Пользователь»

Следует отметить, что при разработке систем контроля доступа персонала к функциям управления в Проекте, несколько графических форм могут иметь одинаковый код доступа. Таким образом, графические элементы можно объединять в группы и производить над ними групповые операции.

Управление профилями пользователей может быть выполнено не только в режиме создания Проекта, но и во время работы МРВ. Для этого используется меню «Вид : Пользователи». Для переключения между пользователями также используется главное меню МРВ.

Для отработки и закрепления материала выполните:

- операции добавления нового пользователя и настройку прав доступа к функциям программы для него;
- исследование функции управления учётными записями пользователя в масштабе реального времени (при запущенном МРВ и Проекте).

5. 1. Вопросы для самоконтроля к главе №5

1. Опишите основные функции управления правами доступа персонала к элементам управления АСУ в Проекте?
2. Возможно ли назначить права изменения состояния группы графических элементов на графическом экране для некоторого пользователя?
3. Какой инструмент в составе РПД позволяет назначить права управления состоянием графического элемента?
4. Какие типы каналов могут быть использованы при формировании профиля нового пользователя?
5. Назовите функции контроля доступа используемые в TRACE MODE 6.06.2?
6. Возможно ли выполнять операции настройки контроля доступа персонала к функциям управления в АСУ в процессе работы МРВ?

6. АРХИВИРОВАНИЕ ТЕХНОЛОГИЧЕСКОЙ ИНФОРМАЦИИ

Архивирование технологической информации в АСУ является очень важной задачей. При возникновении аварийных ситуаций необходимо иметь полную информацию о состоянии объекта управления до момента аварии. Подробный анализ предаварийной ситуации помогает понять причины возникновения аварии и учесть факторы её вызывающие при проектировании новых систем управления и во многих других случаях. Следует понимать, что качество анализа аварийной ситуации напрямую зависит от количества информации об объекте управления и АСУ до её возникновения, но количество этой информации не может быть больше чем система управления может обработать за цикл управления.

Для архивирования данных в TRACE MODE 6.06.2 используются 5 типов архивов:

- SIAD (СПАД) – структурированный промышленный архив данных;
- ОТ – отчёт тревог;
- Глобальный регистратор (глобальный архив);
- Дамп – архив выходных значений каналов в файле на жёстком диске;
- Локальный архив в памяти.

Каждый из архивов в АСУ используется специальным образом, рассмотрим некоторые из них более детально.

6.1. Использование промышленных архивов данных в TRACE MODE 6

СПАД-архив

Данный тип архивов используется для хранения выходных значений каналов, которые поступают в архив по мере их изменения. Сообщения в архив записываются в регламентированном формате по мере их поступления. После заполнения файла архива новое сообщение добавляется вместо самого старого сообщения. Формат строки сообщения имеет следующий вид (рисунок 6.1).

<i>Дата</i>	<i>Время</i>	<i>Значение</i>	<i>Идентификатор канала</i>
26.11.2009	15:51:33.000	39.9359	

Рисунок 6.1. Формат сообщения в СПАД

Архивы SIAD имеют следующие основные характеристики:

- точность задания времени, мс – 1;
- скорость записи значений в архив (для рабочей станции с процессором Pentium-4 с тактовой частотой 2 ГГц), тыс. параметров в секунду – свыше 600;
- объём архива ограничивается только возможностями файловой системы и может измеряться гигабайтами.

Данные из СПАД-архива могут быть экспортированы в файлы на НМД, а также отображаться на дисплее. Для управления данными в реальном масштабе времени используются системные переменные из группы «Системные» и «Диагностика».

Системные переменные из группы «Системные»:

- @Copy_SIAD;
- @Data_from_SIAD;
- @Logging.

Системные переменные из группы «Диагностика»:

- @e_SIAD;
- @q_SIAD_Lost;
- @q_SIAD_Q.

Настройка СПАД выполняется в разделе характеристик Узла (рисунок 6.2).

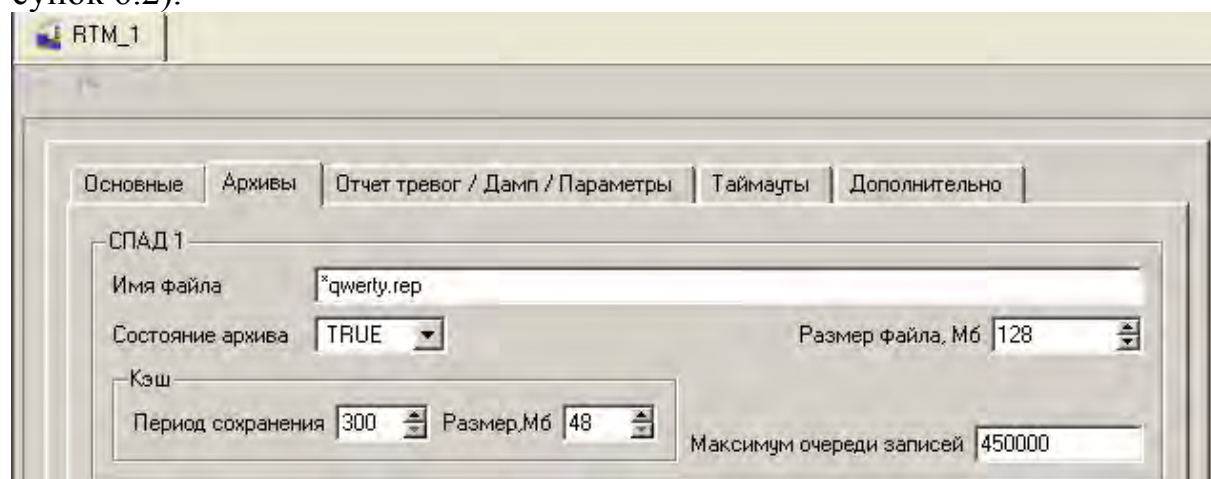


Рисунок 6.2. Вид окна редактора параметров настройки Узла Проекта

В качестве параметров необходимо указать адрес расположения и название файла СПАД, его размер, период сохранения сообщений и

размер очереди записи в ОЗУ. Затем в настройках канала, значение которого необходимо архивировать необходимо в разделе «Системные : Архивация» задать номер сконфигурированного СПАД (1, 2, 3).

Отчёт тревог

Данный тип архивов используется для хранения сообщений, генерируемых МРВ в случае срабатывания системы защиты (например, выход значения канала класса FLOAT за установленные пределы), в случае возникновения важных для работы АСУ событий (например, смена пользователя) или генерируемых оператором. Текст сообщения может быть вызван из словаря или задан значением по умолчанию. Сообщения в ОТ записываются циклически по мере его заполнения. Новые сообщения после заполнения архива добавляются вместо наиболее старых. Формат строки сообщения имеет следующий вид (рисунок 6.3)

Дата	Время	Категория сообщения	Название канала	Кодировка	...
26.11.2009	15:51:33.000	М	Синусоида#1		...
10 байт	12 байт	1 байт	32 байта	21 байт	
	Идентификатор пользователя				
Текст сообщения	квитировавшего сообщение	Время квитирования	Порядковый номер строки		
...	NORMA	gufi	13_15:51:55	30	
	48 байт	5 байт	11 байт	HEX-код 8 символов	

Рисунок 6.3. Формат сообщения в архив «Отчёт тревог»

Настройка ОТ выполняется в разделе параметров настройки Узла (рисунок 6.4).

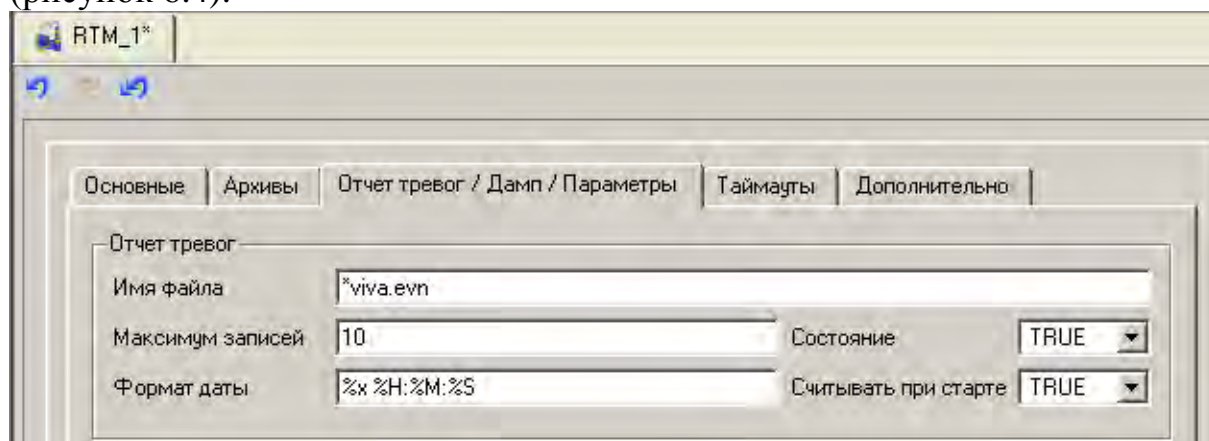


Рисунок 6.4. Вид окна редактора параметров настройки Узла Проекта

Для настройки параметров архивирования необходимо указать название и адрес расположения файла архива, максимальное количество одновременно хранящихся записей, параметр синхронизации при старте МРВ и формат даты и времени. В настройках канала также необходимо поставить флаг «Отчёт тревог» и указать адрес словаря сообщений об аварии (рисунок 6.5, 6.6).

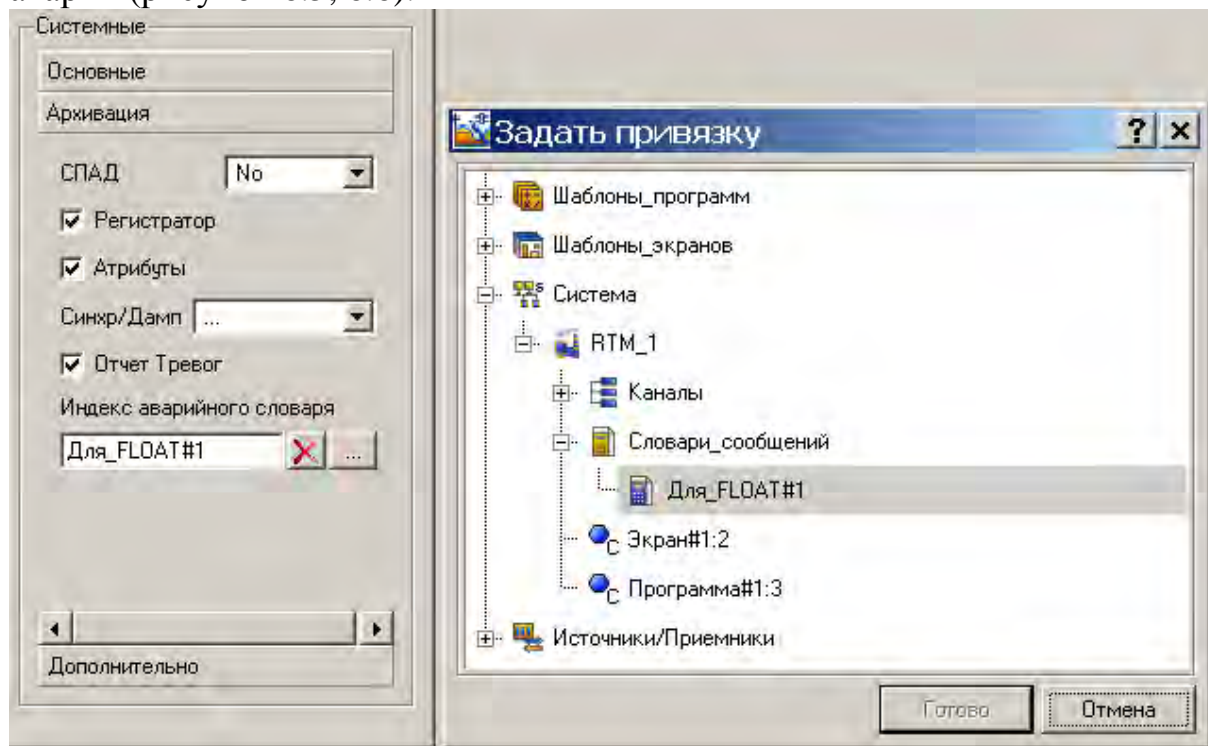


Рисунок 6.5. Вид окна редактора параметров настройки канала

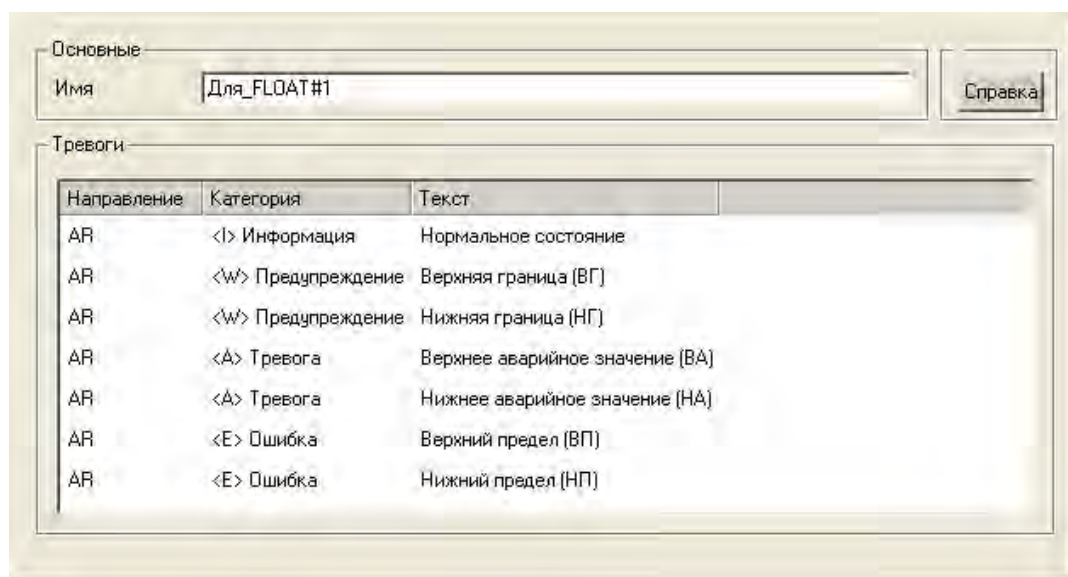


Рисунок 6.6. Вид окна редактора параметров настройки словаря сообщений об аварии

ДАМП

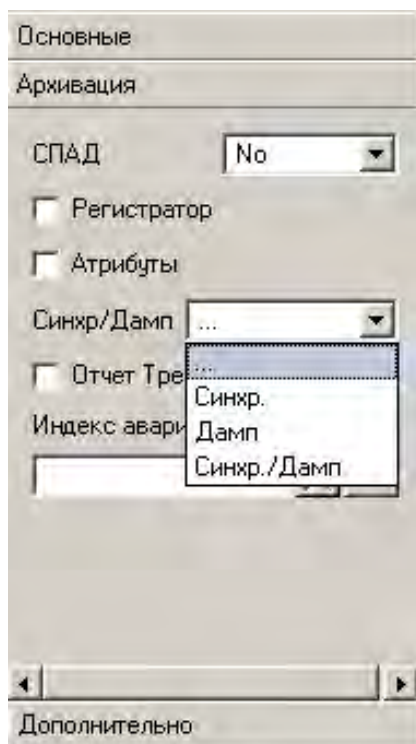


Рисунок 6.7. Вид окна редактора свойств канала

В составе Проекта может быть реализован механизм периодического сохранения данных (последнего значения канала) в ПЗУ (на НМД). Для этого используется функция «Дамп», которая настраивается в каждом канале индивидуально. Настройка выполняется во вкладке «Архивирование» (рисунок 6.7). В списке функций содержится:

- «...» (DumpSync = 0) – канал не использует Дамп и игнорируется при синхронизации резервов;
- «Синхр.» (DumpSync=1) – канал не использует Дамп, но участвует в синхронизации резервов;
- «Дамп» (DumpSync=2) – канал использует Дамп (чтение при старте/запись), но игнорируется при синхронизации резервов;
- «Синхр./Дамп» (DumpSync=3) – канал использует Дамп и участвует в синхронизации резервов.

Кроме настройки функций синхронизации и работы с Дампом необходимо настроить параметры Узла в Проекте. Для этого используется вкладка «Отчёт тревог/Дамп/Параметры». Вид редактора параметров настройки Узла представлен на рисунке 6.8.

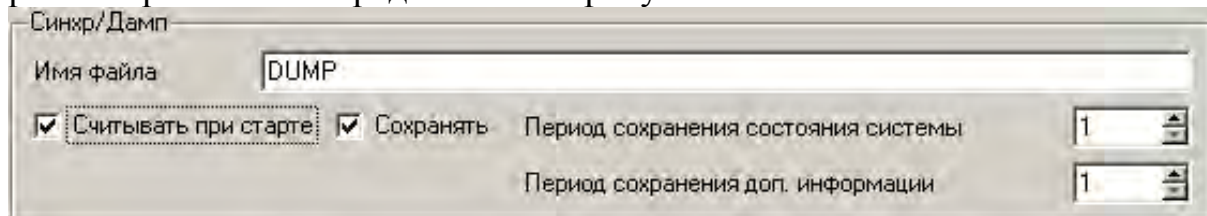


Рисунок 6.8. Вид редактора параметров настройки узла

Во вкладке «Имя файла» указывается путь к файлу Дампа и название файла или только имя файла Дампа. Файл может иметь произвольное расширение или не иметь его вовсе. Если путь к файлу не указан и перед названием файла присутствует знак «*», то файл будет создан в

текущей директории Проекта. Для функции «Дамп» также настраиваются параметры:

- считывать значение из файла Дампа при старте МРВ;
- разрешить сохранять значения в файл Дампа;
- период сохранения состояний каналов, в сек., в диапазоне (1-255).

Период сохранения дополнительной информации указан в мин. Если значение данного параметра равно 1. Сохранение информации выполняется однократно при штатном завершении работы. Если значение данного параметра равно: «2», «3» и т.д., то период сохранения будет равен: 1, 2 мин. и т.д. соответственно.

6.2. Использование системных переменных для управления данными в архивах

В отличие от ОТ, СПАД-архив хранится в бинарном файле. Следовательно, просмотреть его состав встроенными средствами ОС не представляется возможным. Для управления данными этого архива могут быть использованы системные переменные TRACE MODE. Рассмотрим некоторые из них.

@Data from SIAD

Данная переменная типа OUTPUT используется для импорта данных архива в текстовый файл. Атрибут «Параметр» этой системной переменной указывает номер архива: N (0 – System, 1 – SIAD1, 2 – SIAD2, 3 – SIAD3). Код функции, которую необходимо выполнить для управления данными архива, формируется словом данных. Управляющее слово имеет следующий формат (рисунок 6.9) и посылается на вход канала класса FLOAT вызывающего системную переменную в десятичном виде.

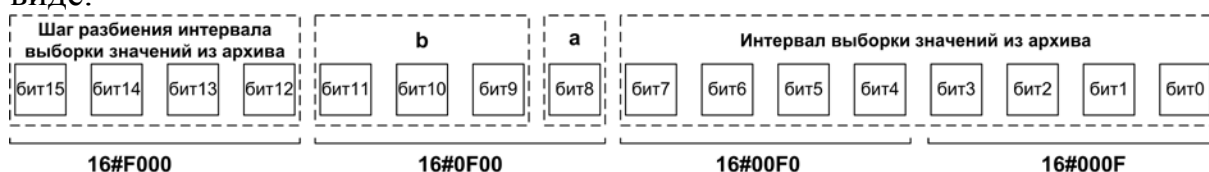


Рисунок 6.9. Формат команды управления выборкой данных из СПАД с использованием системной переменной «@Data_from_SIAD»

Первый байт управляющего слова задаёт интервал выборки данных. Например, 0 – соответствует операции выборки данных за последний час, 14 – за предыдущие сутки, 16 – за предыдущую неделю. Вось-

мой бит (бит a) определяет объём выборки. Например, если «a = 0» – выполнять извлечение всех имеющихся данных, если «a = 1» – в файл извлекаются срезы по (интервал/шаг) строк. Биты b третьего полубайта управляющего слова конфигурируют способ организации извлечённых данных на НМД. Например, при «b = 0», данные сохраняются в один файл с названием «said». При «b = 1» данные из каждого канала Проекта записываются в отдельные файлы. В качестве названия файла используется название канала Проекта. Более подробно изучить формат управляющего слова можно при помощи программы-примера в рамках лабораторных работ.

Следует отметить, что выборка данных из СПАД в Профайлере базовой среды разработки ограничена выборкой за предыдущий час. Следовательно, использовать выборку-срез по времени не получится.

Управление данными архива также возможно в интерактивном виде при помощи меню «Действия» MPB.

@Copy_SIAD

Данная системная переменная используется для создания резервной копии текущего СПАД. Значение атрибута «Параметр» данной системной переменной, как и ранее, используется для задания номера архива, с которым необходимо работать. Для формирования названия и адреса в файловой системе архива используется входное значение данной переменной. Например, код «1» соответствует операции сохранения данных архива в текущую директорию ОС. В качестве названия файла используется название канала, данные которого архивировались, код «3» соответствует операции сохранения архива с именем в формате «ч_д_м_г». Значения ≥ 65 интерпретируются как ANSI-коды (например, 65 – это прописная латинская буква A). Сформировать точный адрес в файловой системе можно, подключив некоторую удалённую или локальную папку в качестве разделяемого ресурса (сетового диска), тогда, указав ANSI-код буквы данного диска в качестве входного значения системной переменной «@Copy_SIAD», автоматически будет назначен путь к месту хранения резервного СПАД.

Особенности использования системной переменной «@Copy_SIAD»

По результатам практического применения данного механизма были отмечены некоторые особенности. Так, например, при создании непрерывного архива методом копирования через определённые интервалы времени, наблюдается временной разрыв. Следовательно, для прак-

тического применения необходимо делать архивы с небольшим временем дублирования на стыках.

Управление архивами также возможно в интерактивном виде при помощи меню «Действия» МРВ.

6.3. Вопросы для самоконтроля к главе №6

1. Назовите основные типы промышленных архивов данных используемых в TRACE MODE 6.06.2?
2. Что такое СПАД? Дайте определение (назначение, основные характеристики), полное название.
3. Какие операции можно выполнять со СПАД при помощи системных переменных TRACE MODE?
4. Что происходит после переполнения файла архива?
5. Что такое ОТ?
6. Какие операции можно выполнять с ОТ?
7. Назовите способы копирования СПАД?
8. Опишите способ извлечения данных из СПАД?
9. Что такое Дамп?
10. Опишите основные операции настройки промышленных архивов в TRACE MODE?

7. ЗАПУСК НА ВЫПОЛНЕНИЕ ГОТОВЫХ ПРОЕКТОВ

После разработки Проекта необходимо сформировать привычный для оператора механизм запуска нового программного приложения в рамках операционной системы. Для этого используется ярлык, который посредством специальных ключей позволит запустить на выполнение МРВ, открыть файл Проекта, перевести МРВ в полноэкранный режим. Рассмотрим ключи, которые выполняют данные функции.

- **«/run»** (Ctrl + R) – ключ используется для автоматического запуска на выполнение МРВ.
- **«/fullscreen»** (Ctrl + F) – ключ используется для перевода МРВ в полноэкранный режим.
- **«/hidemenu»** (Ctrl + T) – ключ используется для того, чтобы скрыть консоль кнопок с пиктограммами.

Таким образом, полная строка запуска на выполнение МРВ будет выглядеть так: «[Path_RTM/rtc.exe] [Path_name.dbb/name.dbb] /run /fullscreen /hidemenu», где rtc.exe – запускаемый файл МРВ, name.dbb – основной файл Проекта.

8. ВЗАИМОДЕЙСТВИЕ С УСО В ОС MS WINDOWS И DOS

Использование микропроцессорной техники в настоящее время значительно расширяет область применения методов математической обработки данных. Использование сложных математических моделей в составе контроллеров АСУ позволяет сократить издержки на проектирование и отладку законов управления элементами АСУ и моделирование штатных и непредвиденных ситуаций. Однако самостоятельно микропроцессорные контроллеры не позволяют сформировать управляющий сигнал для исполнительных элементов АСУ. Эти задачи в составе ЭВМ выполняют устройства связи с объектом управления (УСО).

УСО могут быть расположены в составе микропроцессорных контроллеров или ПЭВМ, а также могут быть внешними и подключаться к управляющей ЭВМ посредством периферийных интерфейсов (например, USB). В том или ином случае УСО выступает в качестве посредника-преобразователя физического параметра ОУ в цифровые (информационные) сигналы понятные ЭВМ и наоборот. Независимо от принципа взаимодействия микропроцессорного контроллера с элементами УСО программное обеспечение в составе данного контроллера должно уметь работать с используемым УСО. Поскольку программным модулем, отвечающим за взаимодействие с внешними элементами системы является драйвер, то количество драйверов различных периферийных устройств в составе среды разработки прикладного программного обеспечения определяет информационную мощность будущего Проекта прикладного ПО и удельную максимальную мощность системы управления, в которой он будет применяться.

В этой главе рассмотрим несколько классических примеров исполнения микропроцессорных контроллеров с электронными платами УСО применяемых в составе АСУ: микропроцессорный контроллер под управлением DOS и микропроцессорный контроллер под управлением ОС MS Windows. Разработка программного обеспечения будет выполняться средствами САПР TRACE MODE 6.06.2. Для отработки описанный в данной главе функций используйте лабораторный комплекс. Описание лабораторного стенда представлено в п.п. 8.5. Для ввода дискретных данных в его составе используется кнопочный блок ввода команд управления и индикации сигналов контроля, подключенный к стенду при помощи УСО. Сетевое взаимодействие может быть настроено между несколькими рабочими станциями лаборатории, поскольку они объединены в локальную вычислительную сеть.

8.1. Принципы разработки и структура Проекта создаваемого в TRACE MODE 6.06.2 для микропроцессорного контроллера с платами УСО

Принципиальным отличием Проектов предназначенных для контроллеров с УСО от других, является наличие дополнительных компонентов настраиваемых в слое «Источники/Приёмники». Рассмотрим рисунок 8.1.

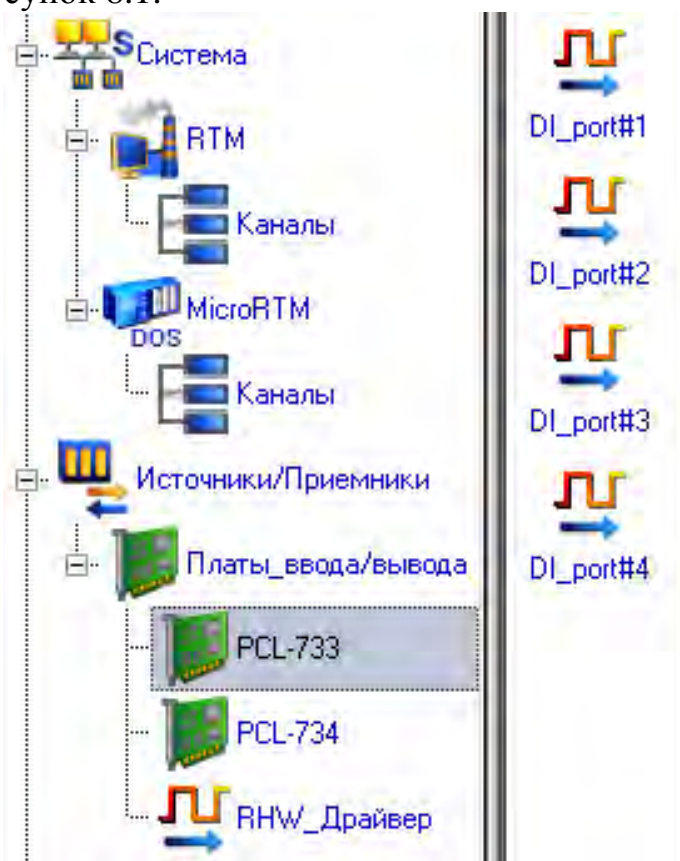


Рисунок 8.1. Вид окна программы «Навигатор проекта»

«PCL-733» – Discrete Input card или «ADVANTECH PCL-734» – Discrete Output card). В том случае, когда необходимого драйвера не существует, он может быть разработан самостоятельно на языке программирования C++. Для использования уникального драйвера в составе Проекта необходимо включить в состав слоя «Платы ввода/вывода» RHW-драйвер.

Рассмотрим структуру Проекта с использованием драйверов УСО на примере рисунков 8.2, 8.3. Пусть имеется плата ввода дискретных сигналов в составе некоторого микропроцессорного контроллера. Плата имеет 32 входа: DI-0 ... DI-31. В составе Проекта инициализируется соответствующий типу и названию платы драйвер. Автоматически в составе слоя с названием драйвера создаются информационные порты в

Поскольку микропроцессорный контроллер может быть создан на базе полноформатной ЭВМ под управлением ОС MS Windows или одноплатной ЭВМ под управлением DOS или MS Windows CE, в составе Проекта должны быть созданы Узлы RTM или MicroRTM соответственно. В составе слоя «Источники/Приёмники» необходимо инициализировать один из видов УСО (например, «Платы ввода/вывода»). Наличие встроенных бесплатных драйверов в составе интегрированной среды разработки позволяет выбрать одну из плат УСО (например, «ADVANTECH

количестве 4-х штук. В составе каждого порта имеется восемь регистров группирующих на выходе порта последовательность из восьми бит данных. Состояние каждого регистра характеризует состояние соответствующего входа платы УСО. Для того чтобы в составе Проекта можно было использовать значения, получаемые на выходе драйвера УСО, необходимо настроить порты драйвера (данная операция будет рассмотрена далее) и выполнить их автопривязку (методом «Drag and Drop») ко входам каналов в составе слоя «Система» (для плат ввода сигналов) или входов портов драйвера к выходам каналов в слое «Система» (для плат вывода сигналов из ЭВМ). Поскольку работать с вектором данных в составе Проекта как с группой битов затруднительно, необходимо использовать некоторую программу (например, FBD) разделяющую вектор данных на составляющие. Каждый из выделенных битов в дальнейшем может быть обработан специальным образом или представлен на экране дисплея. В случае использования платы вывода дискретных сигналов необходимо использовать программу упаковки битов в вектор (байт) данных.

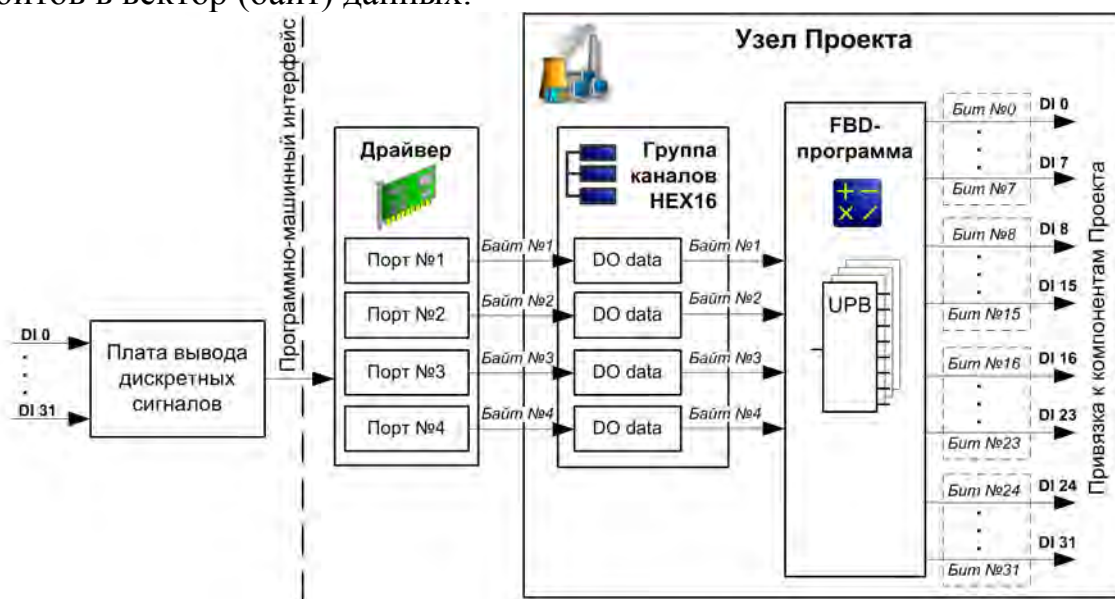


Рисунок 8.2. Структура Проекта с использованием драйвера платы ввода дискретных сигналов

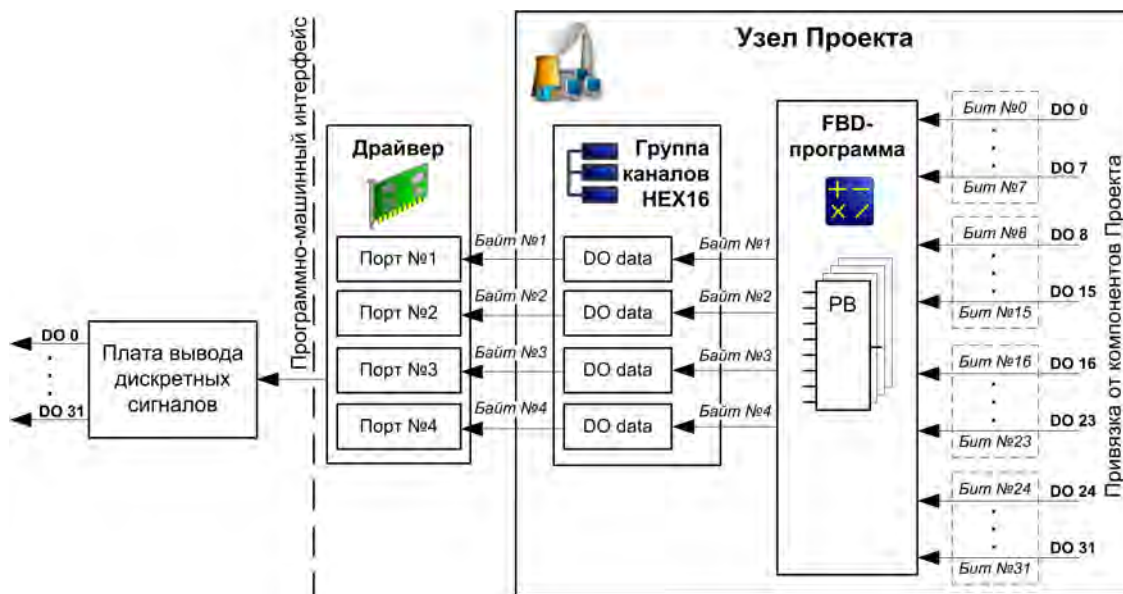


Рисунок 8.3. Структура Проекта с использованием драйвера платы вывода дискретных сигналов

В том случае, когда в составе микропроцессорного контроллера используется модуль аналогового ввода или вывода, компьютерная программа распаковки и упаковки значений не применяется, а каналы передачи аналогового сигнала настраиваются специальным образом (например, масштабирование или смещение сигнала по амплитуде).

8.2. Настройка драйвера в составе Проекта

Стандартный драйвер платы УСО

Стандартный драйвер платы УСО настраивается в специальном редакторе параметров его настройки (рисунок 8.4).

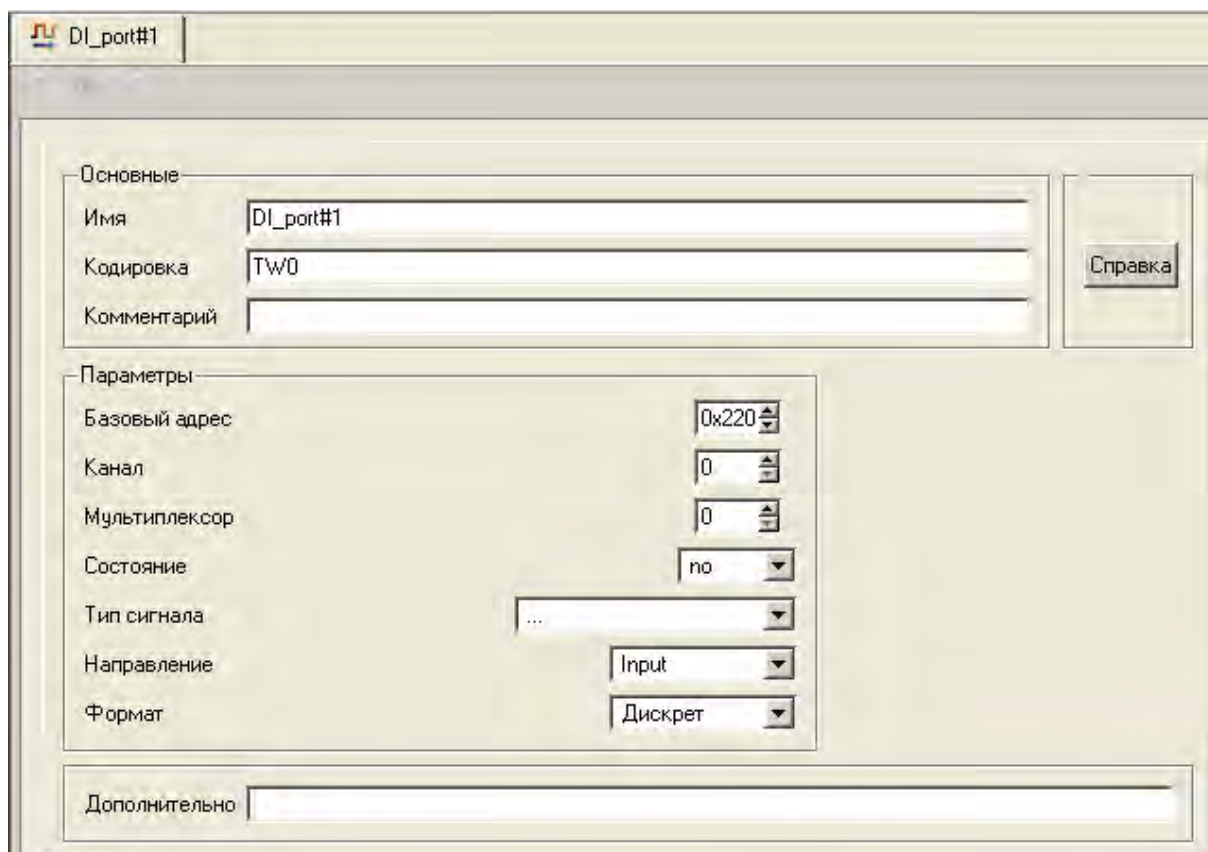


Рисунок 8.4. Вид окна редактора параметров настройки драйвера

Состав полей редактора может быть различным в зависимости от назначения УСО. В этот состав могут входить поля настройки следующих параметров:

1. имя – название порта или канала платы УСО;
2. кодировка – уникальная кодировка компонента Проекта (задаётся автоматически);
3. комментарий – поясняющий текст, в произвольной форме, предназначенный для текущего элемента Проекта;
4. базовый адрес – адрес на локальной шине, начиная с которого УСО выделяется рабочее адресное пространство;
5. канал – смещение от базового адреса для каждого порта драйвера (настраивается автоматически при инициализации связи с драйвером);
6. мультиплексор – номер канала мультиплексора (если используется);
7. тип сигнала – диапазон изменения сигнала в физических единицах измерения (например, 4-20 мА, 1-5 В);
8. направление – направление передачи данных (в/из ЭВМ);

9. формат – формат передаваемых данных (дискретный или аналоговый).

Драйвер RWH

При необходимости использовать УСО, драйвер для которого не существует в стандартной библиотеке TRACE MODE, можно разработать собственный драйвер на языке C++, оформить его в виде библиотеки RWH и добавить в состав Проекта. Методы использования RWH-драйвера не отличаются от методов использования встроенных драйверов. В составе слоя RWH-драйвер необходимо создать достаточное количество портов ввода или вывода сигналов, а затем настроить их в соответствии со структурой данного драйвера.

В составе среды разработки TRACE MODE 6.06.2 также поддерживаются драйверы типа t13.dll, которые использовались в более ранних версиях TRACE MODE. Однако разработчики не рекомендуют использовать данную технологию в новых проектах ПО.

Вопросы создания RWH драйвера в рамках данного пособия рассматриваться не будут.

Драйвер терминала в DOS

Для проектирования средств отображения информации в DOS в составе TRACE MODE 6.06.2 используется драйвер терминала. Компонент «Терминал» в TRACE MODE инициализируется в слое «Источники/Приёмники» в подслое «Терминалы». Данный компонент должен быть вызван каналом класса CALL типа Input с типом вызова «Мнемо» из Узла MicroRTM.

Настройка компонента «Терминал»

В настройках данного компонента необходимо задать значения параметров «Параметр» и «Размер строки».

«Параметр» определяет количество аргументов в канале класса CALL устанавливающих параметры отображения мнемознака (например, 4), «Размер строки» определяет максимальное количество мнемознаков в строке на экране (например, 80).

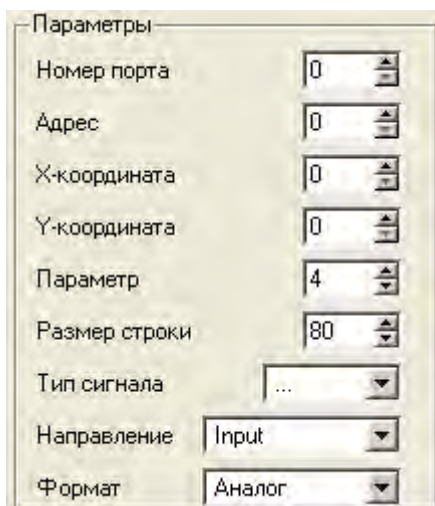


Рисунок 8.5. Вид окна программы-настройщика параметров компонента «Терминал»

Настройка параметров отображения данных

Далее в канале класса CALL, вызывающем компонент «Терминал», должны быть созданы группы аргументов, значения которых выполняют конфигурацию параметров отображения текстовых и числовых знаков на экране терминала. Структура аргументов следующая:

- Arg000 – задаёт позицию начальной строки отображения данных (тип данных int);
- Arg001 – задаёт количество строк на экране (тип данных int);
- Arg002 – зарезервирован, не используется.

Далее аргументы формируются группами для каждого отображаемого параметра в следующем порядке:

- Arg003 – номер позиции знакоместа отображаемых данных по оси X (тип данных UDINT);
- Arg004 – номер позиции знакоместа отображаемых данных по оси Y (тип данных UDINT);
- Arg005 – задаёт цвет отображаемого параметра в соответствии с таблицей кодов (таблица 8.1) (тип данных INT);
- Arg006 – задаёт название отображаемого параметра как название канала, к которому данный аргумент привязан. Внутреннее значение Канала используется как значение отображаемого параметра (тип данных REAL).

Далее блоки аргументов (по 4 шт. – определяется параметром «Параметр» компонента «Терминал») повторяются в зависимости от количества отображаемых параметров на экране. Значения аргументов Arg003 – Arg006 в канале класса CALL, задаются в шестнадцатеричном

формате («16#001» или «16#00A»). Цвет отображаемого параметра задается в формате «16#ху», где х – цвет фона отображаемого параметра, у – цвет текста отображаемого параметра. Таблица кодирования цвета представлена ниже.

Таблица 8.1

Таблица кодов цвета отображаемого на терминале аргумента

Число	Цвет	Число	Цвет
0	Black (черный)	8	Gray (серый)
1	Blue (синий)	9	Light Blue (светло-синий)
2	Green (зеленый)	A	Light Green (светло-зеленый)
3	Cyan (голубой)	B	Light Cyan (светло-голубой)
4	Red (красный)	C	Light Red (светло-красный)
5	Magenta (фиолетовый)	D	Light Magenta (светло-фиолетовый)
6	Yellow (желтый)	E	Light Yellow (светло-желтый)
7	White (белый)	F	Bright White (ярко-белый)

Структура Проекта с использованием компонента «Терминал» представлена на рисунок 8.6.

В Проекте присутствуют компоненты драйверов плат УСО, каналы передачи и преобразования потоков данных, а также компонент «Терминал». Цифрами на круглой подложке обозначена последовательность операций, выполняемых в Проекте. Рассмотрим информационный канал ввода дискретных сигналов при помощи платы УСО PCL-733.

От компонента драйвера платы УСО PCL-733 (рисунок 8.6) данные пакетами по одному байту поступают в каналы Проекта (шаг 1). Далее канал класса CALL вызывает компонент Проекта «Программа» из слоя «Шаблоны_программ» (шаг 2). Компьютерная программа забирает значения входных аргументов (шаг 3) и преобразует их в биты (распаковка байта в биты, шаг 4). После выполнения программы результат помещается в выходные аргументы компонента «Программа» связанные с каналами класса FLOAT в узле Проекта (шаг 5). Канал класса CALL, вызывающий компонент «Терминал», который содержит в себе группы аргументов описанных выше. В аргументы этого канала, привязанные к каналам класса FLOAT, в которых содержатся обработанные значения, посылаются данные (шаг 6). Название этих каналов используются компонентом «Терминал» для обозначения отображаемых параметров, а внутренние значения каналов – в качестве значений отображаемых параметров (рисунок 8.7).

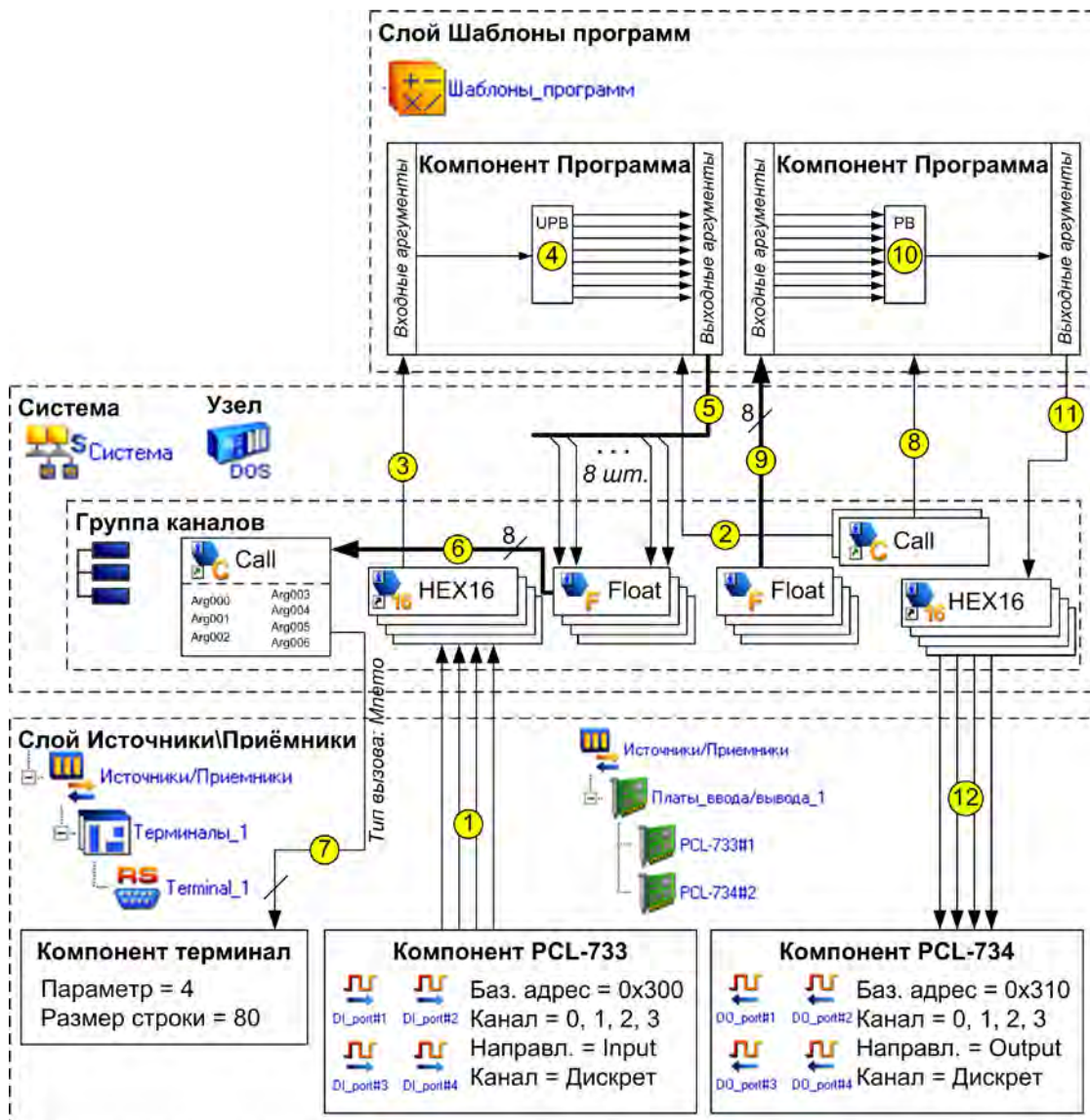


Рисунок 8.6. Структура Проекта, использующего компонент «Терминал». Цифрами указана последовательность операции обработки передачи данных


```
ADVANTECH PCL-733

DI_1 0      DI_9 0      DI_17 0     DI_25 0
DI_2 0      DI_10 0     DI_18 0     DI_26 0
DI_3 1      DI_11 0     DI_19 0     DI_27 0
DI_4 0      DI_12 0     DI_20 1     DI_28 0
DI_5 1      DI_13 0     DI_21 0     DI_29 0
DI_6 0      DI_14 1     DI_22 0     DI_30 0
DI_7 0      DI_15 0     DI_23 0     DI_31 0
DI_8 0      DI_16 0     DI_24 0     DI_32 0
```

Рисунок 8.7. Снимок экрана, сделанный при запуске на выполнение Проекта в микропроцессорном контроллере, работающем под управлением DOS

На рисунке 8.7 представлен снимок экрана, сформированный компонентом «Terminal» в DOS. Название параметра – «DI_1» определяется названием канала, из которого берётся отображаемое на экране значение. Каждое знакоместо сконфигурировано при помощи групп аргументов канала CALL.

8.3. УСО в распределённых АСУ

АСУ могут быть значительно распределёнными и содержать множество различных узлов, каждый из которых выполняет сбор данных. Часто обработку данных выполняют в рабочих станциях и микропроцессорных контроллерах верхнего уровня (рисунок 8.8). В составе распределённой АСУ также могут находиться несколько рабочих станций, для которых очень важно обмениваться некоторыми данными (например, рабочая станция гл. инженера и управляющая ЭВМ). Обмен данными между такими узлами АСУ может быть эффективно выполнен с использованием интерфейса Ethernet. Рассмотрим структуру Проекта в котором выполняется сетевой обмен между Узлами и принципы настройки сетевого обмена данными.

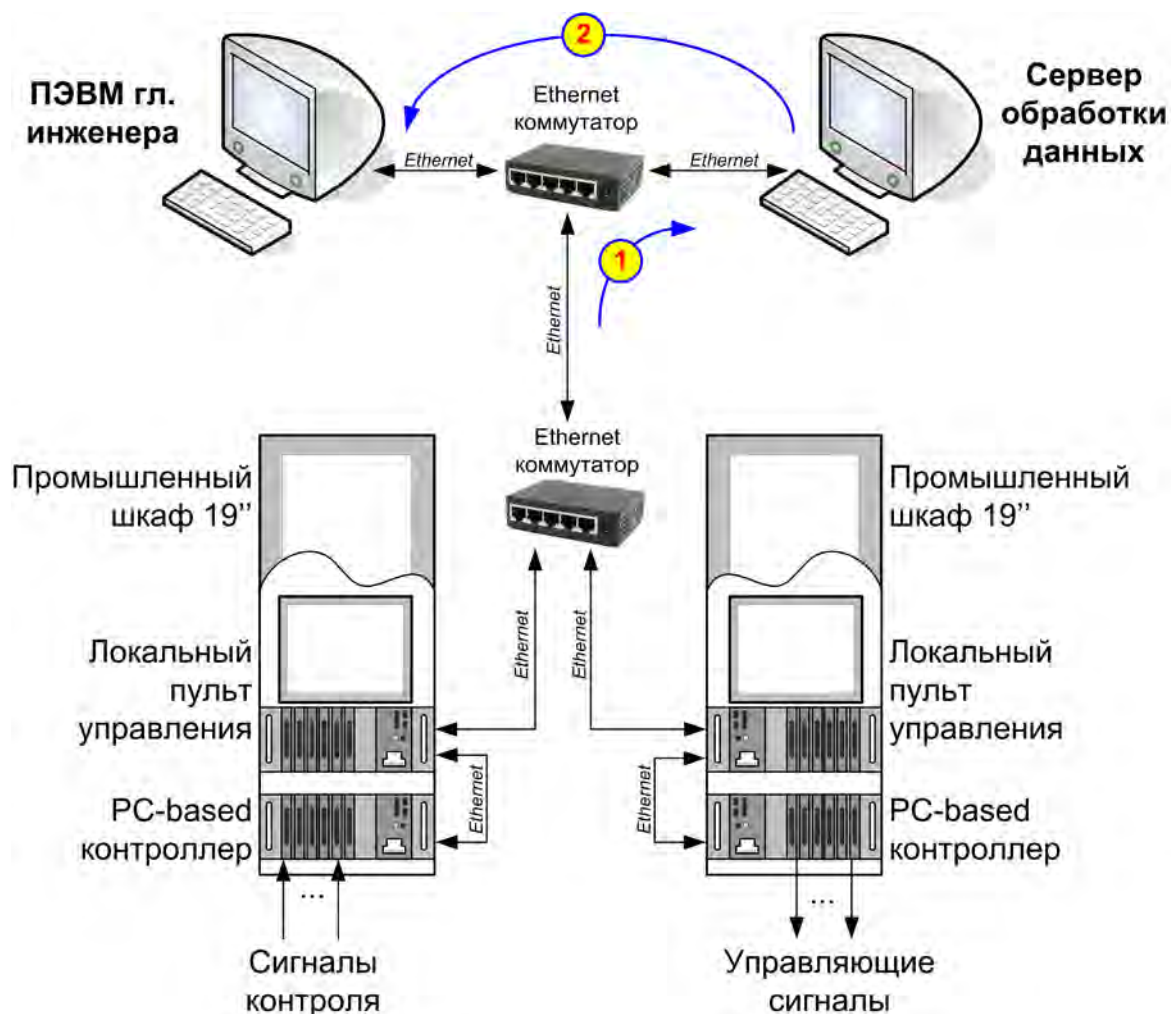


Рисунок 8.8. Структура АСУ в составе которой используется сетевой обмен данными: 1 – передача данных от устройств сбора данных; 2 – передача информации о состоянии производства в экономический отдел для расчёта коммерческих характеристик производственной линии

Настройка сетевого обмена данными

На рисунке 8.8 представлена группа узлов АСУ, каждый из которых содержит один или несколько микропроцессорных модулей. Пусть для каждого из узлов АСУ создаётся Узел Проекта, для которого должен быть настроен сетевой интерфейс. При настройке сетевого взаимодействия в TRACE MODE 6 по интерфейсу Ethernet необходимо выполнить следующие действия:

1. настроить IP-адрес для каждого Узла (рисунок 8.9);
2. выбрать сетевой контроллер, используемый для обмена с другими Узлами Проекта;
3. настроить индивидуальный и групповой адрес Узла в Проекте;
4. выполнить привязку каналов между Узлами Проекта.

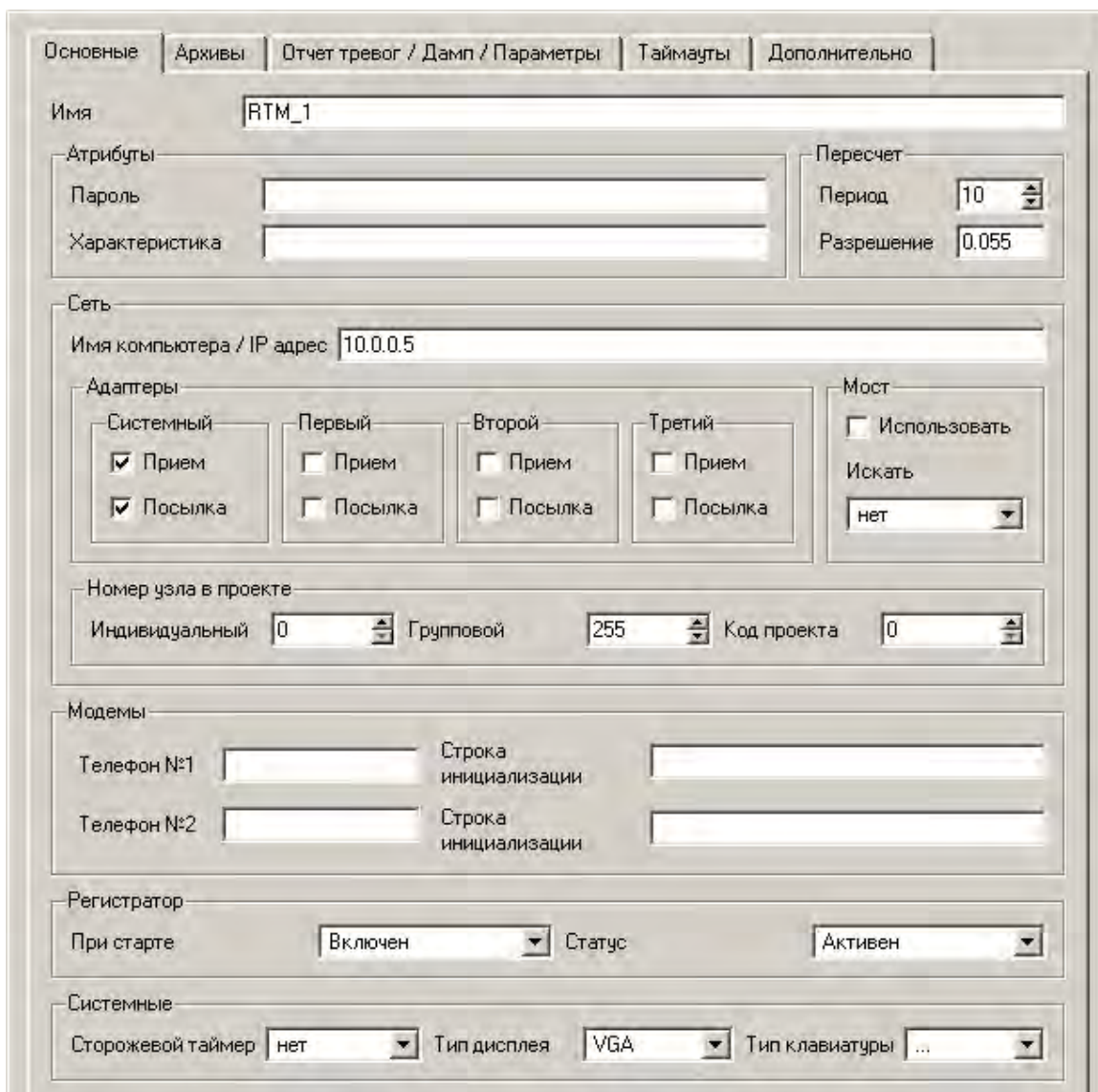


Рисунок 8.9. Вид окна программы «Редактор параметров узла»

Следует отметить, что MPB может определить IP-адрес самостоятельно и назначить его Узлу, обмен данными при этом будет осуществляться по уникальному индивидуальному и групповому адресу Узла, но указывать IP-адрес явным образом рекомендуется. Важно правильно определить сетевой адаптер и указать разрешённые направления передачи данных. Для ОС MS Windows XP рекомендуется указывать «Системный», а для DOS или ОС MS Windows CE «Первый», «Второй» или «Третий». Индивидуальный и групповой адрес Узла устанавливается автоматически, но может быть в дальнейшем настроен.

Привязку каналов Узлов, для которых настраивается сетевой обмен, можно выполнить методом «Drag and Drop» (перетяните канал из

одного Узла в другой, автоматически настроится канал передачи данных по Ethernet). Дополнительной настройки каналы не требуют. При этом тип канала указывает направление передачи данных. Схема сетевого взаимодействия Узлов Проекта представлена на рисунке 8.10.

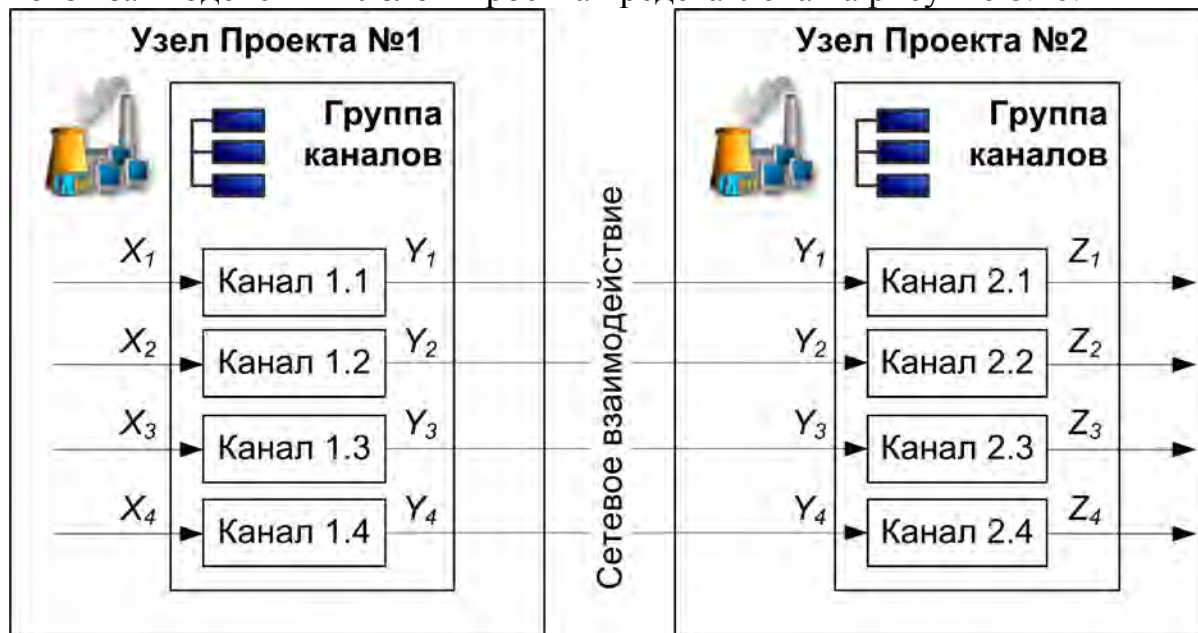


Рисунок 8.10. Схема сетевого взаимодействия Узлов Проекта

8.4. Элементы индикации и управления, используемые в составе интерфейса пользователя при работе с УСО

Индикация состояний дискретных сигналов



Рисунок 8.11. Способы индикации состояния входов устройства ввода дискретных сигналов

Индикация состояний входов платы ввода дискретных сигналов на этапе наладки или в рабочем Проекте может быть выполнена, например, цветом, текстом или динамическим элементом (рисунок 8.11). Следует помнить, что нормативными документами² устанавливаются определённые цветовые индикаторы для выполнения различных функций.

Например, ошибка или закрытое состояние должны обозначаться красным цветом, а правильная работа прибора или включенное состоя-

² ГОСТ Р МЭК 60073-2000. Интерфейс человекo-машинный. Маркировка и обозначения органов управления и контрольных устройств.

ние обозначается зелёным. В TRACE MODE в качестве индикатора может использоваться любой графический объект, для которого далее необходимо настроить необходимую индикацию. Рассмотрим способы настройки индикатора на примере графического объекта «Текст» (рисунок 8.12).

Настройки графического объекта «Текст», представленные на рисунке 8.12 запрограммированы следующим образом:

- заливка фона графического объекта выполняется динамически в зависимости от равенства значений аргумента «z1» и константы «1». В случае их равенства цвет заливки – зелёный, в противном случае – красный;
- цвет текста также изменяется динамически в зависимости от выполнения равенства значения аргумента «y1» и значения константы «1». Когда равенство выполняется, цвет текста будет зелёным, в обратном случае цвет текста будет красным.

При разработке графических элементов ввода команд управления необходимо учитывать, что данный элемент может быть использован как для ввода команд, так и для индикации результатов выполнения этих команд. В этом случае элемент управления должен иметь три состояния индикации, поскольку при работе с ним потребуется выполнить индикацию ситуации, когда команда принята, и ситуации когда команд выполнена. Для подтверждения ввода команды может быть использован синий цвет индикатора, а для подтверждения её выполнения может использоваться сигнал управляемого технологического элемента или косвенный параметр и зелёный цвет индикатора.

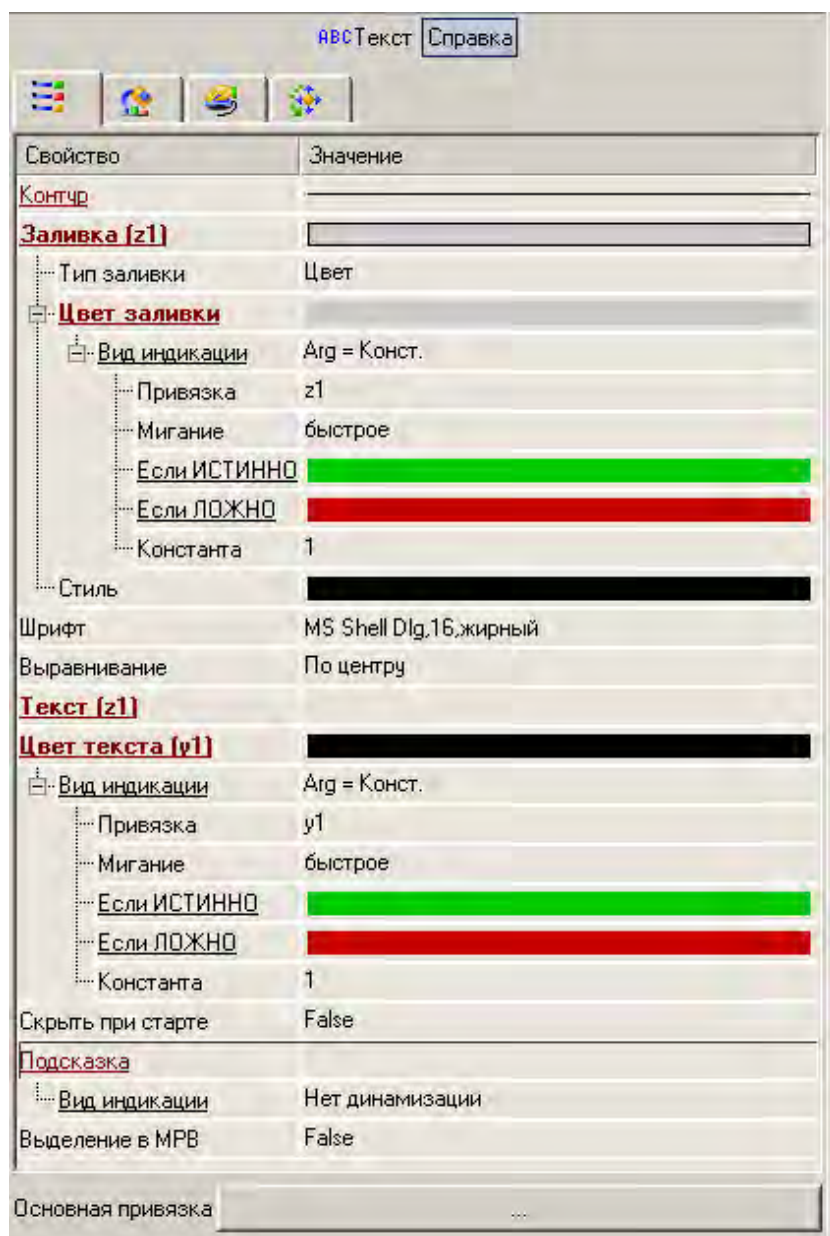


Рисунок 8.12. Вид окна программы «Редактор настройки параметров графического объекта “Текст”»

Индикация значений аналоговых сигналов

Индикация состояний входов устройства ввода аналоговых сигналов несколько отличается от индикации описанной выше. При отображении текущего значения необходимо настроить размер, тип и цвет шрифта, для того, чтобы восприятие информации было максимально быстрым и адекватным. Необходимо, также, настроить формат отображаемого числа и его точность (используйте тип данных FLOAT). Для отображения аналогового сигнала могут быть использованы виртуаль-

ные приборы или графики, а также графические элементы с динамической заливкой. Для повышения скорости поиска нужного отображаемого параметра в некоторых случаях необходимо использовать функцию перемещения графического индикатора по экрану дисплея или его выделение цветом.

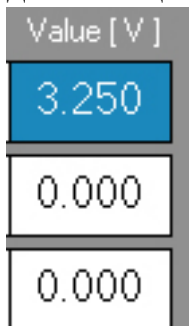


Рисунок 8.13. Изображение текстового индикатора значения аналогового сигнала с цветовой сигнализацией того, что значение отлично от нуля

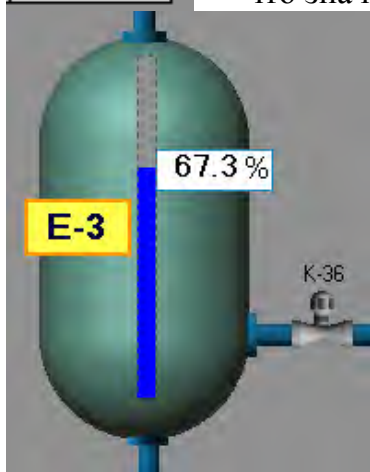


Рисунок 8.14. Изображение текстового индикатора значения аналогового сигнала с функцией перемещения по вертикали в зависимости от уровня жидкости в баке

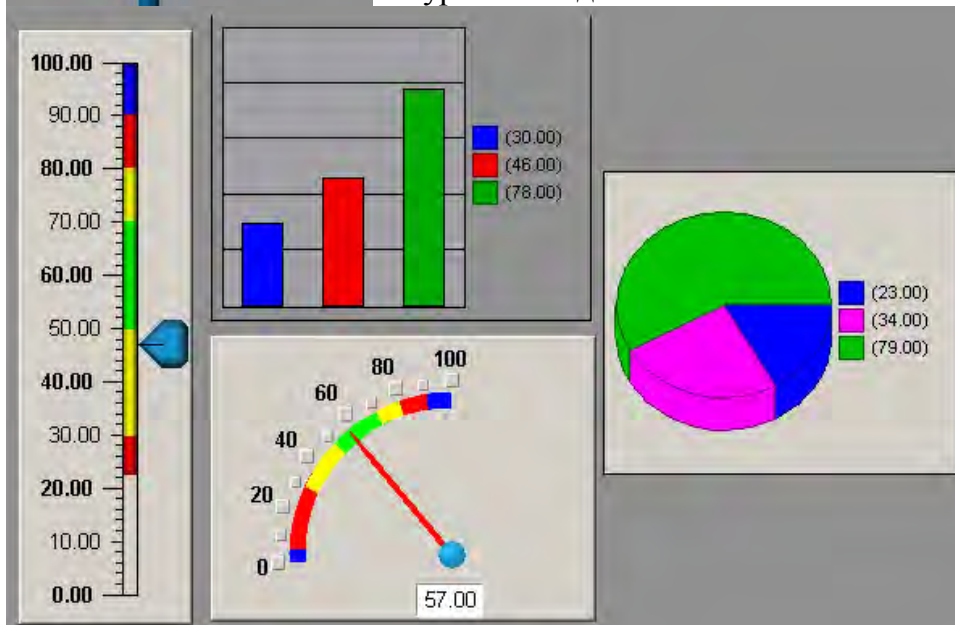


Рисунок 8.15. Изображение графического элемента «Задатчик уставки» типа «Ползунок» (слева) и изображения графических индикаторов значения аналогового сигнала справа и по центру

Представленный на рисунке 8.13 тип графического индикатора позволяет с точностью до 3-го знака после запятой отобразить значение аналогового сигнала в единицах измерения вольт, а также предупредить оператора о том, что значение, отображаемое в графической форме, отличается от нуля.

Представленный на рисунке 8.14 индикатор состоит из 2-х частей: полосы с динамической заливкой по вертикали и текстового индикатора, который перемещается горизонтально в зависимости от текущего значения в информационном канале. Полоса с динамической заливкой позволяет визуально выполнить оперативную оценку уровня жидкости в баке. В том случае, когда уровень жидкости в баке понижается или повышается до нижнего или верхнего предела оператор может детально оценить его значение при помощи текстового индикатора.

Представленные на рисунке 8.15 графические элементы управления и отображения значения аналогового сигнала демонстрируют основные возможности TRACE MODE 6.06.2. Стрелочный прибор и «Ползунок» имитируют реальные приборы отображения и генерации значений аналогового сигнала. Гистограмма может быть использована для отображения группы функционально объединённых параметров (например, температуры в нескольких зонах трубы – распределение температуры по длине трубы).

Необходимо отметить, что существуют системы, ввод значения аналогового сигнала в которых должен быть точным (например, уставка температуры в химическом реакторе). Для выполнения подобной задачи необходимо комбинировать графический элемент «Ползунок» с кнопками добавления и вычитания процента шкалы диапазона аналогового сигнала. Рассмотрим некоторые методы и элементы отображения и ввода значений аналогового сигнала на примере канала типа Output, по которому передаётся аналоговый сигнал (рисунок 8.17). Структура канала класса FLOAT типа Output по которому может быть передан и в котором может быть обработан аналоговый сигнал была рассмотрена ранее в главе №2. В канале Проекта-примера выполнены следующие настройки (рисунок 8.16).

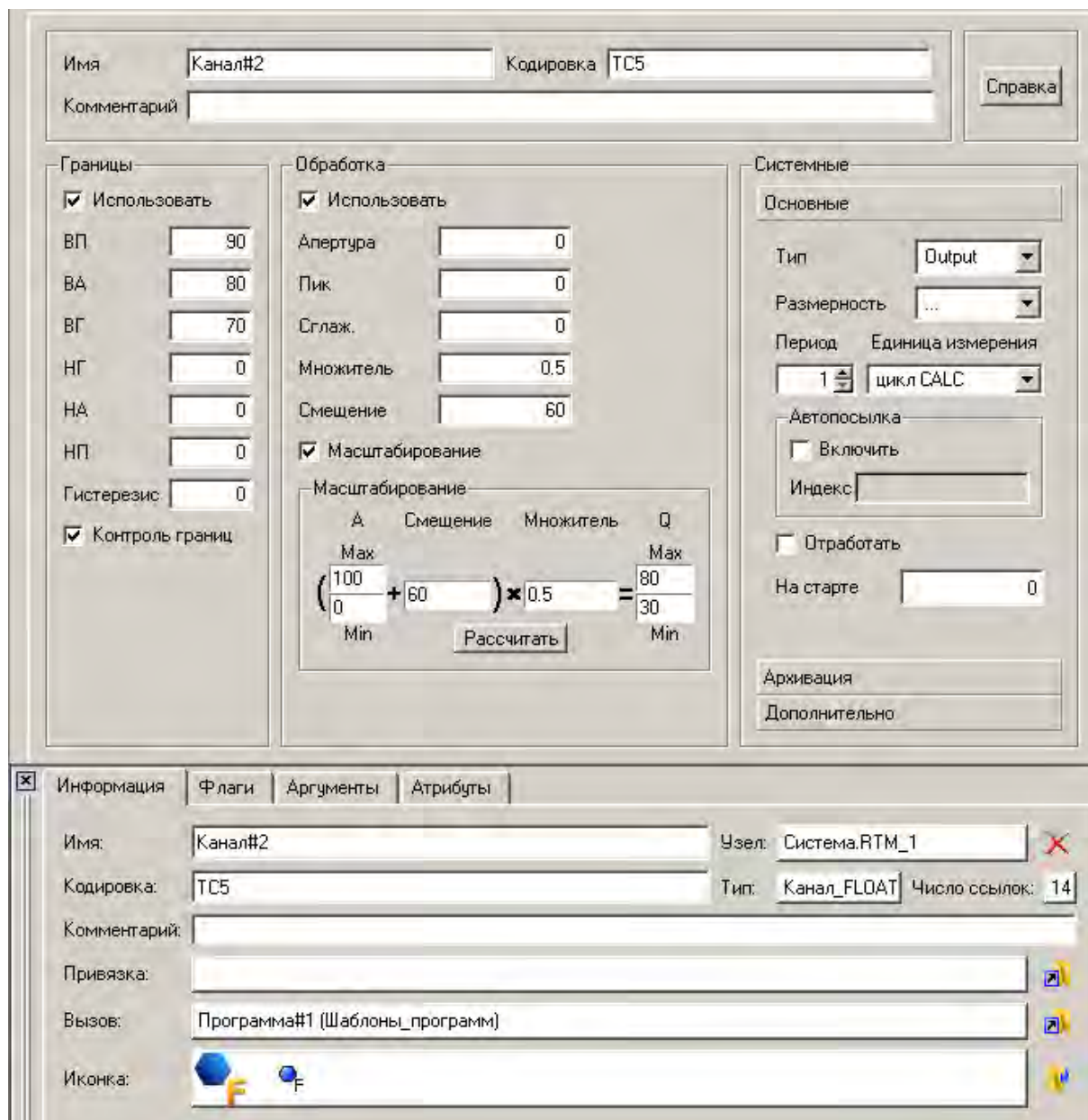


Рисунок 8.16. Вид окна редактора параметров канала класса FLOAT в Проекте-примере

83

Входное значение

87

Аппартное значение

74

Выходное значение

ВП (HL) - значение верхней границы
 ВА (HA) - значение верхней аверсионной границы
 ВТ (HW) - значение верхней предграницы полной границы
 НГ (DM) - значение нижней предграницы полной границы
 НА (LA) - значение нижней аверсионной границы
 НП (LL) - значение нижней границы

Экспоненциальное зглаживание

Линейное зглаживание

Клипирование
Ограничение значения сигнала по уровню

$val \times HL \text{ до } H = HL$
 $val \times LL \text{ до } B = LL$
 $val \times L \text{ до } H = L$

Трансляция
Трансляцией называется вызов программы числовым каналом (это единственное назначение свойства вызов числового канала)

$R \Rightarrow \text{программа} \equiv X$

Масштабирование
Изменение масштаба и сдвиг значения сигнала

$Q = (A - Z) / (M - X)$

Масштабирование

A	Сдвиг	Множитель	Q
Max	(100 + 60)	0.5	Max 80
Min	0		Min 30

Рассчитать

Программа

Блок зашты от отрицательного значения сигнала

Рисунок 8.17. Вид окна программы-примера

В составе графического интерфейса пользователя программы представлен графический объект «Ползунок», который выполняет роль ручного задатчика значения аналогового сигнала в атрибут «In» канала класса FLOAT. В канале настроена процедура «Клиппирование» и указаны допустимые границы изменения амплитуды аналогового сигнала. Горизонтальная линия имеет настроенную функцию перемещения по вертикали в зависимости от значения атрибута «In» канала класса FLOAT. Реальное значение в канале отображается в графическом объекте «Текст» и с помощью полосы с динамической заливкой. Динамическую заливку также имеет фон и текст графического объекта «Текст». Процедура «Трансляция», настроенная в канале, выполняет вызов программы, написанной на языке программирования Техно FBD. Диаграмма программы представлена в составе графического интерфейса пользователя программы-примера. Аппаратное значение канала на выходе процедуры «Клиппирование» отображается с использованием графического объекта «Текст» и полосы с динамической заливкой. Далее аппаратное значение канала преобразуется процедурой «Масштабирование» в значение атрибута «Выходное». Параметры настройки данной процедуры обработки данных также представлены в составе графического интерфейса пользователя тестовой программы.

8.5. Описание лабораторного стенда

В составе лабораторного комплекса «Проектирование пультов управления современных АСУТП» применяется специализированный кнопочный блок ввода команд управления и индикации сигналов состояния оборудования «ФК-01л» (ФК) (рисунок 8.18, 8.19).

Дисплеи многоэкранной видеостены

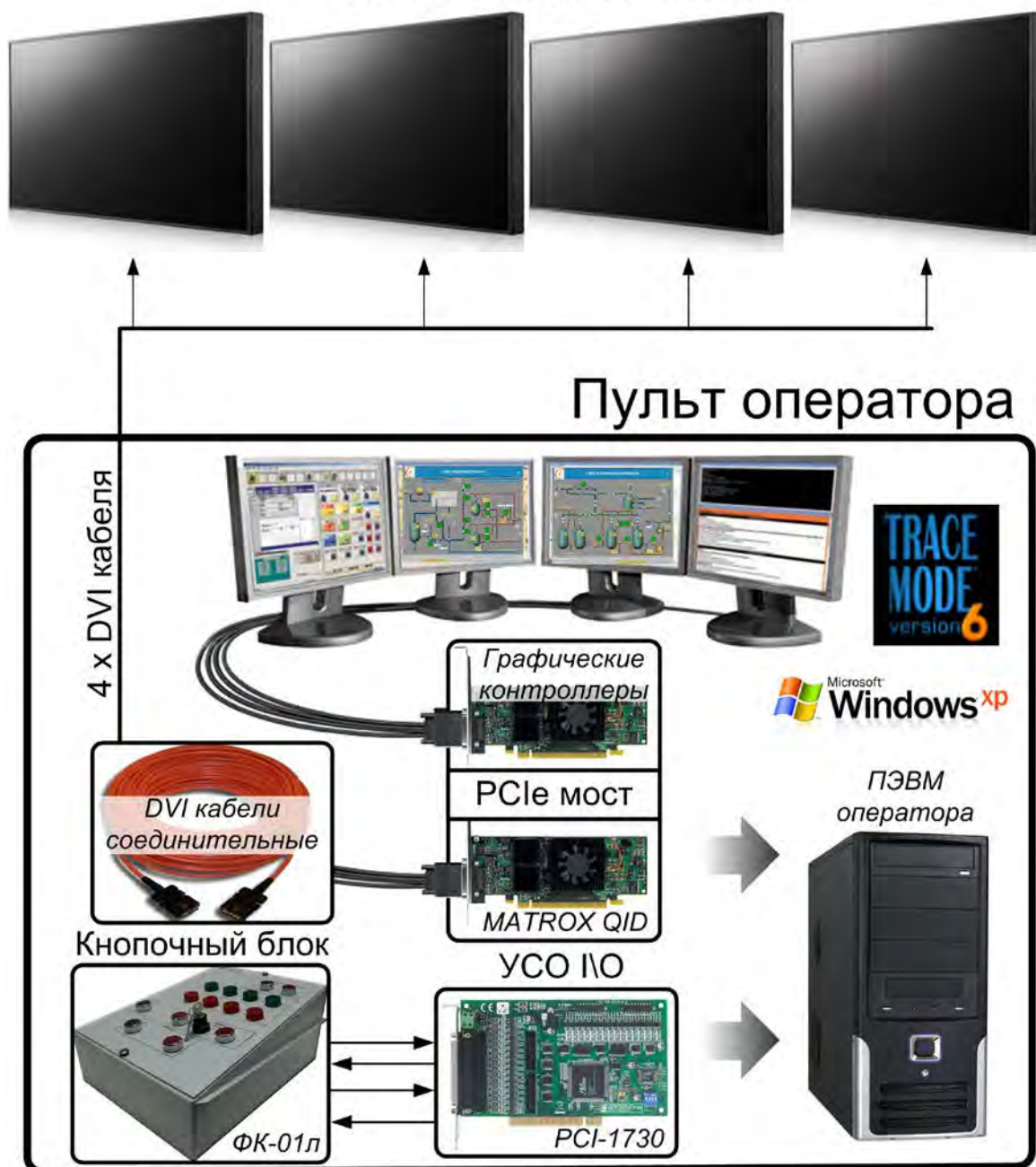


Рисунок 8.18. Структура лабораторного комплекса



Рисунок 8.19. Кнопочный блок «ФК-01л». Вид в перспективе

ФК подключен кабелем соединительным к УСО ADVANTECH PCI-1730 установленной в ПЭВМ лабораторного стенда на локальную шину PCI. Таблица соответствия входов и выходов платы УСО и подключенных к плате элементов коммутации электрических сигналов и индикации представлена ниже.

Таблица 8.2

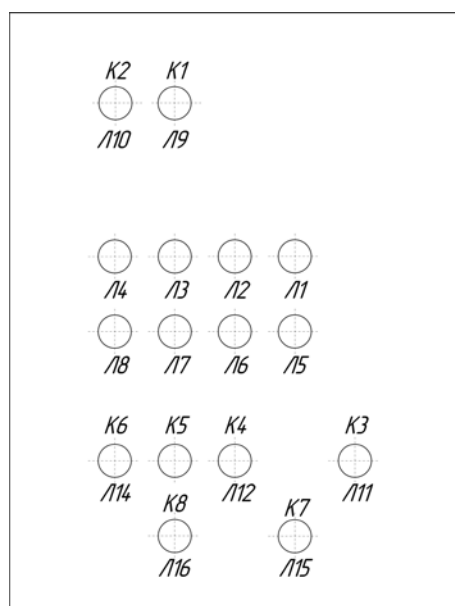
Таблица соответствия входов и выходов платы УСО подключенным к ним элементам коммутации электрических сигналов и их индикации

п.п	Обозначение порта драйвера платы УСО	Обозначение контакта на плате УСО PCI-1730	Обозначение элемента коммутации или индикации
1	Port#1 Тип: In Адрес на PCI шине: base +0 (base – произв-ый)	DI-0	K1
2		DI-1	K2
3		DI-2	K3
4		DI-3	K4
5		DI-4	K5
6		DI-5	K6
7		DI-6	K7
8		DI-7	K8

Таблица 8.2

Окончание

п.п	Обозначение порта драйвера платы УСО	Обозначение контакта на плате УСО PCI-1730	Обозначение элемента коммутации или индикации
9	Port#2 Тип: In Адрес на PCI шине: base +1 (base – произв-ый)	DI-8	Резерв
10		DI-9	Резерв
11		DI-10	Резерв
12		DI-11	Резерв
13		DI-12	Резерв
14		DI-13	Резерв
15		DI-14	Резерв
16		DI-15	Резерв
17	Port#3 Тип: Out Адрес на PCI шине: base +0 (base – произв-ый)	DO-0	Л1
18		DO-1	Л2
19		DO-2	Л3
20		DO-3	Л4
21		DO-4	Л5
22		DO-5	Л6
23		DO-6	Л7
24		DO-7	Л8
25	Port#4 Тип: Out Адрес на PCI шине: base +1 (base – произв-ый)	DO-8	Л9
26		DO-9	Л10
27		DO-10	Л11
28		DO-11	Л12
29		DO-12	Резерв
30		DO-13	Л14
31		DO-14	Л15
32		DO-15	Л16



Обозначения в таблице 8.2 представлены в соответствии с рисунком 8.20. Обозначения выполнены с обратной стороны изделия.

Для закрепления материала выполните практические задания из Приложения В.

Рисунок 8.20. Крышка «ФК-01л». Обозначения выполнены с обратной стороны

8.6. Вопросы для самоконтроля к главе №8

1. Что такое УСО, какие виды УСО Вы можете назвать?
2. Опишите назначение объекта Проекта «Terminal»?
3. Назовите назначение RWH-драйвера?
4. Нарисуйте структуру канала обработки дискретных данных в Проекте для плат ввода\вывода дискретных данных?
5. Назовите базовые параметры настройки драйвера платы УСО в TRACE MODE?
6. Что такое «Сетевой обмен», назначение процедуры и правила настройки в TRACE MODE?
7. Какой стандарт регламентирует обозначение элементов управления и индикации в АСУ, назовите его основные положения?
8. Опишите основные особенности элементов отображения и индикации значений аналоговых и дискретных сигналов?
9. Опишите основные возможности TRACE MODE по проектированию средств отображения информации и индикации событий?
10. Что такое «Клиппирование»? Каким образом данная процедура настраивается и применяется в TRACE MODE 6.06.2?
11. Может ли процедура «Трансляция» не использоваться при обработке данных в каналах Проекта?
12. Что такое «Трансляция» в рамках Проекта?
13. Назначение процедуры «Масштабирование» в рамках Проекта?
14. Что такое «Дамп»? Правила настройки данной функции?
15. Правила проектирования интерфейса пользователя в DOS?
16. Каким образом определяется количество параметров настройки отображения одного параметра в TRACE MODE 6.06.2 (в DOS)?
17. Каким образом рассчитывается координата знакоместа в текстовом интерфейсе пользователя для DOS в TRACE MODE 6.06.2?
18. Какое количество цветов может быть запрограммировано для одного отображаемого параметра в текстовом интерфейсе пользователя для DOS в TRACE MODE 6.06.2?
19. Назначение драйвера t13.dll в TRACE MODE?

9. МЕЖПРОГРАММНОЕ ВЗАИМОДЕЙСТВИЕ В ОС MS WINDOWS С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМА DDE, ВЗАИМОДЕЙСТВИЕ С РЕЛЯЦИОННОЙ БАЗОЙ ДАННЫХ ПОД УПРАВЛЕНИЕМ СУБД MS ACCESS

При разработке программного обеспечения распределённой АСУ, часто требуется передача данных между программами. Например, передача параметров процесса производства в экономический отдел в программу 1С или MS Excel для расчёта стоимости конечного продукта или скорости его получения. Для выполнения таких задач, на информационном уровне АСУ можно применять механизм динамического обмена данными между приложениями операционной системы (для MS Windows – DDE).

Для выполнения функций учёта состояния оборудования или архивирования параметров производства может также быть использована база данных (БД). Для управления БД используются специальные программы – системы управления БД (СУБД). Обращение к СУБД из Проекта TRACE MODE с использованием SQL-запросов автоматизировано разработчиками пакета. Данная функция также значительно облегчает процесс разработки прикладного ПО и значительно расширяет его функционал. Рассмотрим эти возможности TRACE MODE более детально.

9.1. Data dynamic exchange или механизм динамического обмена данными между приложениями ОС MS Windows

DDE – это программный механизм ОС MS Windows. В дальнейшем на основе устаревшего DDE была разработана технология OLE (ActivX). DDE применяется программистами для передачи данных из одного приложения MS Windows другому. Существует три режима обмена данными с использованием DDE: «Горячий канал», «Тёплый канал» и «Холодный канал». Значительное отличие в режимах обмена данными по DDE основано в области структуры канала передачи данных и скорости обмена данными.

Механизм DDE имеет клиент-серверную архитектуру. Это означает то, что одно из приложений должно выступать в качестве DDE-сервера (источника данных), а второе в качестве DDE-клиента (приёмника данных требующего их получить). Структура канала представлена на серии рисунков 9.1 – 9.3.

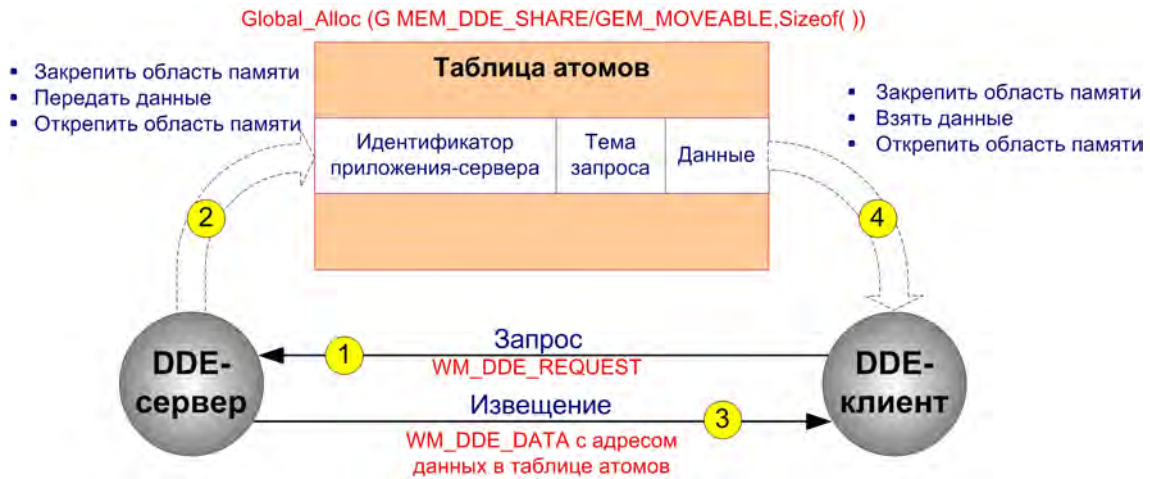


Рисунок 9.1. Архитектура канала DDE («Холодный канал»)

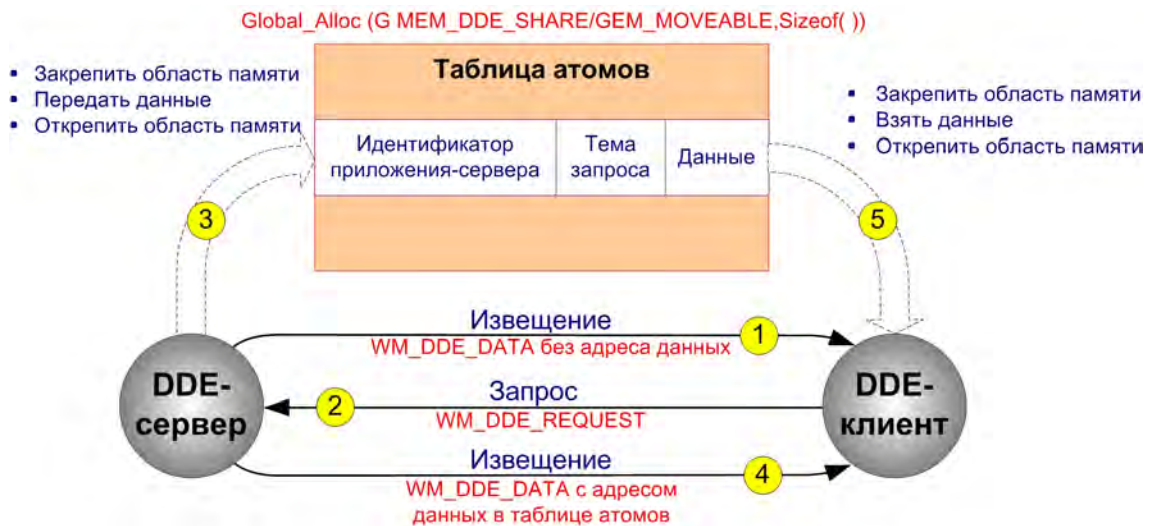


Рисунок 9.2. Архитектура канала DDE («Тёплый канал»)

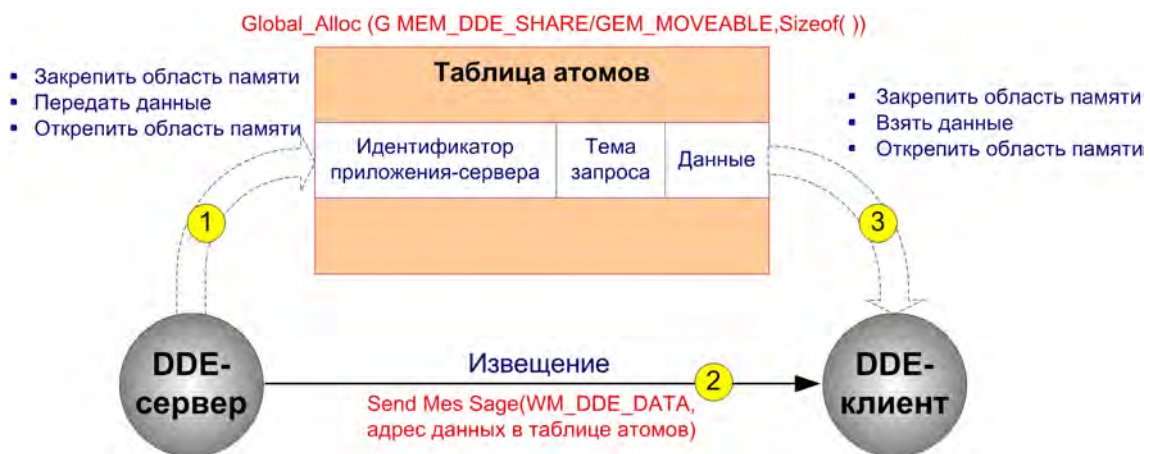


Рисунок 9.3. Архитектура канала DDE («Горячий канал»)

TRACE MODE по умолчанию является DDE-сервером. Обмен данными осуществляется через специальную область оперативной памяти, которую называют «Таблица атомов».

В первом случае приложение-клиент явным образом посылает запрос приложению-серверу, указывая нужный элемент данных. Приложение-сервер, получив запрос, предоставляет ему данные и сообщает адрес данных в области Таблицы атомов. Приложение-клиент забирает данные по указанному адресу.

Во втором случае, приложение-сервер отправляет приложению-клиенту извещение, о том, что данные изменились. Приложение-клиент, получив извещение, может отправить запрос на получение этих данных, после чего приложение-сервер отправляет новые данные в Таблицу атомов, а приложению-клиенту извещение с адресом данных в Таблице атомов.

В третьем случае приложение-сервер в каждый новый момент времени, независимо от требований приложения-клиента отправляет данные в Таблицу атомов. Затем отправляет приложению-клиенту извещение с адресом данных.

Следует отметить, что в большинстве случаев действующие программисты используют либо «Горячий канал» (быстрый обмен данными), либо «Холодный канал» (режим обмена данными по запросу), «Тёплый канал», как правило, используют реже.

Режим работы: TRACE MODE – DDE-сервер

Ранее упоминалось о том, что TRACE MODE RTM является приложением-сервером в клиент-серверной архитектуре DDE. Следовательно, любое приложение-клиент может быть зарегистрировано в операционной системе как приложение-клиент и подключиться к источнику данных в Проекте. Рассмотрим на примере архитектуру данного механизма.

Пусть в некотором Проекте создан генератор данных и канал, в который эти данные поступают. Также в составе Проекта присутствует некоторый компонент «Экран» с графическим объектом «Текст» выполняющим функцию отображения данных, поступающих из канала (рисунок 9.4).

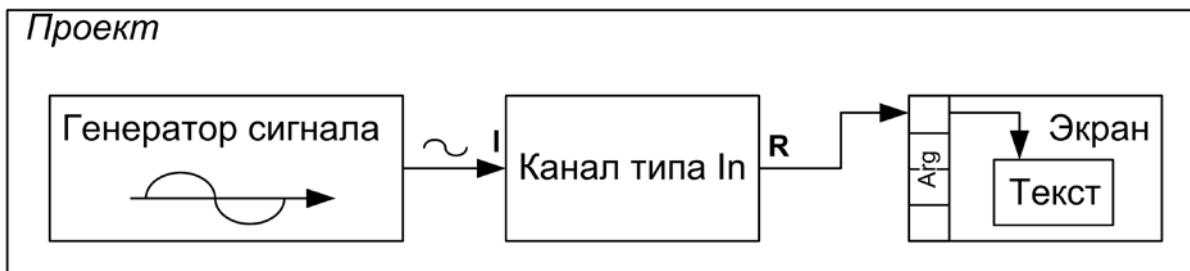


Рисунок 9.4. Структура программы

Далее в соответствии с архитектурой механизма DDE необходимо добавить программу-клиент. Пусть в качестве таковой выступает MS Excel, в некоторой ячейке таблицы которой необходимо получить реальное значение канала Проекта. Для выполнения данной задачи MS Excel выполняет запрос к MPB для получения указанного значения канала. В зависимости от выбранного режима обмена данными, MPB непрерывно или по запросу отправляет данные с выхода канала не только в компонент «Экран», но и в программу MS Excel (рисунок 9.5).



Рисунок 9.5. Структура системы динамического обмена данными

Представленная на рисунке 9.5 схема показывает только одно направление передачи данных. Обратное направление передачи данных тоже возможно. Рассмотрим принципы и набор команд, которые используются в TRACE MODE и MS Excel для инициализации DDE.

Команды управления DDE

Чтение данных

Для формирования команды в MS Excel необходимо:

- выбрать ячейку в таблице с конкретными координатами, например «С3»;
- сформировать команду запроса;
- выполнить команду.

Формат команды запроса данных следующий: «=

мер, «GET» для режима ADVISE или «PUT» для – REQUEST); С – название канала, данные которого необходимо передать в приложение-клиент; N – номер атрибута канала, значение которого необходимо передать. Список атрибутов каналов следующий: (3, C), (4, I), (7, P); (8, W), (26, HL), (27, LL), (28, HA), (29, LA), (30, HW), (31, LW), (79, CODE) и (127, NAME). По умолчанию используется значение (0, R). Пример команды запроса, в конечном итоге, может выглядеть таким образом: «=RTM0|GET!pila». Данная команда означает, что приложению-серверу данных с названием RTM и номером «0» необходимо передать в ячейку MS Excel, в которой записана данная команда реальное значение канал с названием «pila» в режиме DDE «Горячий канал».

Запись данных

Рассмотрим случай, когда необходимо обновить значения атрибутов каналов Проекта. В данном случае инициатором обмена и источником данных является приложение-клиент (MS Excel). Для инициализации процесса передачи данных из MS Excel в TRACE MODE в составе таблицы MS Excel необходимо создать специальный макрос. Разработчики предлагают использовать для выполнения этой задачи язык программирования Visual Basic. В руководстве пользователя к среде разработки TRACE MODE для каждого из режимов обмена приводится соответствующий программный код. Рассмотрим режим передачи данных REQUEST, текст программы, реализующий данный режим обмена, представлен ниже.

```
Sub write_pila_In()  
    chNum = Application.DDEInitiate("RTM0", "PUT")  
    Application.DDEPoke chNum, "pila", Worksheets("Лист1").Range("C3")  
    Application.DDETerminate chNum  
End Sub
```

В тексте данной программы переменные выделены жирным шрифтом. В их состав входят: RTM0 – название приложения-сервера данных, PUT – режим передачи данных, Лист1 – номер таблицы в составе книги MS Excel, C3 – ячейка таблицы MS Excel из которой необходимо взять и передать в приложение-сервер значение.

Следует отметить, что в составе одной книги MS Excel может быть создано несколько скриптов Visual Basic и переменные в составе скрипта могут меняться произвольно. Т.о., возможности использования данного механизма значительны.

9.2. Взаимодействие TRACE MODE с СУБД

Системы управления базами данных являются специализированными программами, которые содержат инструменты управления данными в базах данных на основании запросов сформированных на SQL. Данный подход к управлению данными в АСУ значительно упрощает процесс проектирования программного обеспечения. Язык программирования запросов (SQL) прост для понимания и позволяет реализовать необходимые операции. Благодаря этой особенности САПР TRACE MODE получает ещё один мощный механизм работы с данными в АСУ.

Существует множество различных СУБД для большинства операционных систем. В составе стандартного пакета MS Office для ОС MS Windows присутствует СУБД MS Access. Для работы с реляционной базой данных будем использовать её и стандартный ODBC-драйвер.

Структура Проекта, использующего связи с СУБД

Проект, создаваемый с учётом связей с СУБД, отличается от других Проектов только тем, что в его составе содержатся компоненты «Связь с СУБД» (рисунок 9.6). Вызов данных компонентов, как и других компонентов Проекта, осуществляется каналом класса CALL с типом вызова «SQLQuery». В составе редактора свойств данного компонента может быть создано множество SQL-запросов, управление которыми осуществляется при помощи значения посылаемого в канал CALL. Посылаемое значение соответствует номеру SQL-запроса.

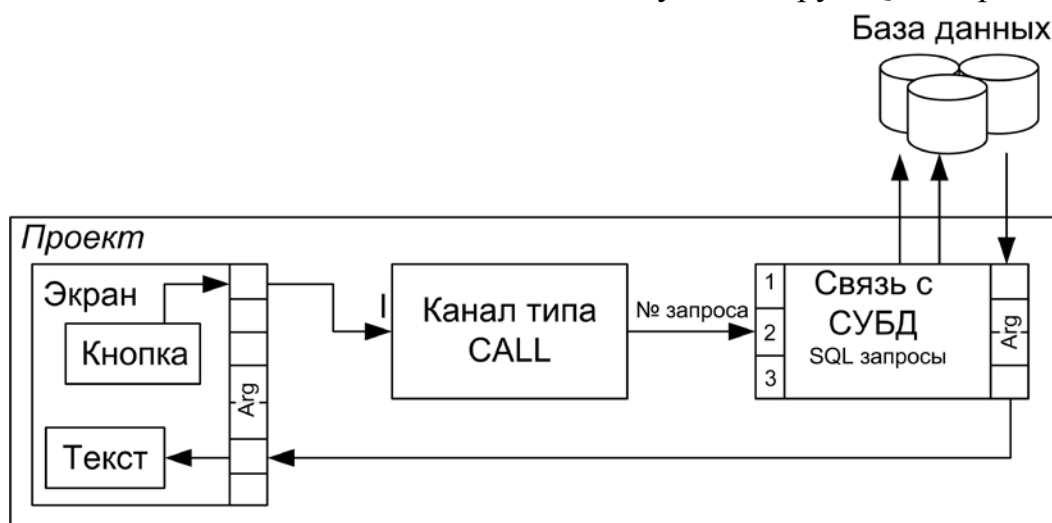


Рисунок 9.6. Структура программы

Алгоритм настройки компонента «Связь с СУБД»

Для настройки компонента «Связь с СУБД» необходимо:

- зарегистрировать базу данных как источник данных в ОС;
- создать структуру Проекта, представленную на рисунке 9.6;
- настроить связь с источником данных в компоненте «Связь с СУБД»;
- создать SQL-запрос, проверить правильность его выполнения.

Для более детального рассмотрения алгоритма настройки Проекта для связи с СУБД рассмотрим графический интерфейс пользователя соответствующего редактора в составе интегрированной среды разработки (рисунок 9.7).

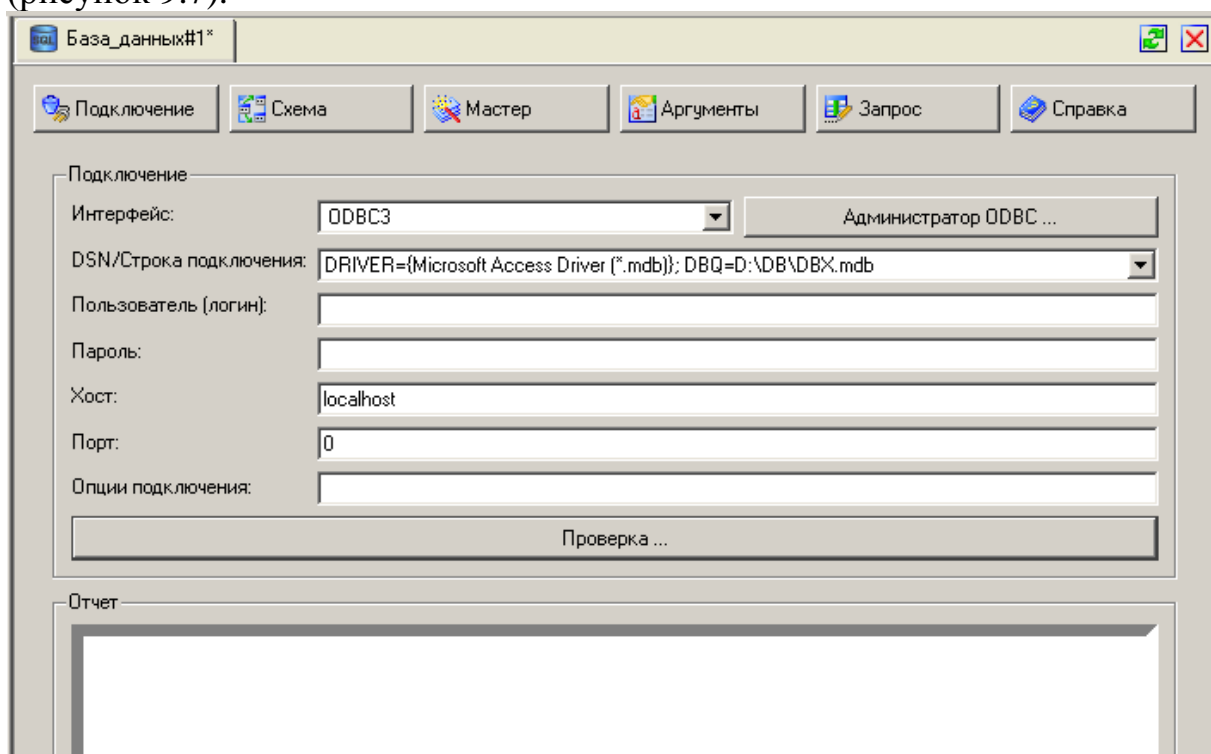


Рисунок 9.7. Вид окна программы редактора параметров настройки компонента Проекта «Связь с СУБД»

Редактор свойств компонента «Связь с СУБД» содержит следующие кнопки управления собственными функциями:

- Подключение – используется для открытия вкладки со свойствами подключения к источнику данных;

- Схема – используется для открытия вкладки навигации по структуре источника данных (в случае правильного подключения);
- Мастер – используется для открытия редактора SQL-запросов в интерактивном режиме;
- Аргументы – используется для открытия вкладки редактора аргументов компонента «Связь с СУБД»;
- Запрос – используется для открытия вкладки редактора SQL-запросов в ручном режиме;
- Справка – используется для перехода в раздел руководства пользователя ИС посвящённый правилам работы с компонентом «Связь с СУБД».

Следует отметить несколько важных особенностей данного редактора, которые необходимо учитывать или использовать при работе.

1. Для подключения к источнику данных (например, база данных) можно использовать два метода: зарегистрировать базу данных как источник данных в ОС; в явном виде записать параметры подключения к источнику данных в форме ввода «Строка подключения». Для регистрации базы данных в качестве источника данных необходимо использовать кнопку «Администратор ODBC» далее использовать стандартные процедуры ОС Windows. После регистрации источника данных необходимо выбрать данный источник в форме ввода «Строка подключения».
2. В нижней части редактора присутствует специальная форма «Отчёт» После выполнения каждого пункта настройки рекомендуется использовать кнопку «Проверить» для проверки правильности выполнения процедуры. Результат выполнения проверки отображается в форме «Отчёт».
3. Для того, что бы избежать ошибок при построении SQL-запросов и ускорить процесс разработки Проекта, на начальных этапах рекомендуется использовать редактор SQL-запросов в интерактивном режиме.

Рассмотрим примеры SQL-запросов созданных в компоненте «Связь с СУБД».

1. Загрузка данных из БД. Текст программы представлен ниже.

<pre> SELECT tabl.Y #ARG_000# FROM tabl WHERE </pre>	<pre> – <i>mun SQL-запроса Select</i> – <i>передача значения из поля «Y» в аргумент ARG_000</i> </pre>
--	--

```
tabl.X = 5
```

– искать по условию «значение в ячейке поля “X” на строке выборки = 5»

2. Добавить значение в конец поля в БД. Текст программы представлен ниже.

```
INSERT INTO TABL  
  (Y)  
VALUES  
  ('#ARG_001#')
```

– тип SQL-запроса Insert
– добавить значение в поле «Y»
– использовать для добавления значение аргумента ARG_000

3. Заменить значение в ячейке БД. Текст программы представлен ниже.

```
UPDATE tabl  
SET  
  Y=#ARG_002#  
WHERE  
  tabl.Y = 3
```

– тип SQL-запроса Update
– заменить значение в поле «Y», использовать значение аргумента ARG_002
– заменить значение = 3 в ячейке поля «Y»

4. Удалить строку данных в БД. Текст программы представлен ниже.

```
DELETE FROM tabl  
WHERE  
  tabl.Код = 9
```

– тип SQL-запроса Delete
– удалить строку, в которой значение ячейки в поле «Код» = 9

9.3. Вопросы для самоконтроля к главе №9

1. Опишите графический интерфейс пользователя редактора «Связь с СУБД»?
2. Каким образом выполняется выбор номера SQL-запроса в режиме работы МРВ?
3. Что такое DDE?
4. Какой вид DDE запроса в TRACE MODE соответствует режиму работы канала DDE «Горячий канал»?
5. Какой SQL-запрос используется для выборки данных из БД?
6. Какой SQL-запрос используется для удаления строки данных из БД?
7. Какой SQL-запрос используется для замены данных в БД?

8. Какой SQL-запрос используется для добавления данных в БД?
9. Чем отличается вид DDE запроса «ADVISE» от «REQUEST»?
10. Может ли быть использован SQL-запрос в TRACE MODE без регистрации БД в качестве источника данных в операционной системе?
11. Какие параметры используются при программировании макроса в составе таблицы Excel при формировании канала DDE для передачи данных из MS Excel в канал TRACE MODE?
12. Каким образом выполняется вызов компонента «Связь с СУБД» в TRACE MODE?
13. Формат команды-DDE-запроса в MS Excel в режиме «ADVISE»?
14. Нарисуйте структуру канала DDE типа «Горячий»?
15. Нарисуйте структуру канала DDE типа «Тёплый»?
16. Нарисуйте структуру канала DDE типа «Холодный»?
17. Запишите команду MS Excel, которая позволяет сделать запрос данных по каналу DDE. Каким образом определяется атрибут канала, из которого выполняется чтение значения?

СПИСОК ЛИТЕРАТУРЫ

1. Столингс У.. Операционные системы. – М: Вильямс, 2002. – 848 с.
2. Танненбаум Э.. Современные операционные системы. – С-Пб: Питер, 2002. – 1040 с.
3. Деменков Н.П. SCADA-системы как инструмент проектирования АСУ ТП. – М: Изд-во МГТУ им. Баумана, 2004. – 236 с.
4. Андреев Е.Б., Куцевич Н.А., Синенко О.В. SCADA-системы: взгляд изнутри. – М: Издательство РТСофт, 2004. – 176 с.
5. Петров И.В.. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М: Солон-Пресс, 2004. – 256 с.
6. Аристова Н.И., Корнеева А.И. Промышленные программно-аппаратные средства на отечественном рынке АСУ ТП. – М: Научтехлитиздат, 2001. – 324 с.
7. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. – С-Пб: Питер, 2002 – 736с.
8. Техническое и программное обеспечение лабораторного комплекса «Организация пультов управления современных АСУ ТП»: учебное пособие / А. А. Мезенцев, В. М. Павлов, К. И. Байструков. – Томск: Изд-во ТПУ, 2007. – 128 с.
9. Linux SCADA // linuxscada.info/list.htm.

ПРИЛОЖЕНИЕ А. Практические задания к главе №3

1. Разработайте на языке программирования «Техно ПЛ» функцию сложения 3-х чисел и вычитания полученной суммы из 4-го числа ($y=x_1-(x_2+x_3+x_4)$).
2. Разработайте на языке программирования «Техно ПЛ» функцию сложения квадратов 2-х чисел: $y=x_1^2+x_2^2$.
3. Разработайте на языке программирования «Техно FBD» функцию сложения 4-х чисел с использованием одного и нескольких блоков сложения.
4. Разработайте на языке программирования «Техно FBD» функцию сложения квадратов 2-х чисел: $y=x_1^2+x_2^2$; с использованием элементарных операций, а также – одного функционального блока.
5. Разработайте на языке программирования «Техно FBD» функцию косинуса сложного аргумента: $y=x^2+1$. В качестве сложного аргумента использовать один функциональный блок.
6. Разработайте на языке программирования «Техно ПЛ» функцию произведения 2-х сумм: $y=(x_1+x_2)*(x_3+x_4)$.
7. Разработайте на языке программирования «Техно FBD» функцию косинуса сложного аргумента: $y=x^2+1$. В качестве сложного аргумента использовать элементарные блок-функции.
8. Разработайте на языке программирования «Техно ST» функцию сравнения значений 2-х аргументов. В качестве индикатора результата использовать символы «1» и «0».
9. Разработайте на языке программирования «Техно SFC» компьютерную программу, которая реализует циклический счётчик (20 циклов) с выходом по некоторому условию на десятом цикле.
10. Разработайте на языке программирования «Техно FBD» функцию натурального логарифма от сложного аргумента. В качестве сложного аргумента функции использовать квадратный трёхчлен вида $y=K_1*x^2+K_2*x+K_3$.
11. Разработайте на языке программирования «Техно FBD» функцию защиты от аварии. Аварийным случаем считать выход значения некоторого унифицированного сигнала за верхнюю или нижнюю границы допустимого интервала значений. Зависимость значения экспериментальной функции от аргумента выбрать самостоятель-

но. Выходом программируемой функции должен быть «0» или «1» – «Допустимое значение» или «Авария» соответственно.

12. Разработайте на языке программирования «Техно FBD» функцию, которая будет циклически, один раз в минуту включать некоторое устройство. Условием включения считать единицу на выходе программы. Длительность импульса в 1 с считать достаточной для включения устройства.
13. Разработайте на языке программирования «Техно FBD» функцию квантования по уровню непрерывного сигнала (не менее 5 уровней квантования). Источник непрерывного сигнала выбрать самостоятельно.
14. Напишите комментарии к тексту компьютерной программы, разработанной на языке программирования Техно IL:

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

Выпишите операторы, используемые в программе, опишите их назначение с учётом всех применяемых к ним модификаторов.

15. Напишите комментарии к тексту компьютерной программы, разработанной на языке программирования «Техно IL»:

```
VAR VAR_000 : INT := 20; END_VAR
VAR VAR_001 : INT := 30; END_VAR
VAR VAR_002 : BOOL; END_VAR
LD 1
GT VAR_000 VAR_001
RET
LD VAR_002
LT VAR_000 VAR_001
CALC fff(VAR_000)
```

Выпишите операторы, используемые в программе, опишите их назначение с учётом всех применяемых к ним модификаторов.

16. Найдите ошибку(и) в тексте компьютерной программы, написанной на языке программирования «Техно IL», чему равен результат выполнения программы:
PROGRAM

```

VAR_INPUT ARG_000 : REAL := 0; END_VAR
VAR_OUTPUT ARG_001 : REAL := 0; END_VAR
VAR VAR_000 : REAL := 1; END_VAR
VAR VAR_001 : REAL := 0; END_VAR
VAR VAR_002 : REAL := 1; END_VAR
GT VAR_000 0 //Сравнение VAR_000 с числом
              (VAR_000 < 0, result:=false)
JMPC label //Условный переход пропускается
CAL func(VAR_001) //Безусловный вызов функции func
JMP label2 //Безусловный переход к метке label2
label:
CAL func(VAR_002) //Оператор не выполняется
LD VAR_002 //Оператор не выполняется
ST ARG_001 //Оператор не выполняется
label2:

END_PROGRAM

FUNCTION_BLOCK func
    VAR_INPUT ARG_000 : REAL; END_VAR
    VAR_OUTPUT ARG_001 : REAL; END_VAR

    ADD ARG_000 5
    LD ARG_000
    ST ARG_001

END_FUNCTION_BLOCK

```

17. Разработайте функцию сравнения значений двух аргументов на языке программирования «Техно ПЛ». В качестве результат выполнения функции на выходе компьютерной программы должен быть сформирован маркер: «0» или «1», при условии: «In1 > In2» или «In2 > In1» соответственно.
18. Разработайте функцию сортировки 3-х элементов на языке программирования «Техно ПЛ». Значения входных аргументов являются исходными данными, заданными в случайном порядке. Расположить значения функции в порядке убывания на выходе проектируемой компьютерной программы.
19. Разработайте функцию поиска максимального и минимального из 4-х элементов значения на языке программирования «Техно ПЛ». Компьютерная программа должна иметь два аргумента типа Out, один из них должен иметь максимальное значение, а второй минимальное.
20. Разработайте функцию сортировки элементов (5 шт.) одномерного массива на языке программирования «Техно ST».

21. Найдите ошибку(и) в тексте компьютерной программы, написанной на языке программирования «Техно ST», выполните комментарии к тексту программы:

```
PROGRAM
  VAR i : INT := 0; END_VAR
  VAR Xi : ARRAY OF INT[ 1 .. 5 ] := .5, .01, 0, .2, 3.14; END_VAR
  FOR i = 1 TO 5 DO
    IF Xi[i] > 3 THEN
      Xi[i] = cos(Xi[i]);
    ELSE
      Xi[i] = sin(Xi[i]);
    END_IF
  END_FOR
END_PROGRAM
```

22. Разработайте на языке программирования «Техно ST» функциональный блок защиты от аварии. Аварийным случаем считать выход значения некоторого унифицированного сигнала за верхнюю или нижнюю границы допустимого интервала значений. Зависимость значения унифицированного сигнала от аргумента выбрать самостоятельно и представить в виде дополнительного функционального блока. Выходом программируемой функции должен быть маркер: «0» или «1», соответствующие условию: «Допустимое значение» или «Авария» соответственно.
23. Используя методы построения шагов и переходов разработать на языке программирования «Техно SFC» следующую диаграмму:

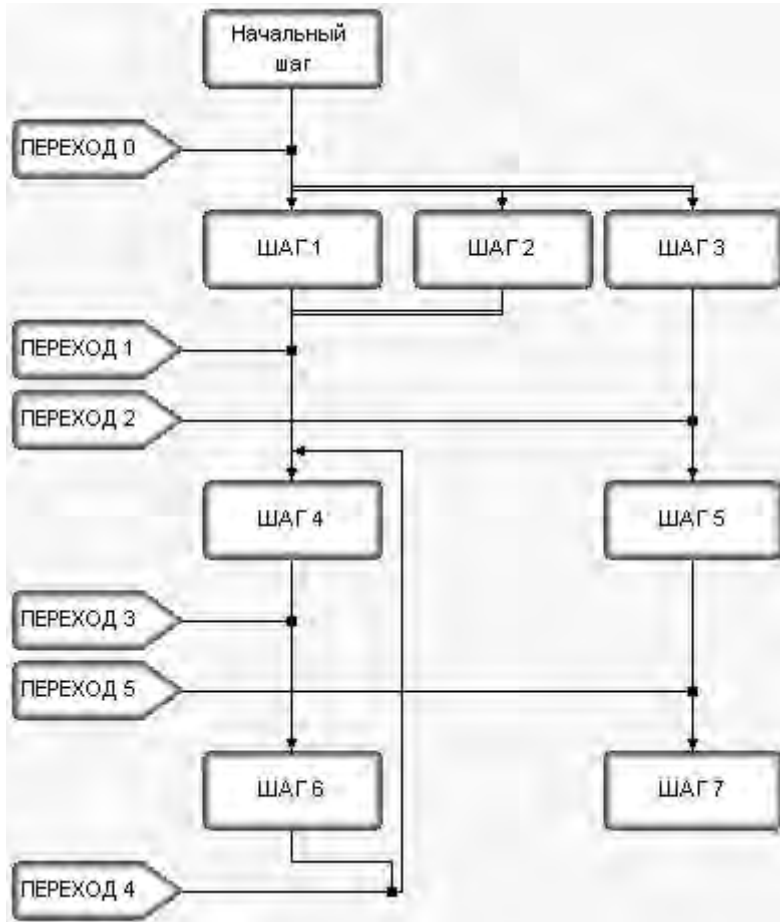


Рисунок 10. Диаграмма SFC-программы

24. Разработайте на языке программирования «Техно ST» логическую функцию инвертора выходного значения с сохранением конечного результата. Входной сигнал импульсный. Считать, что двухпозиционная кнопка без фиксации, формирующая входной сигнал имеет нормально-замкнутый контакт.



Рисунок 11. Диаграмма состояния системы

ПРИЛОЖЕНИЕ Б. Практические задания к главе №4

Задание №1

Объект управления – ёмкость с механической мешалкой, в которой смешиваются две жидкости А и В. В качестве показателя эффективности процесса перемешивания примем C_C – концентрацию какого-либо из компонентов в смеси, а целью управления – получение смеси с определённой концентрацией этого компонента.

$$C_C = \frac{G_A C_A + G_B C_B}{G_A + G_B}$$

Концентрации C_A и C_B считаются постоянными. Регулирование выполняется путём изменения расходов жидкостей. Для нормального хода технологического процесса необходимо поддержание заданного уровня в смесителе.

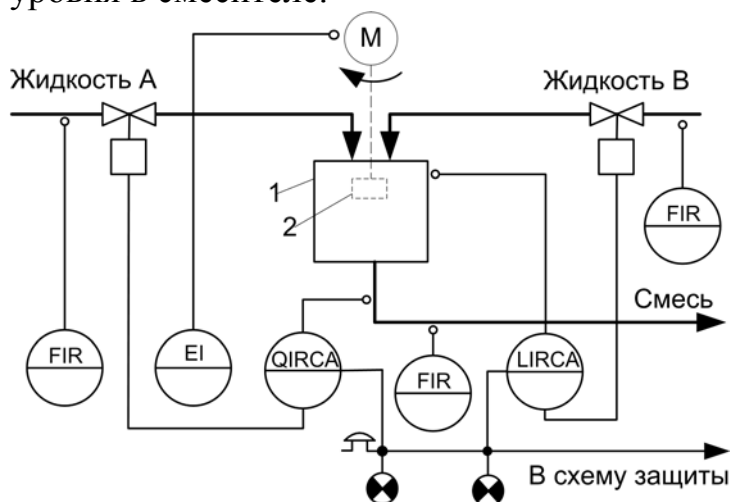


Рисунок 12. ФСА процесса смешивания жидкостей:

1 – ёмкость;

2 – механическая мешалка

Задание №2

Процесс отстаивания проводится с целью полного извлечения твёрдой фазы из жидкости, поэтому показателем эффективности процесса можно считать концентрацию твёрдой фазы в осветлённой жидкости, а целью управления её поддержание на заданном уровне. На процесс управления влияют: изменение расхода суспензии, осветлённой жидкости, а также мутность осветлённой жидкости. Также контролируется уровень границы раздела зон. Работа механической части отстойников контролируется путём непосредственного измерения момента на валу двигателя.

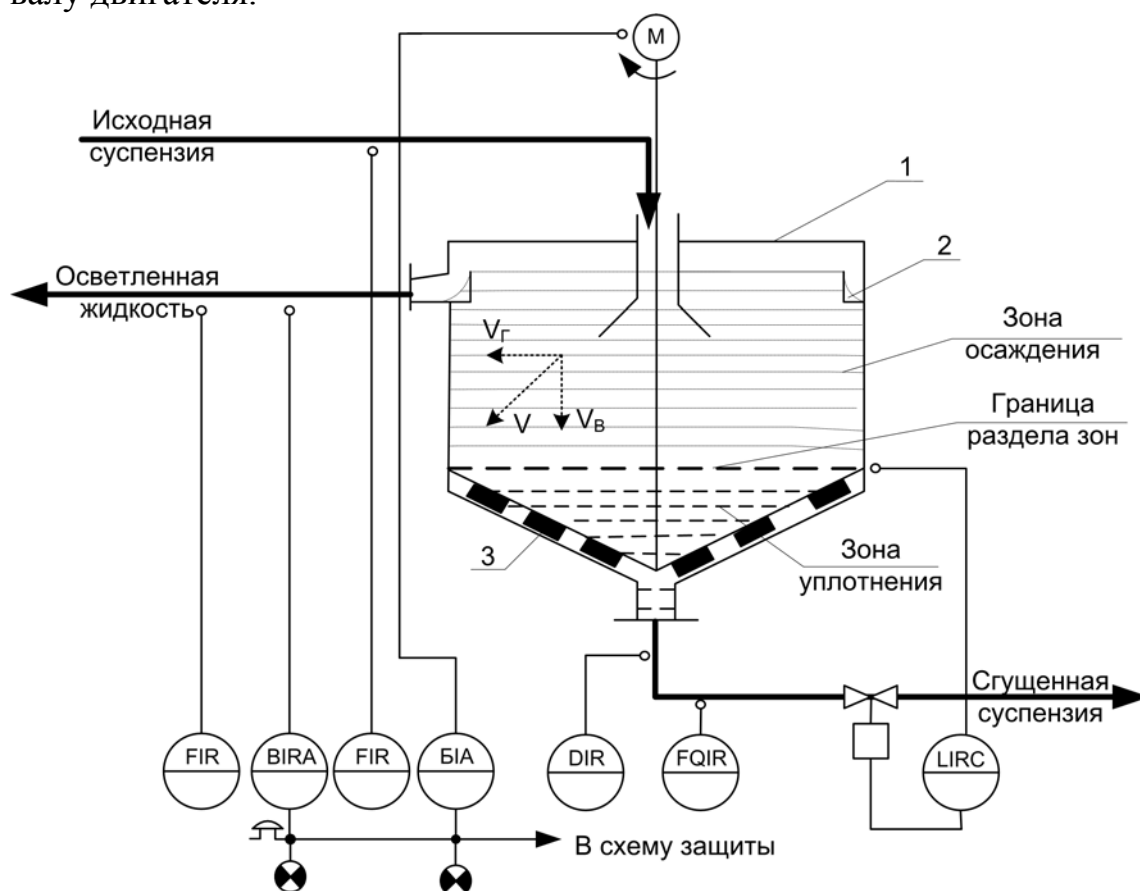


Рисунок 13. ФСА процесса отстаивания:

- 1 – отстойник;
- 2 – переливное устройство;
- 3 – мешалка;
- Б – момент на валу электродвигателя;
- В – мутность воды

Задание №3

Полученный в процессе центрифугирования осадок в дальнейшем должен поступать на сушку, поэтому его влажность должна быть минимальной (в диапазоне 10-30%). Этот показатель и будет целью управления. Наибольшим возмущающим воздействием будет изменение подачи суспензии. В системе управления должны контролироваться расходы суспензии, фугата и количество потребляемого электродвигателем масла. При перегрузке электродвигателя срабатывает сигнализация. Контролю и сигнализации подлежат также давление масла в системе и температура подшипников. При резком изменении давления масла и температуры должны срабатывать устройства защиты, отключающие центрифугу. Отключение также должно производиться при сильной вибрации центрифуги.

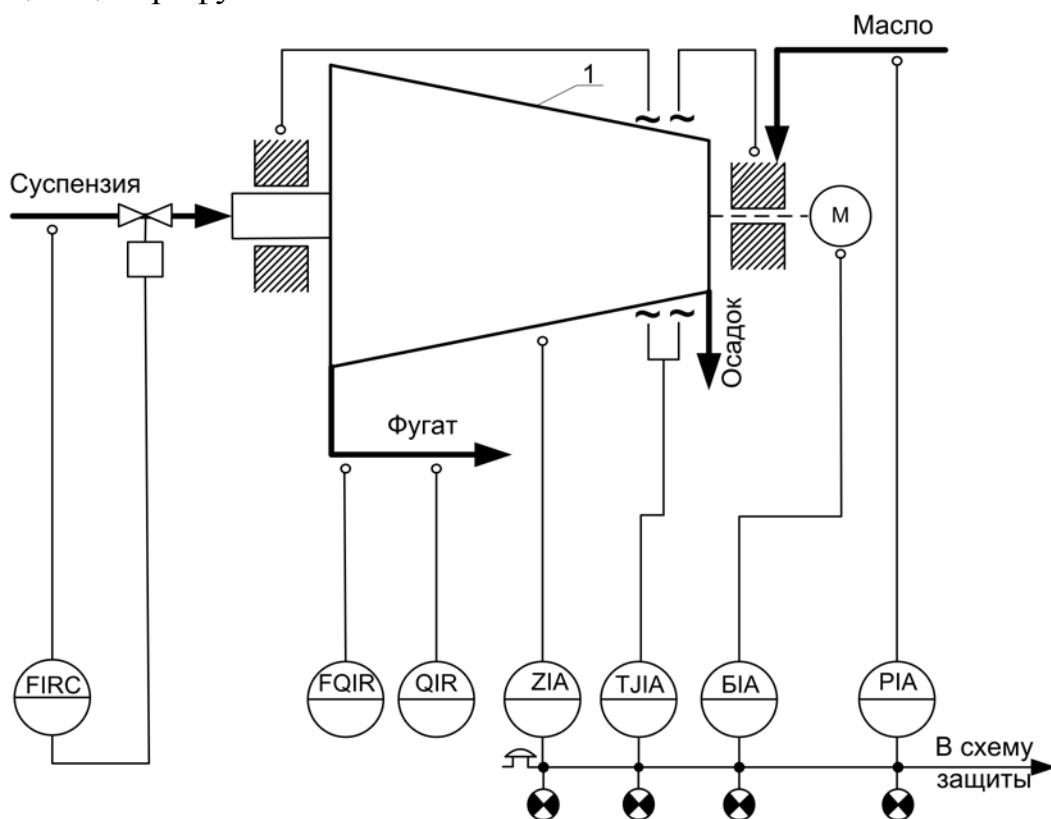


Рисунок 14. ФСА процесса центрифугирования:

- 1 – барабан центрифуги;
- Б – момент на валу электродвигателя;
- Z – уровень вибрации

Задание №4

Фильтрация производится барабанным вакуумным фильтром. Наиболее сильным возмущающим воздействием является изменение подачи суспензии. Параметр, требующий регулирования – уровень суспензии в ванне. Серьезное нарушение в работе фильтра – порыв фильтровальной ткани. Для предотвращения такой ситуации устанавливают датчики мутности фильтрата, а также устройство сигнализации и защиты. Кроме того, устанавливается датчик сигнализации и устройство защиты от перегрузки электродвигателя. Контролю подлежат расход суспензии и фильтрата, уровень жидкости в ванне, разрежение вакуум-линии, перепад давления до и после фильтровальной ткани, мутность фильтрата, мощность электродвигателя.

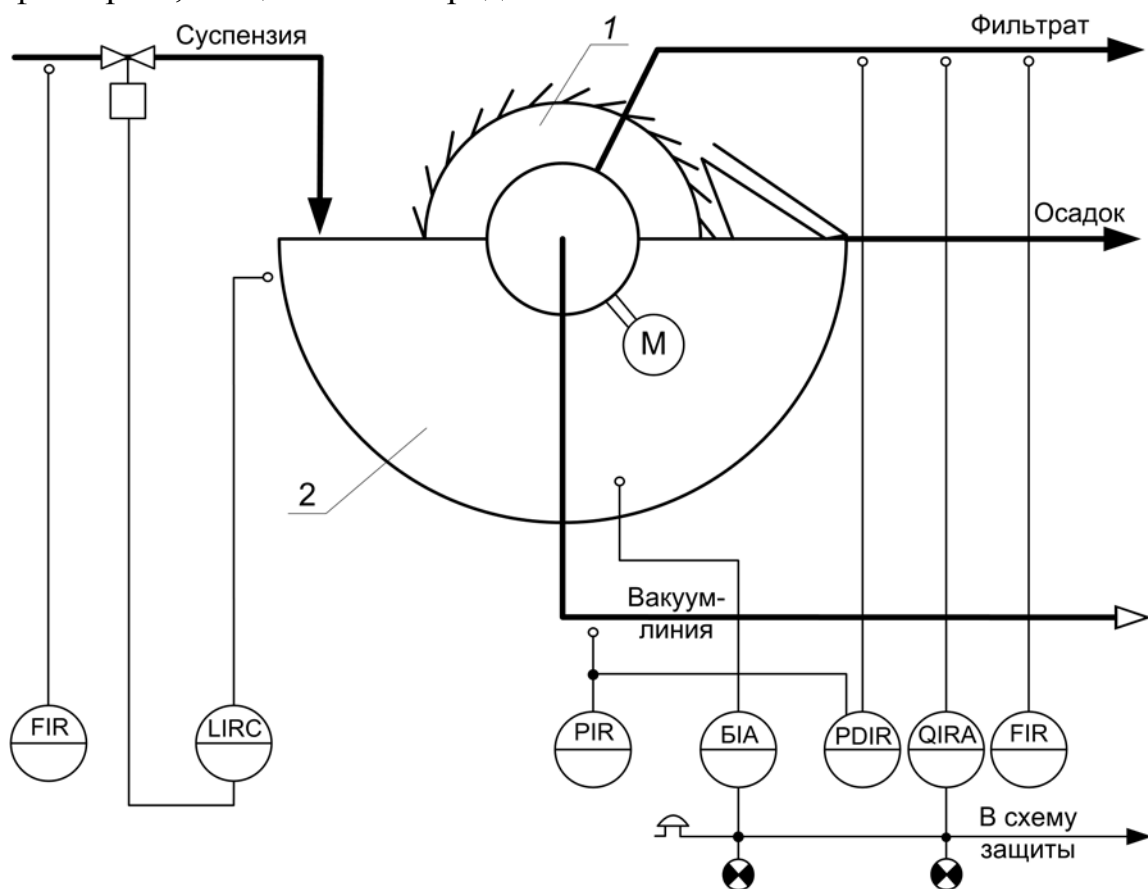


Рисунок 15. ФСА процесса фильтрации жидкостей:

1 – барабан;

2 – ванна;

Б – момент на валу электродвигателя

Задание №5

Рукавные фильтры устанавливают для полной очистки газа от твёрдых веществ, являющихся ценным продуктом, поэтому показателем эффективности выполнения процесса необходимо считать концентрацию твёрдых веществ в газе на выходе из фильтра, а целью управления – поддержание его на минимальном уровне. Наиболее полно отражает ход процесса перепад давления ΔP в камерах загрязнённого и очищенного газов. При достижении максимального перепада давления ΔP прецизионный регулятор выдаёт сигнал на электронные клапаны, установленные на магистрали сжатого газа. Клапаны открываются, импульсы сжатого воздуха через сопла поступают в рукава и деформируют ткань, сбивая при этом пыль. Регенерация ткани происходит до достижения минимального перепада давления. Качественная регенерация фильтрующей ткани рукавов будет происходить только при определённом давлении сжатого воздуха, подаваемого на продувку. Для стабилизации этого давления устанавливают регулятор.

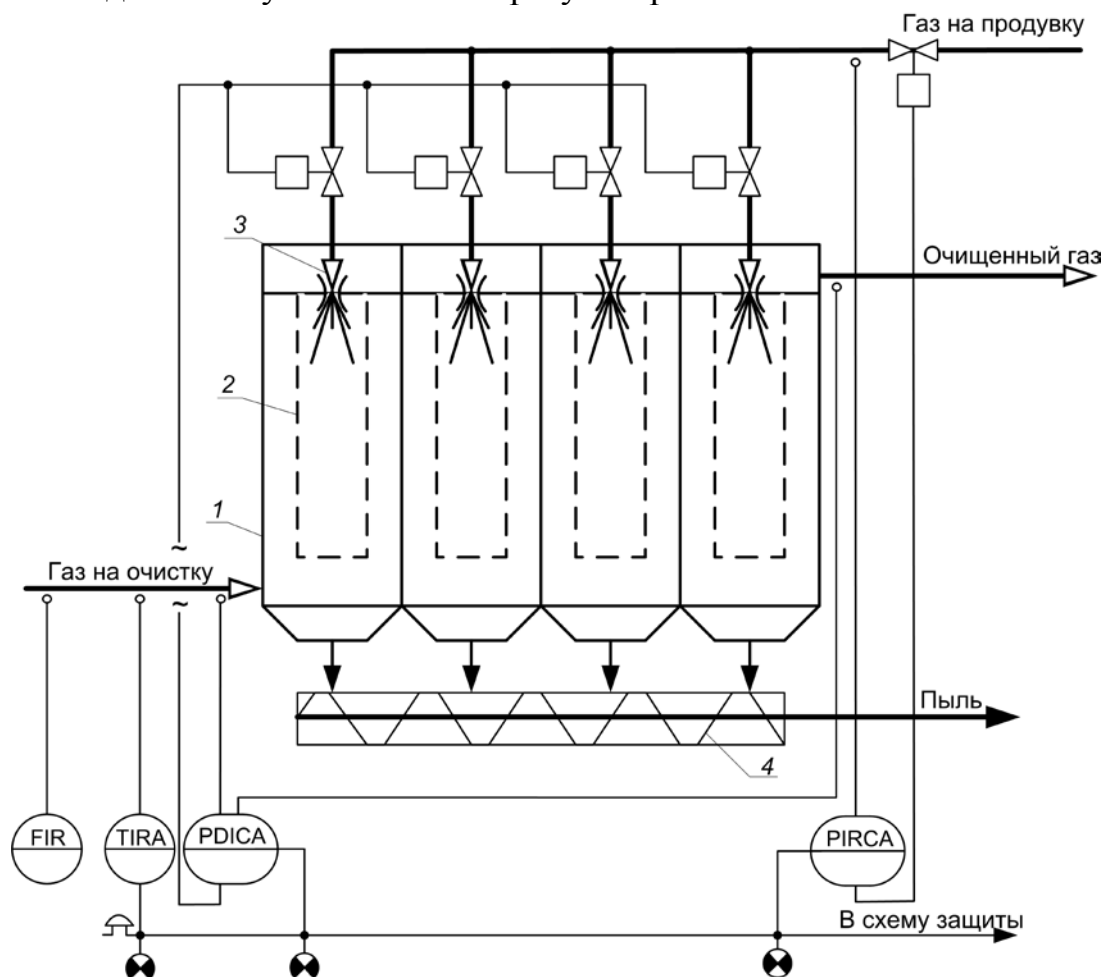


Рисунок 16. ФСА процесса фильтрации газов: 1 – корпус фильтра; 2 – рукав; 3 – сопла импульсной продувки; 4 – шнек

Контролю и сигнализации подлежат следующие параметры: температура загрязнённого газа (фильтровальная ткань рассчитана на определённое значение температуры), давление сжатого воздуха, перепад давлений (при превышении давления сжатого воздуха и перепада давлений заданных значений возможен разрыв ткани рукавов). Контролю также подлежит расход газового потока.

Задание №6

Очистка выполняется в фансуточной трубе Вентури, в которую через распылитель подаётся под небольшим давлением жидкость. Распылитель расположен параллельно газовому потоку движущемуся с большой скоростью. Для очистки важно поддерживать количество и размер капель жидкости, что достигается стабилизацией давления жидкости. При взаимодействии капель жидкости с потоком газа происходит их дополнительное распыление, что увеличивает степень очистки. Дисперсность вторичного распыления зависит от скорости газового потока W . Для стабилизации W достаточно поддерживать постоянным перепад давления на трубе Вентури ΔP . Регулирующее воздействие вносится изменением поперечного сечения горловины трубы Вентури. Постоянный перепад давления ΔP является гарантией постоянного расхода газа. Влажные пылеочистители подвержены к забиванию. Поэтому при достижении критического давления ΔP необходима сигнализация и аварийное отключение установки.

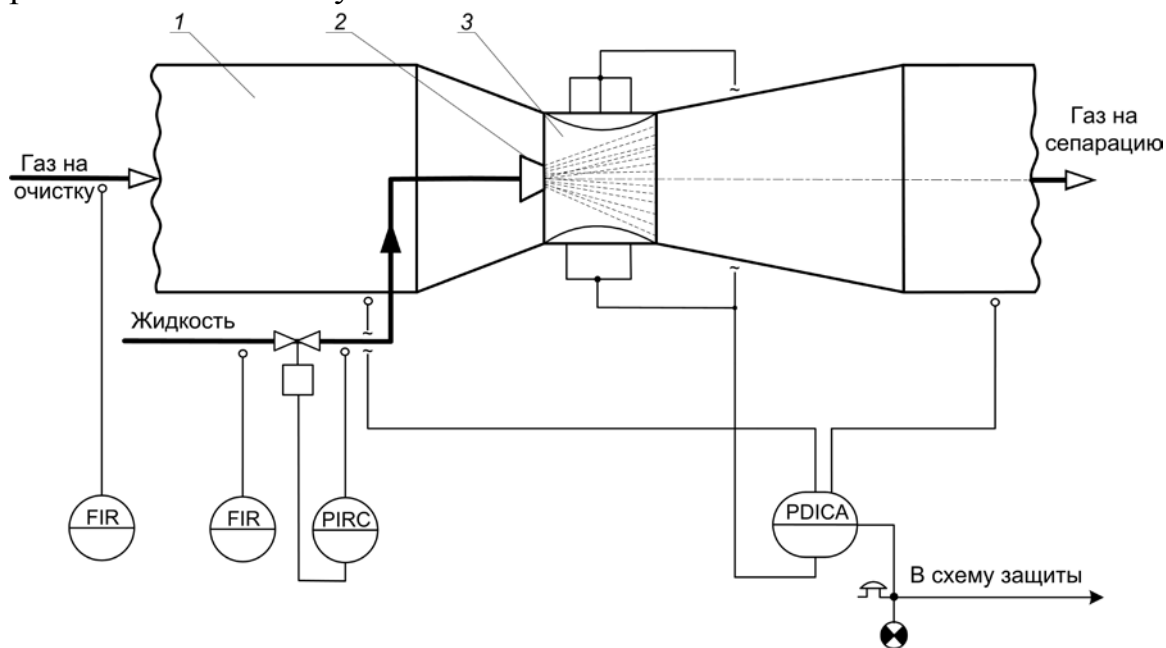


Рисунок 17. ФСА процесса флажной очистки газов: 1 – корпус трубы Вентури; 2 – форсунки; 3 – регулируемая горловина трубы Вентури

Задание №7

Параметрами, от которых зависит концентрация пыли на выходе из электрофильтра, является напряжение питания U , начальная концентрация пыли и расход газа на очистку G . Расход газа нужно стабилизировать. Напряжение U должно поддерживаться на максимально высоком уровне, близком к критическому. Для этого устанавливают автоматическое устройство, которое осуществляет плавное повышение значения U до возникновения пробоя в электродном промежутке. В момент возникновения пробоя срабатывают реле максимального тока и минимального напряжения, они дают команду на быстрое снижение напряжения U до значения, обеспечивающего гашение дуг. Через некоторое время устройство вновь начинает увеличивать напряжение до пробоя. Цикл повторяется. Контролю в данном процессе подлежит расход, температура и влажность газового потока, напряжение и сила тока, температура масла трансформаторов.

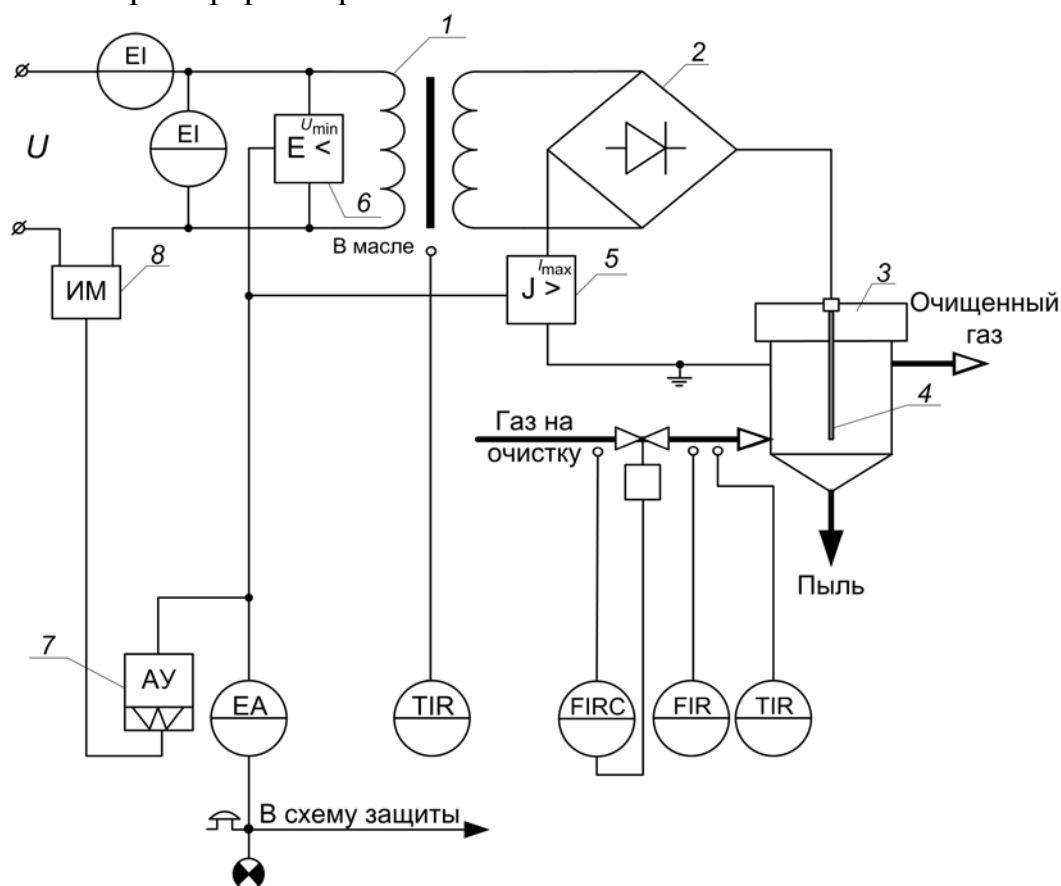


Рисунок 18. ФСА процесса очистки газов электрической дугой: 1 – трансформатор; 2 – высоковольтный выпрямитель; 3 – электрофильтр; 4 – коронирующий электрод; 5 – реле максимального тока; 6 – реле минимального напряжения; 7 – устройство автоматического управления; 8 – исполнительный механизм

Задание №8

Процесс нагревания представлен на примере поверхностного кожухотрубчатого теплообменника, в который попадают нагреваемый продукт и теплоноситель. Показателем эффективности процесса нагрева является температура продукта t'' на выходе аппарата.

$$t'' = \frac{G_t c_t}{G c} (t_t' - t_t'') + t'$$

где G и G_t – расходы, c и c_t – удельные теплоёмкости, t' и t_t' – начальные температуры, t'' и t_t'' – конечные температуры продукта и теплоносителя.

В качестве регулирующего воздействия на конечную температуру продукта следует выбрать расход теплоносителя. В качестве контролируемых параметров следует выбрать расходы, начальные и конечные температуры, давление продукта и теплоносителя. В связи с тем, что резкое снижение расхода продукта может привести к выходу из строя теплообменника, необходимо перекрывать линию горячего теплоносителя.

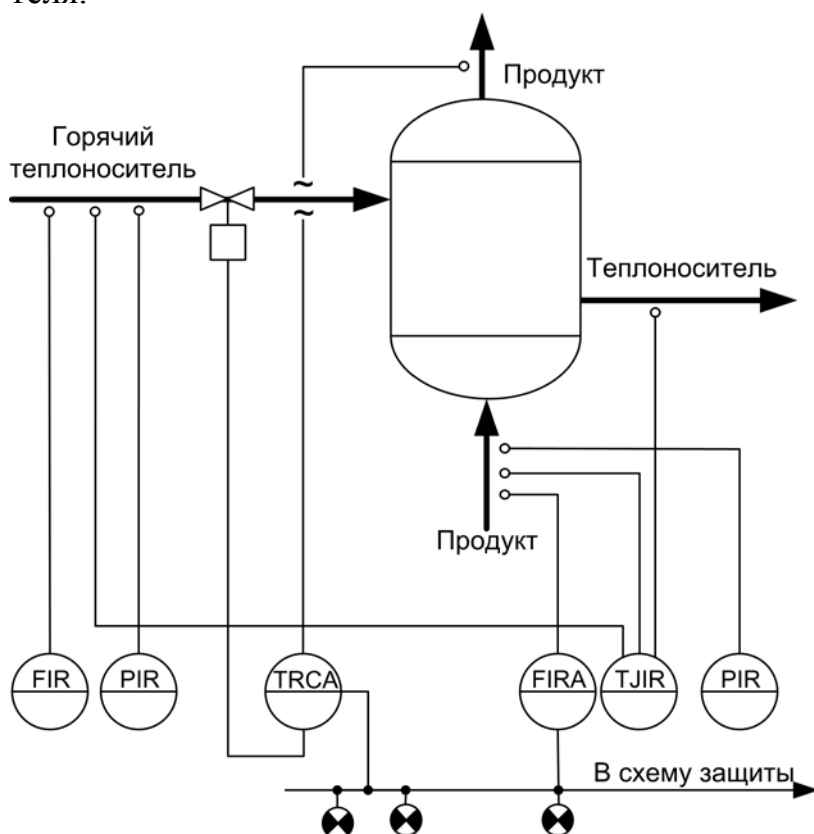


Рисунок 19. ФСА процесса нагрева продукта

Задание №9

В трубчатой печи продукт прокачивается через змеевик, нагревается за счёт тепла, выделяющегося при сжигании топлива. Регулирование температуры продукта выполняется за счёт изменения расхода топлива, подаваемого в печь. В связи с тем, что печь является сильно инерционным объектом совместно с контуром регулирования расхода топлива по температуре продукта, используется дополнительный контур коррекции расхода по температуре топочных газов.

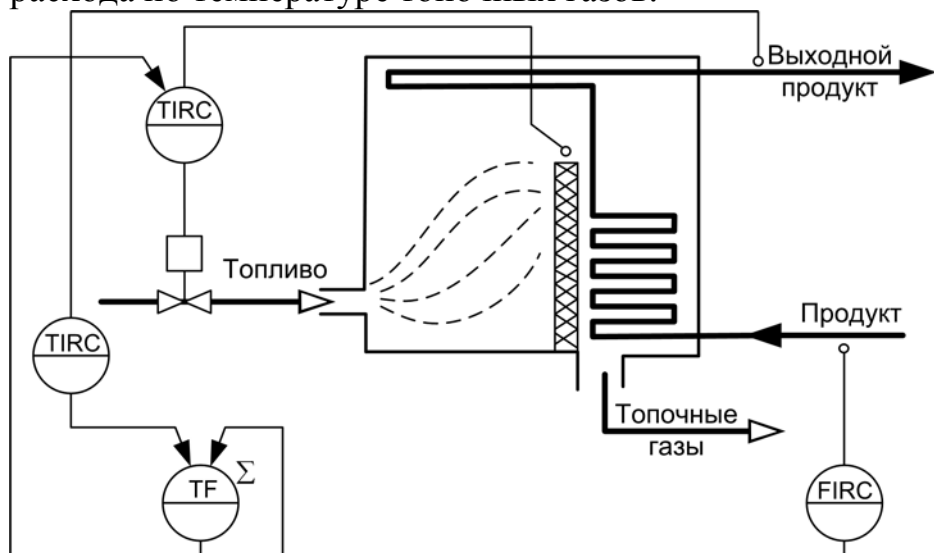


Рисунок 20. ФСА процесса нагрева продукта в трубчатой печи

Задание №10

Разряжение при вакуумной выпарке создаётся при помощи барометрических конденсаторов и вакуум-насосов, служащий для удаления смеси несконденсированных газов с воздухом. Регулирование разряжения может осуществляться изменением расхода воздуха, подаваемого вакуум-насосом из атмосферы. Расход воды при этом изменяется в зависимости от температуры стоков из барометрического конденсатора.

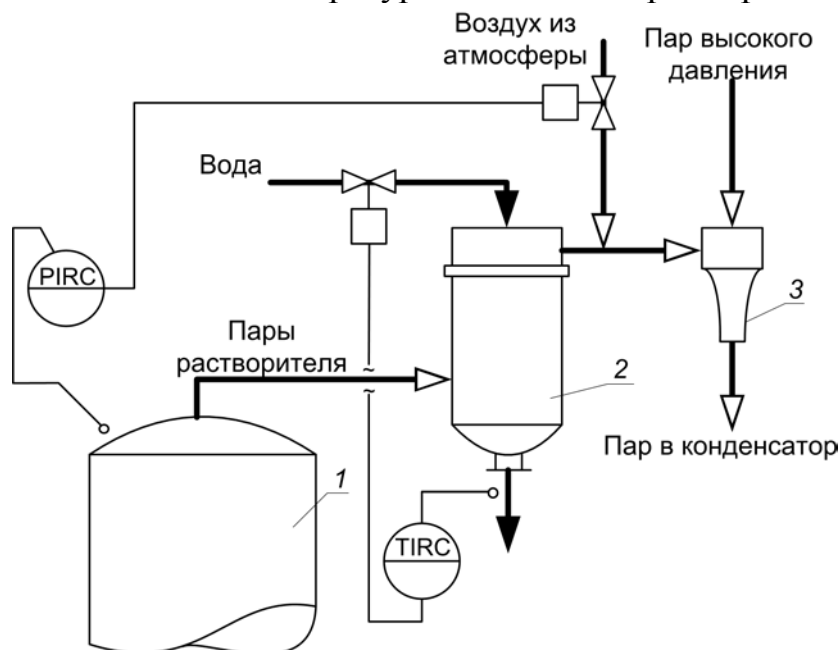


Рисунок 21. ФСА процесса вакуумной выпарки:

- 1 – выпарной аппарат;
- 2 – барометрический конденсатор;
- 3 – вакуум-насос

Задание №11

Кристаллизация разбавленного продукта производится в технологической схеме за счёт испарения части растворителя в аппаратах выпарного типа. Часть потока разбавленной кислоты отводится в кипятильник, где испаряется, а затем кристаллизуется в верхней камере аппарата. Особенность технологической схемы в том, что необходимо регулировать перепады уровней температуры в верхней и нижней камерах кристаллизатора.

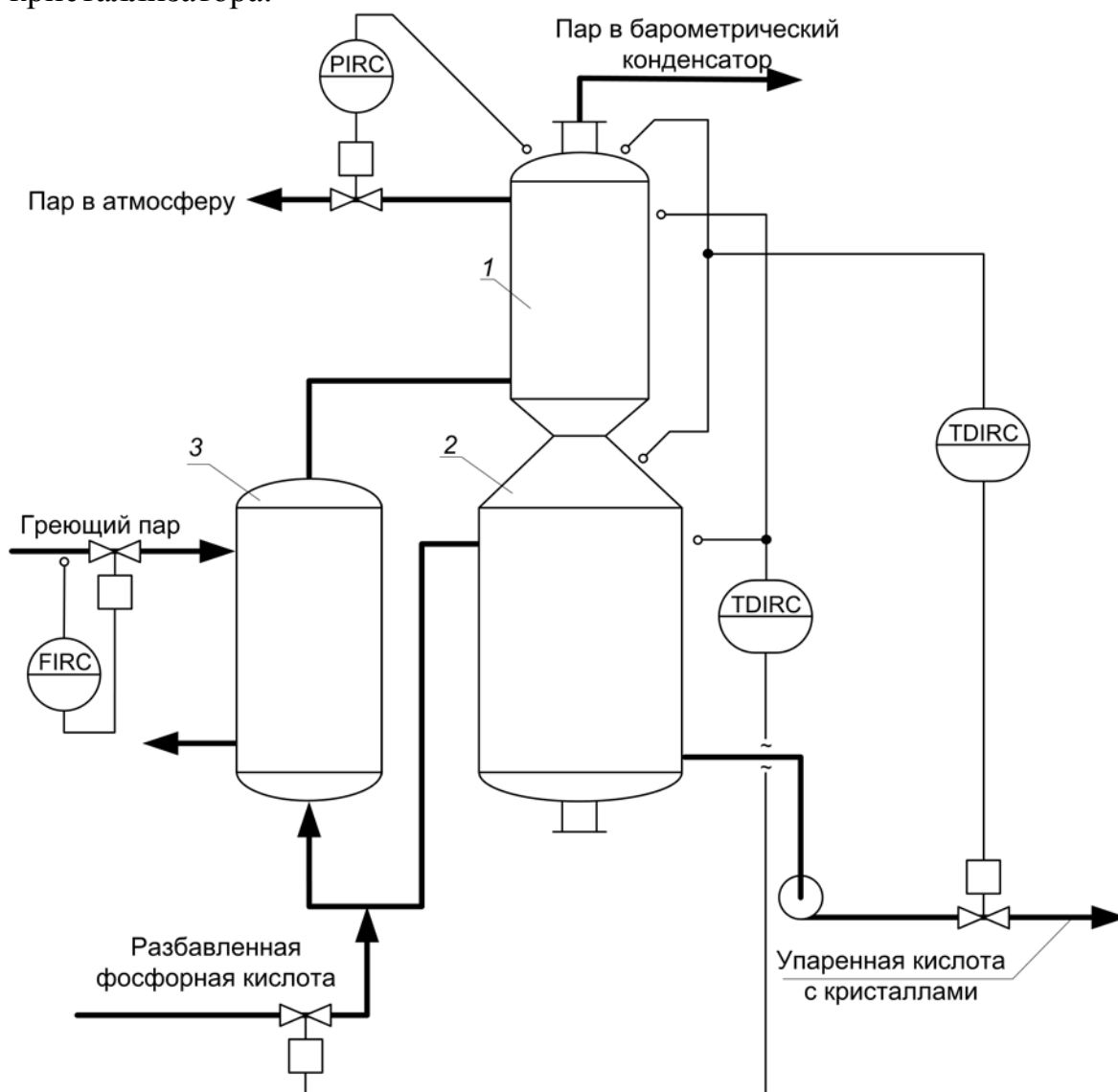


Рисунок 22. ФСА одноступенчатого кристаллизатора выпарного типа:

- 1 – верхняя камера;
- 2 – нижняя камера;
- 3 – кипятильник

Задание №12

На схеме представлена ректификационная установка, в которой исходная смесь продуктов, нагреваемая теплоносителем, разделяется на два компонента – дистиллят и остаток. В дефлегматоре конденсируется высококипящий компонент. При сильном изменении состава исходной смеси нормальное протекание процесса разделения возможно только при коррекции качества смеси в колонне путём регулирования расходов дистиллята и теплоносителя. Для решения данной задачи используется вычислительное устройство.

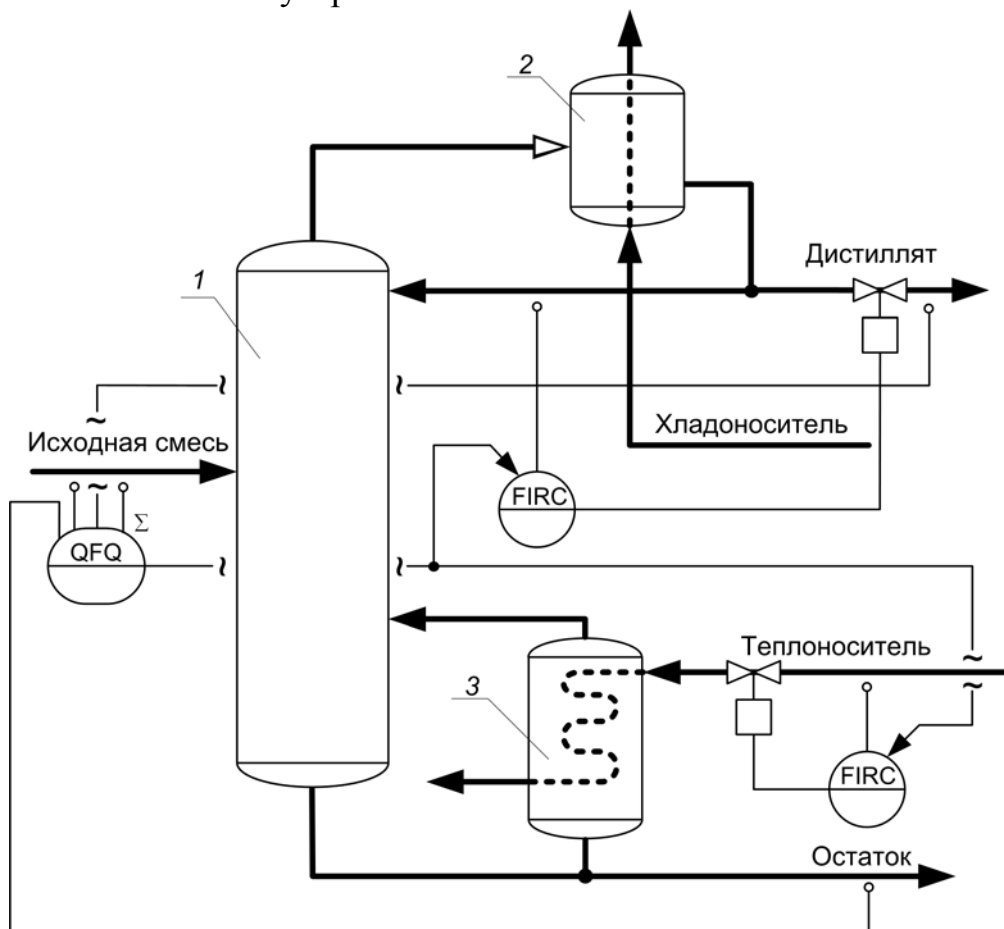


Рисунок 23. ФСА системы регулирования расхода продукта с коррекцией по составу целевых продуктов:

- 1 – колонна;
- 2 – дефлегматор;
- 3 – кипятыльник

Задание №13

Абсорбент, возвращаемый с участка десорбции, может содержать некоторое количество компонентов газовой смеси, что значительно ухудшает качество процесса абсорбции. В этом случае необходимо выводить часть отработанного абсорбента из системы и вводить равную часть свежего. Баланс между расходами отработанного абсорбента и свежего поддерживается с помощью регулятора, действующего на расход сливаемого абсорбента.

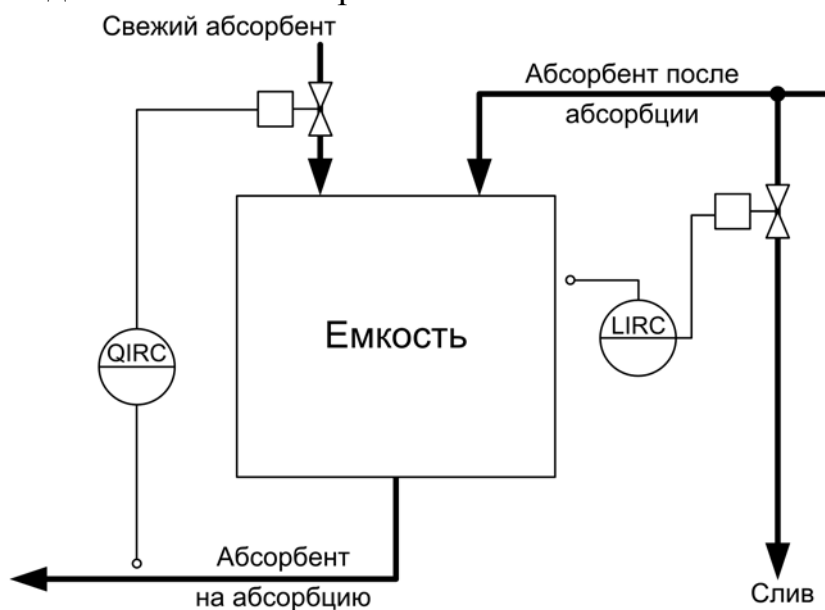


Рисунок 24. ФСА процесса регулирования состава абсорбента в колонне

Задание №14

На верхнюю тарелку адсорбера из дозатора попадает адсорбент, который под действием силы тяжести проваливается с тарелки на тарелку и выводится через низ аппарата. На встречу ему снизу вверх движется газовая смесь. Один из компонентов газовой смеси поглощается частицами адсорбента. Цель управления – поддержание заданной величины остаточной концентрации поглощаемого компонента, что достигается путём изменения расхода адсорбента. Контролю подлежат: расход газовой смеси, конечная концентрация поглощаемого компонента, температура газовой смеси и адсорбента, температура адсорбента на различной высоте колонны, давление в верхней и нижней частях колонны, перепад давления между ними. Сигнализации подлежат: концентрация адсорбируемого компонента в отводимом газе и давление в колонне. При резком возрастании последнего должно срабатывать устройство защиты.

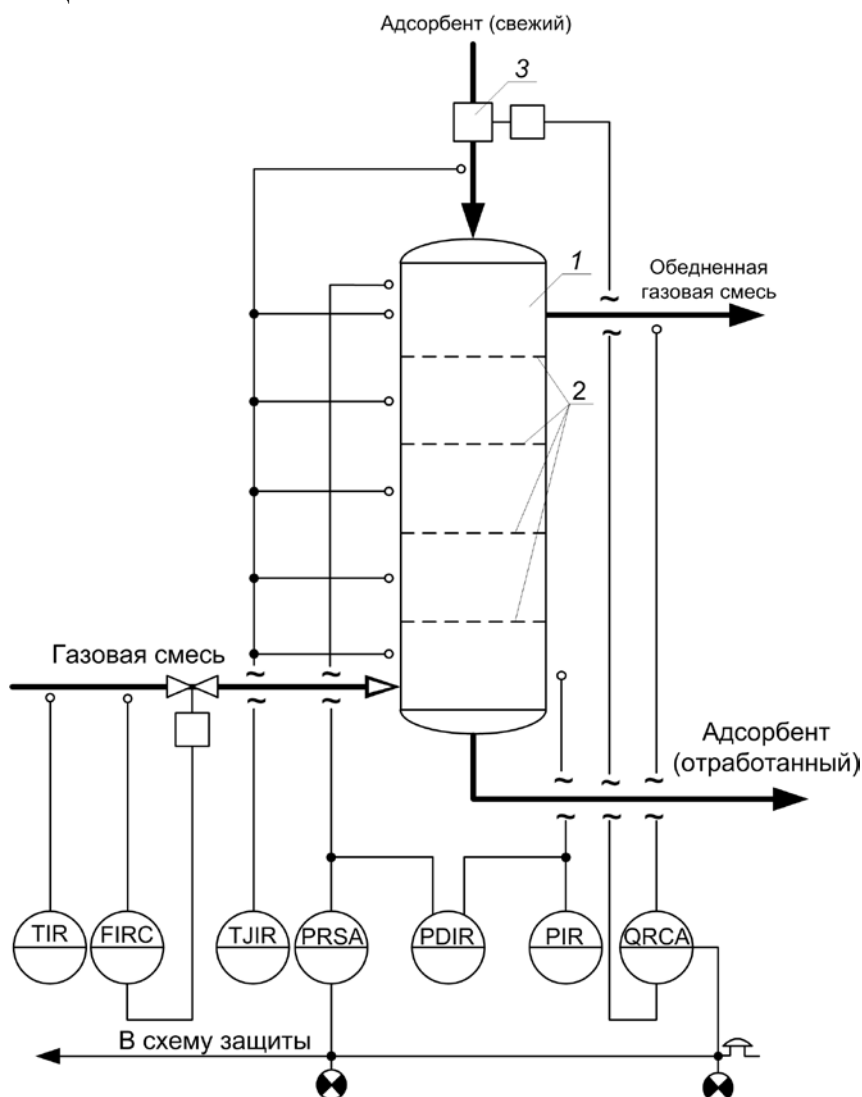


Рисунок 25. ФСА процесса адсорбции:
1 – адсорбционная колонна;
2 – тарелки;
3 – дозатор

Задание №15

Выделение из адсорбента поглощённого вещества проводится в кипящем слое противоточных тарельчатых сорбционных аппаратов. Адсорбент попадает на верхнюю тарелку, а в нижнюю часть после калорифера поступает нагретый воздух. Система регулирования включает в свой состав следующие узлы: регулирования перепада давления и расхода воздуха. Для лучшего выделения поглощённого вещества стабилизируют температуру воздуха на выходе калорифера путём изменения расхода теплоносителя.

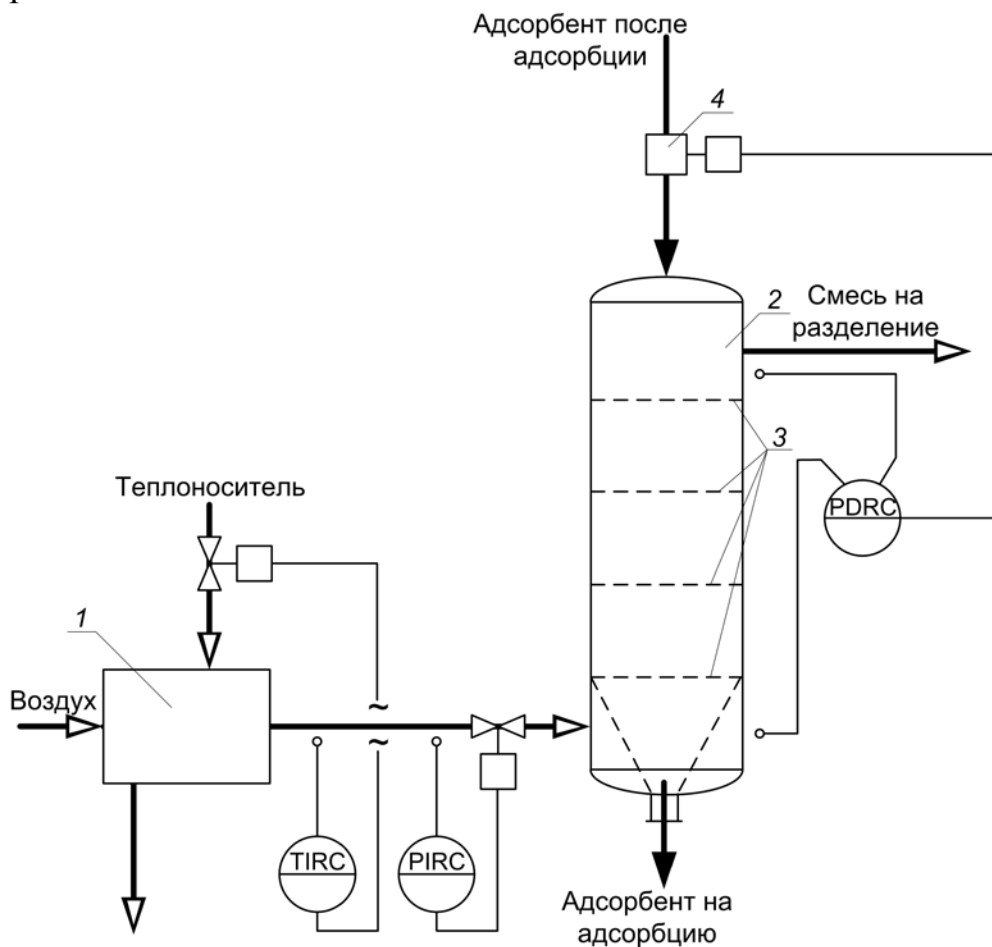


Рисунок 26. ФСА процесса десорбции в кипящем слое:

- 1 – калорифер;
- 2 – десорбционная колонна;
- 3 – тарелки;
- 4 – дозатор

Задание №16

Сушка производится в барабанной прямоточной сушилке, в котором сушильным агентом служат дымовые газы, получаемые в топке. Показатель эффективности процесса – влажность материала на выходе, а регулирующим воздействием на это параметр является количество сушильного агента. Соотношение между расходами топлива и воздуха подаваемого в топку должно быть стабилизировано. Температура сушильного агента на входе в барабан должна стабилизироваться путём изменения расхода вторичного воздуха. Необходимо также стабилизировать расход влажного материала и разрежение в сушилке путём изменения расхода отбираемого сушильного агента. Следует также контролировать расходы топлива, первичного и вторичного воздуха, влажного и сухого материала, температуру сушильного агента на входе и выходе барабана, температуру в сушилке, разрежение в смесительной камере. При значительном отклонении показателя эффективности от заданного значения, опасном повышении температуры сушильного агента на входе в сушилку должна производиться сигнализация, а при останове электродвигателя должна быть прекращена подача материала в сушилку.

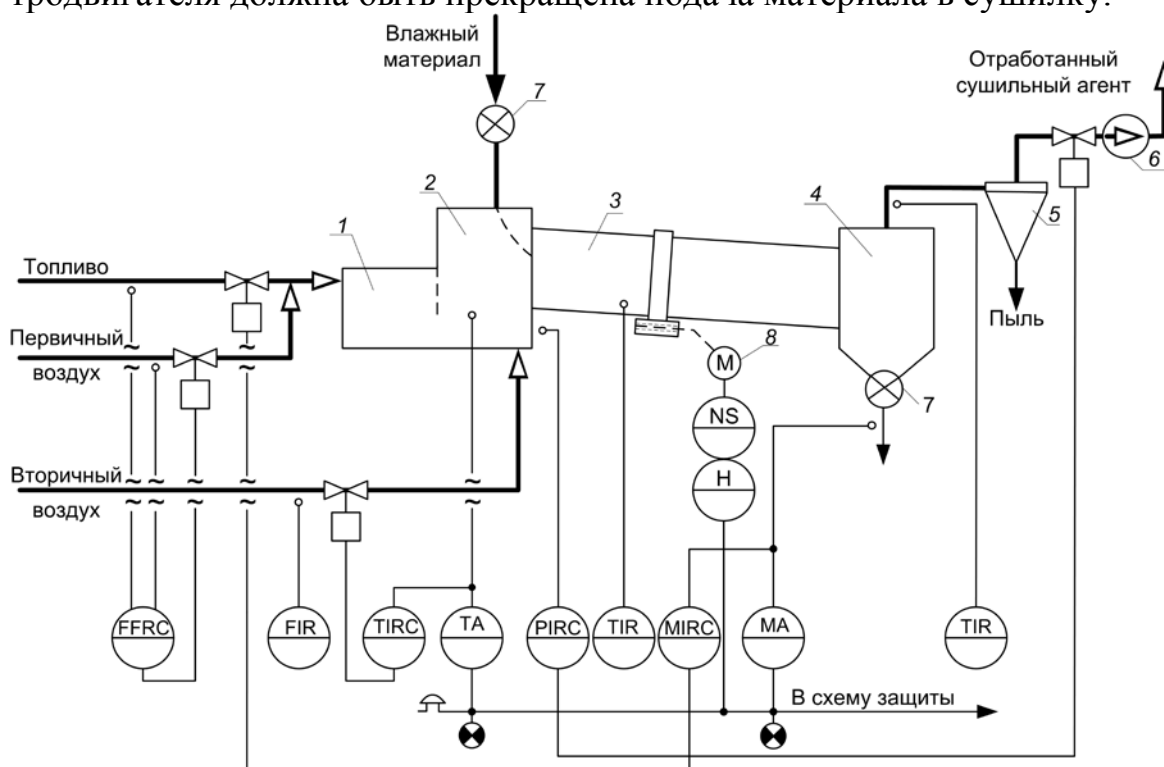


Рисунок 27. ФСА процесса сушки влажного материала: 1 – топка, 2 – смесительная камера, 3 – барабан, 4 – бункер, 5 – циклон, 6 – вентилятор, 7 – автоматический дозатор, 8 – электродвигатель барабана

Задание №17

В противоточной сушилке для предотвращения разложения материала под действием высоких температур в качестве основной регулируемой величины нужно использовать температуру материала на выходе из сушилки и вносить регулирующее воздействие изменением количества сушильного агента. Температура воздуха на входе в барабан регулируется изменением расхода теплоносителя, подаваемого на воздухонагреватель, а влажность изменением расхода рециркулирующего воздуха. Изменение расхода сушильного агента в противоточной сушилке может быть осуществлено и в зависимости от влажности материала, а так же, – от температуры в барабане.

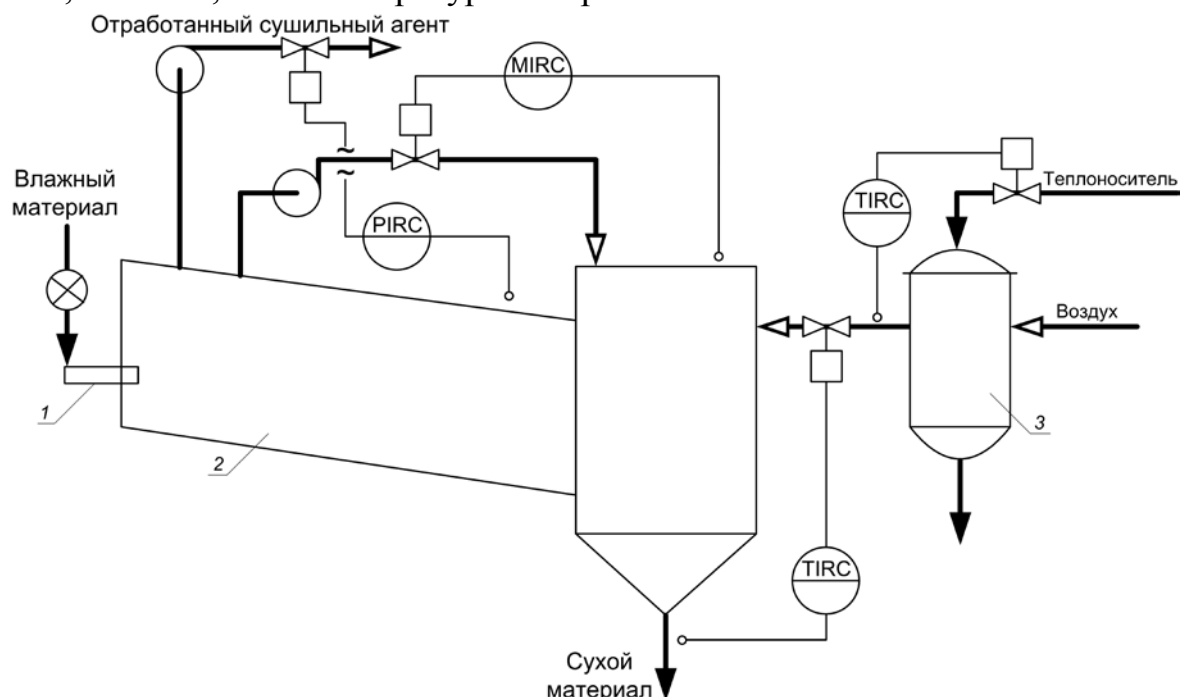


Рисунок 28. ФСА противоточной барабанной сушилки:

- 1 – транспортёр влажного материала;
- 2 – барабан;
- 3 – воздухонагреватель

Задание №18

Регулирование ленточных сушилок подобно барабанным, рассмотренным в задании №17. Стабилизации подлежат влажность сухого материала или конечная температура сушильного агента, температура сушильного агента на входе в сушилку, разрежение в сушилке. Наличие транспортёра позволяет выполнять регулирование уровня влажности материала путём изменения скорости движения ленты транспортёра. При наличии дополнительного нагревателя под транспортёром расход теплоносителя в подогревателе стабилизируется.

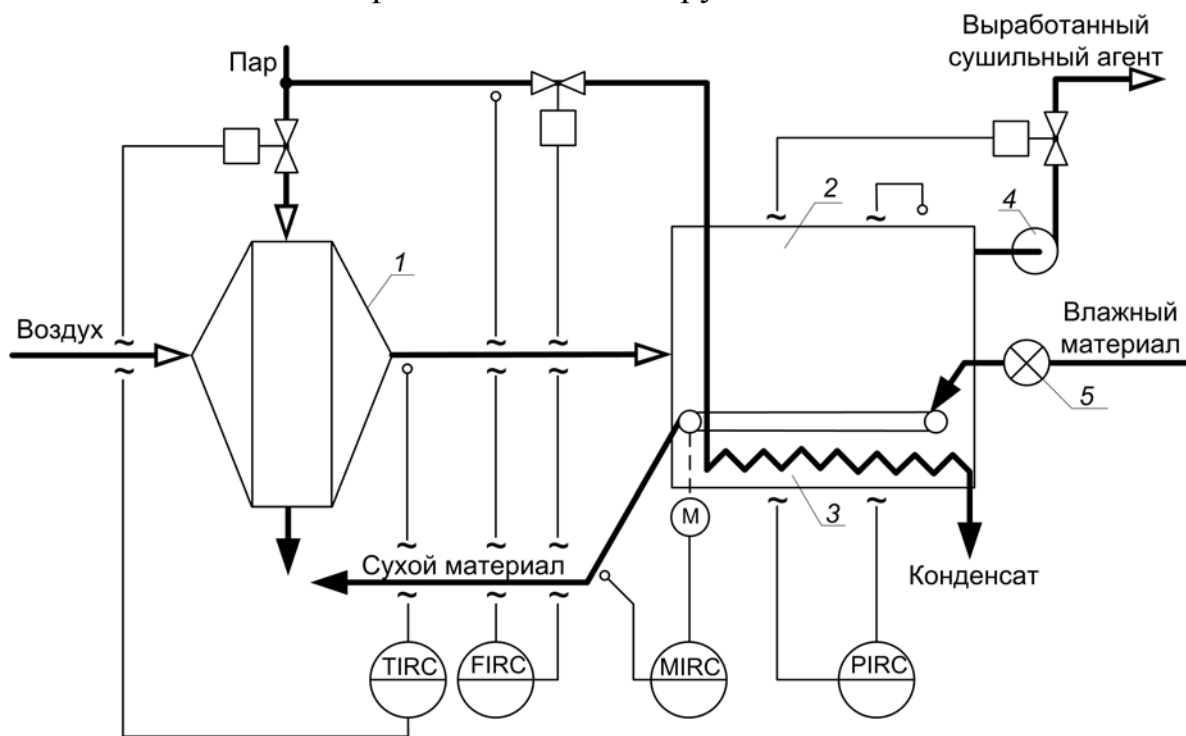


Рисунок 29. ФСА ленточной сушилки:

- 1 – калорифер;
- 2 – сушилка;
- 3 – дополнительный подогреватель;
- 4 – вентилятор;
- 5 – питатель

Задание №19

В струйных сушилках выполняется сушка суспензий различных неорганических соединений (предварительно нагретых в теплообменнике) за счёт распыливания их сушильным агентом. При этом необходимо получить не только заданный уровень влажности, но и заданный гранулометрический состав материалов. Дисперсность распыла определяется в основном соотношением расходов сушильного агента и суспензии. Поэтому к уже известным решениям (задание №16) добавляется узел регулирования размера частиц.

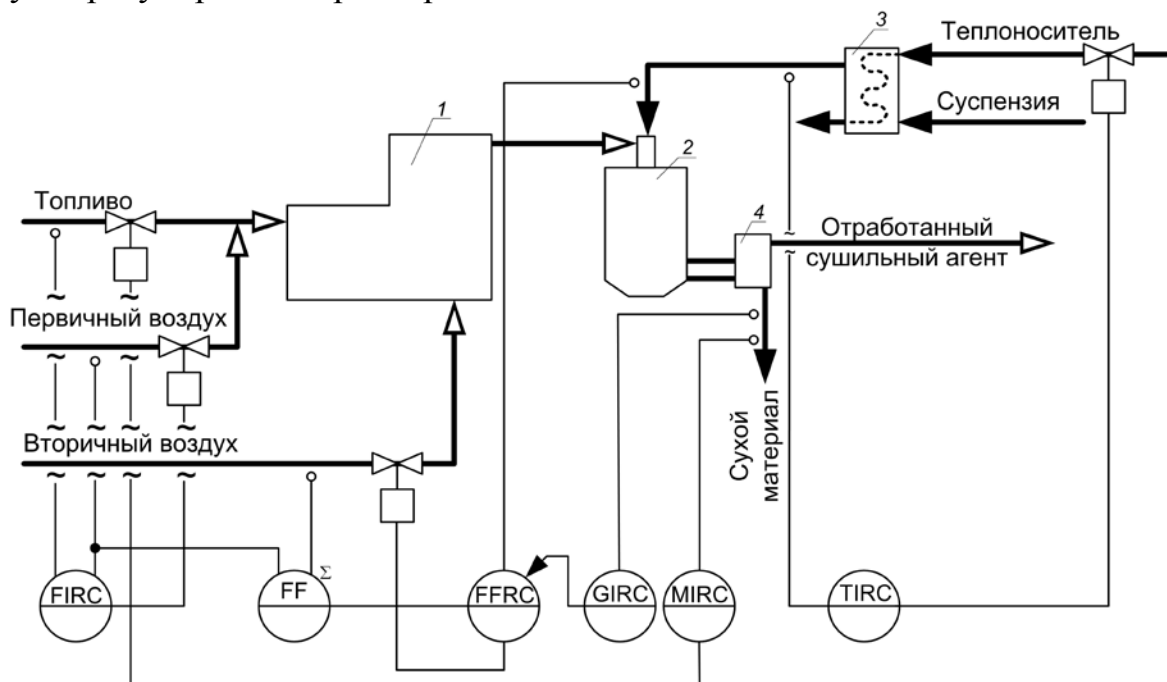


Рисунок 30. ФСА струйной сушиллки:

- 1 – топка;
- 2 – сушиллка;
- 3 – теплообменник;
- 4 – сепаратор;
- G – линейный размер частиц

Задание №20

В сушилках с механическим распылением суспензия распыливается за счёт давления создаваемого на входе механического распылителя (форсунки), которое следует стабилизировать. Все остальные узлы, такие же, как у струйных сушилок, описанных в задании №19. На рисунке представлена схема управления влажностью сухого продукта, путём изменения расхода суспензии, подаваемой в сушилку. Изменение выполняется путём возврата части суспензии с выхода насоса на его вход. В представленной сушилке одновременно выполняется разделение высушенного продукта мешочными фильтрами. Для их регенерации предусмотрен встряхивающий механизм, который управляется по жёстко заданной временной циклограмме.

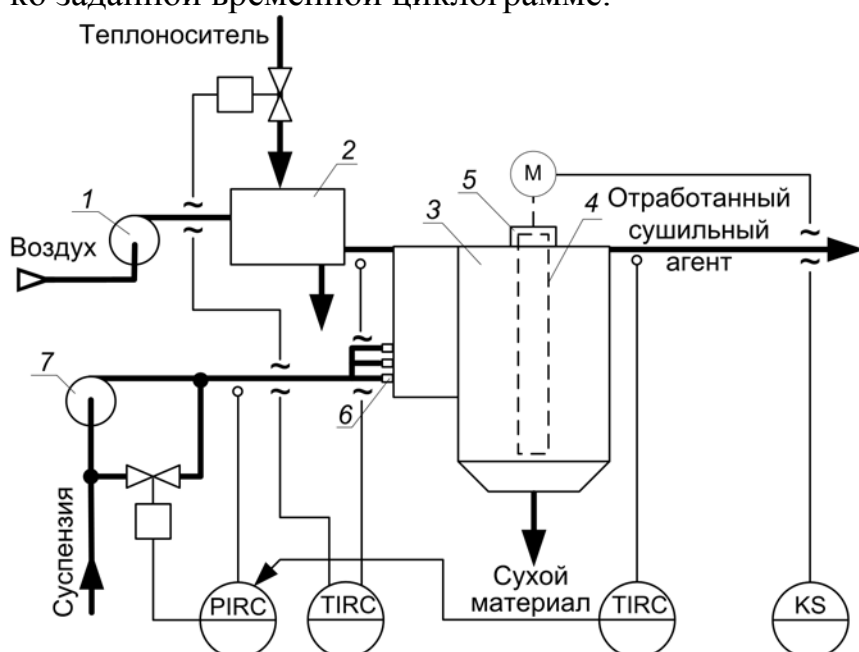


Рисунок 31. ФСА распылительной сушилки:

- 1 – вентилятор;
- 2 – теплообменник;
- 3 – сушилка;
- 4 – мешочный фильтр;
- 5 – встряхивающее устройство;
- 6 – механические распылители (форсунки);
- 7 – питающий насос

Задание №21

При регулировании процессов сушки в радиационных сушилках в качестве основной регулируемой величины используется косвенный показатель – температура поверхности излучателя или же температура отработанного сушильного агента (воздуха). Кроме регулятора температуры излучателя в схеме предусмотрен регулятор соотношения расходов топлива и воздуха и регулятор влажности отработанного сушильного агента.

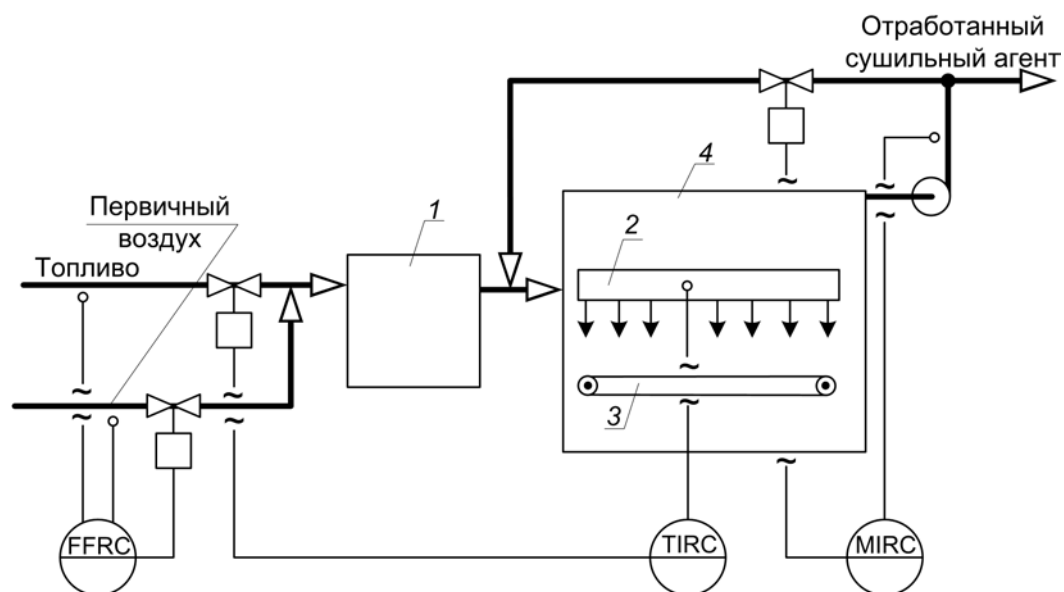


Рисунок 32. ФСА радиационной сушилки:

- 1 – топка;
- 2 – излучатель;
- 3 – конвейер;
- 4 – сушильная камера

Задание №22

Объектом управления является барабанная мельница сухого молота. Показателем эффективности процесса является линейный размер кусков измельчённого материала. Измельчение происходит с использованием металлических шаров, которыми наполнен барабан мельницы. Контролю в данной системе подлежат: расход материала, амплитуда шума создаваемого мельницей, количество потребляемой мельницей энергии.

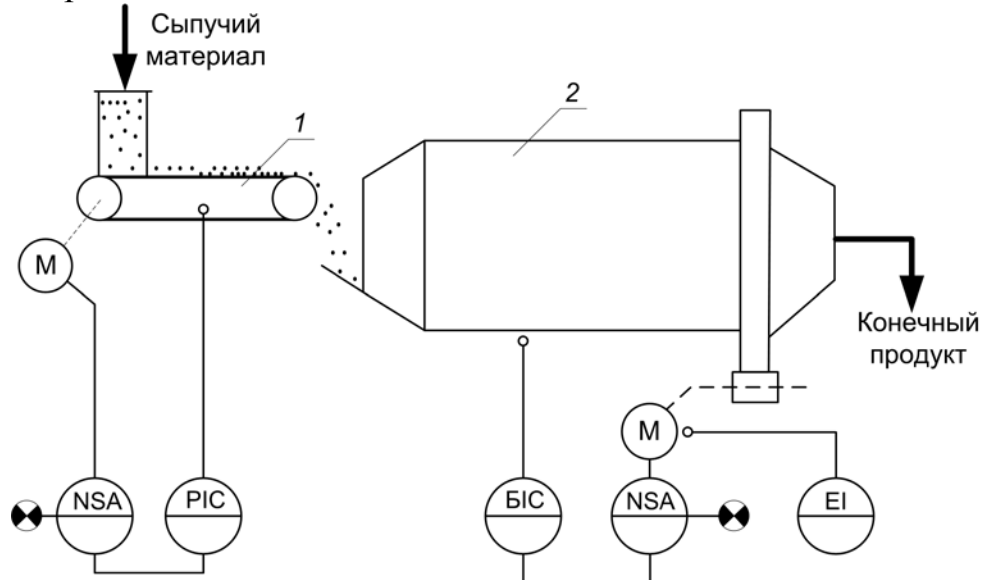


Рисунок 33. ФСА процесса измельчения:

- 1 – ленточный питатель;
- 2 – барабан;
- Б – амплитуда шума;
- Н – аппаратура включения

ПРИЛОЖЕНИЕ В. Практические задания к главе №8

Задание №1

Разработать компьютерную программу в интегрированной среде разработки TRACE MODE 6.06.2, которая позволит, используя текстовый интерфейс пользователя отследить состояние входов платы ввода дискретных сигналов ADVANTECH PCL-733. Проект должен быть разработан для микропроцессорного контроллера, работающего под управлением DOS. Структурная схема лабораторной установки представлена на рисунке 34.

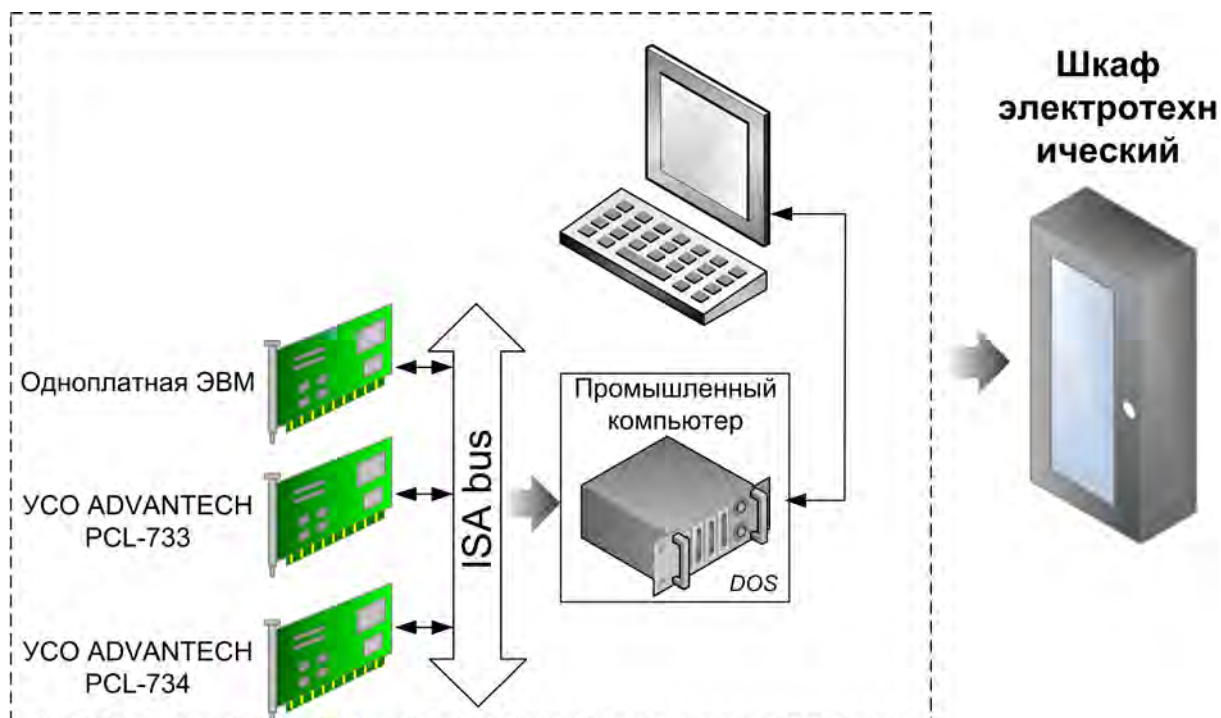


Рисунок 34. Структурная схема лабораторной установки

Рекомендации:

Базовый адрес платы PCL-733 на шине ISA – 0×300.

Базовый адрес платы PCL-734 на шине ISA – 0×310.

Задание №2

В интегрированной среде разработки TRACE MODE 6.06.2 разработать компьютерную программу для микропроцессорного контроллера, работающего под управлением ОС MS Windows. Программа, используя графический интерфейс пользователя, должна выполнять индикацию состояний входов платы ввода дискретных сигналов ADVANTECH PCL-733 установленной в другом микропроцессорном контроллере, работающем под управлением DOS и функции ввода команд управления состоянием выходов платы вывода дискретных сигналов ADVANTECH PCL-734. Структурная схема лабораторной установки представлена на рисунке 35

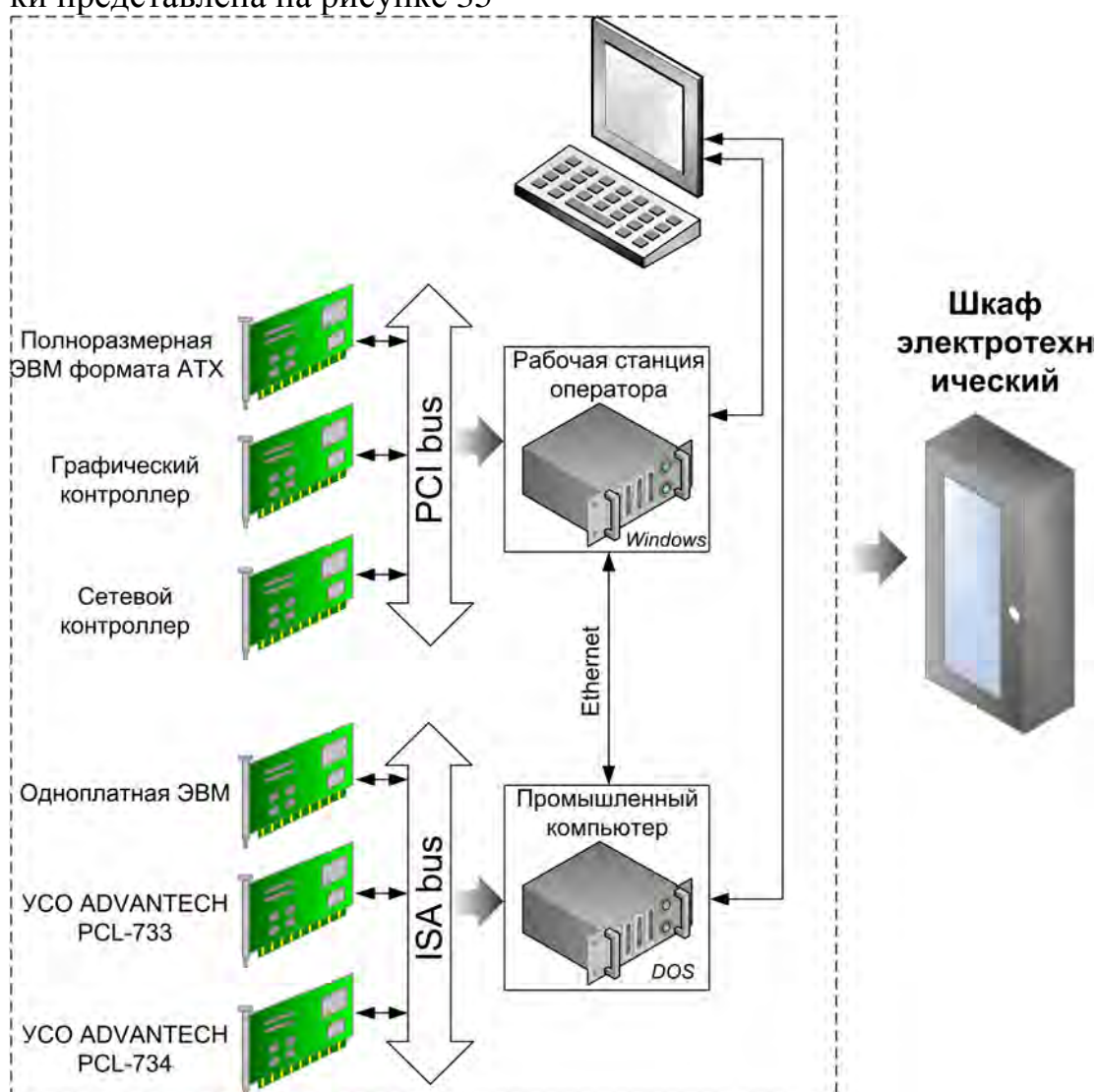


Рисунок 35. Структурная схема лабораторной установки

Рекомендации:

Для каналов вывода данных в составе узла верхнего уровня АСУ настроить Дамп текущих значений в ПЗУ. В качестве IP-адреса использовать: «10.0.0.1» и «10.0.0.4» для RTM и MicroRTM соответственно. Базовый адрес платы PCL-733 на шине ISA – 0×300. Базовый адрес платы PCL-734 на шине ISA – 0×310.

Задание №3

В интегрированной среде разработки TRACE MODE 6.06.2 разработать компьютерную программу для ЭВМ оператора, которая работает под управлением ОС MS Windows. Компьютерная программа должна выполнять следующие функции:

1. ввод команд управления с использованием функциональной клавиатуры;
2. индикацию состояния оборудования с использованием элементов индикации в составе функциональной клавиатуры.

В составе программы необходимо разработать мнемосхему виртуального технологического процесса (например, процесса нагрева продукта в резервуаре с использованием некоторого теплоносителя) или использовать одну из схем Приложения Б. Функцию ввода команды управления с использованием кнопочных блоков в составе функциональной клавиатуры реализовать по схеме коммутации выходного сигнала с фиксацией, с учётом того, что кнопки ФК имеют нормально замкнуты контакт. Индикацию сигналов контроля и факта ввода и обработки команд в составе графического интерфейса пользователя реализовать в соответствии с ГОСТ Р МЭК 60073-2000. Структура лабораторного стенда представлена на рисунке 8.18. Таблица соответствия входов и выходов платы УСО подключенным к ней элементам ввода команд управления и индикации сигналов характеризующих состояние оборудования представлена в таблице 8.2.

ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ И ОПРЕДЕЛЕНИЯ

АСУТП – автоматизированная система управления технологическим процессом.

SCADA-система – (от англ. *Supervisory control and data acquisition*, что означает Система диспетчерского контроля и сбора данных). Данное понятие в общем виде обозначает программно-технический комплекс, выполняющий задачи контроля и управления в АСУ, однако его часто применяют к программным комплексам, используемым для проектирования прикладного ПО АСУ. В настоящей работе будем принимать словосочетание SCADA TRACE MODE как САПР TRACE MODE, с помощью которой выполняется разработка прикладного ПО АСУ участвующего в процессе диспетчерского контроля и управления данными.

СИСТЕМА (от греч. *συστήμα* – составленный) – совокупность объектов и связей между ними, выделенных из среды на определённое время и с определённой целью. В TRACE MODE – это ещё головной слой в структуре создаваемого программного обеспечения.

ТЕХНОЛОГИЧЕСКИЙ ПРОЦЕСС – совокупность последовательно выполняемых технологических операций, образующих вместе единый процесс преобразования исходных материалов в конечный продукт.

ТЕХНОЛОГИЧЕСКАЯ ОПЕРАЦИЯ – часть технологического процесса, непрерывно выполняемая на одном рабочем месте.

ТЕХНОЛОГИЧЕСКИЙ РЕЖИМ – последовательность протекающих технологических процессов производства и комплекс требований, направленных на обеспечение технологической дисциплины.

IDE или **ИС** (от англ. *Integrated development environmen*) – интегрированная среда разработки программного обеспечения.

ПРОЕКТ – набор файлов прикладного программного обеспечения созданного с использованием ИС. В Проекте зашифрован математический аппарат, алгоритмы управления данными в АСУТП, графический интерфейс оператора, методы и алгоритмы взаимодействия элементов автоматики между собой. Не стоит путать понятие «Проект» в рамках SCADA TRACE MODE с понятием проект в общем смысле (проект (от лат. *projectus* – выступающий вперёд,

выдающийся) – уникальная деятельность, в отличие от операции, направленная на достижение некоторой цели, имеющая определённые временные границы и требования к конечному результату).

RTM (от англ. *Real time monitor*) – Монитор реального времени (МРВ), специальная программа-интерпретатор Проектов TRACE MODE.

УСО (устройство связи с объектом) – специализированные электронные устройства, позволяющие ЭВМ взаимодействовать на электрическом и информационном уровне с технологическим управляющим оборудованием АСУ.

КВИТИРОВАНИЕ – процесс, в результате которого два независимых друг от друга объекта аппаратного или программного обеспечения координируют свои сигналы и получают возможность работать совместно. Методы передачи данных с квитированием используются для согласования скорости передачи информации, надёжной синхронизации. Подтверждение события оператором или программой является примером квитирования.

МНЕМОСХЕМА – это совокупность графических или физических индикаторов и статических изображений, используемых для графического представления объекта управления и элементов АСУ, индикации сигналов контроля и ввода команд управления. Наряду с элементами мнемосхем в составе прикладного ПО используют элементы человеко-машинного интерфейса пользователя: графические и физические.

ТОПОЛОГИЯ (от греч. *τόπος* – место). В разделе техники термин применяется для описания схем расположения и соединения объектов. Например, сетевая топология. В TRACE MODE 6.06.2 термин используется для обозначения структуры программы при её проектировании с учётом структуры предприятия.

АРМ оператора (автоматизированное рабочее место оператора) – совокупность программных и технических средств необходимых для выполнения функций контроля и управления технологическими элементами АСУ оператором данной АСУ.

МОДЕЛЬ (от лат. *modulus* – мера, образец) – некоторый материальный или мысленно представляемый объект или явление, с достаточной точностью замещающий оригинальный объект или явление (т.е.

сохраняя только требуемые его свойства). Существует также понятие математическая, натурная модель и т.д.

ДИНАМИЗАЦИЯ – процесс изменения (например, программирования) свойств объекта с целью добавления ему функции изменения собственного состояния или характеристик в течение времени в зависимости от контролируемой характеристики другого объекта или процесса.

GUI (графический интерфейс пользователя, оператора) – совокупность динамических графических элементов необходимых для взаимодействия оператора с ПЭВМ.

КОМПЬЮТЕРНАЯ ПРОГРАММА – последовательность инструкций, предназначенная для исполнения устройством управления ЭВМ.

САУ – система автоматического управления.

САР – система автоматического регулирования.

САПР – система автоматизированного проектирования. В случае проектирования программного обеспечения используется понятие CASE (от англ. *Computer-aided software engineering*) – набор инструментов и методов программной инженерии, используемых для проектирования программного обеспечения, который помогает обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов

ПОРТ – Логическое и физическое соединение информационных каналов некоторого разъёма с регистрами ЭВМ можно назвать портом компьютера. Регистры являются отправной точкой данных, с которыми в дальнейшем работает системное и прикладное ПО. В рамках TRACE MODE, как программы, портом будем называть сгруппированные по функциональному и количественному признаку группы информационных потоков формируемых ЭВМ или периферийным устройством в регистрах платы УСО.

ИНТЕРФЕЙС – (от англ. *Interface*) – поверхность раздела, перегородка. Совокупность средств и методов взаимодействия между элементами системы. Если в качестве системы использовать человека и ЭВМ, то интерфейс между ними называется человеко-машинным. Он может быть текстовым или графическим и позво-

ляет обоим объектам системы понимать друг друга и взаимодействовать.

ДРАЙВЕР – (от англ. *Driver*) – объект системы, направляющий движение другого объекта. Компьютерная программа позволяющая операционной системе контролировать работу периферийных устройств.

DDE (от англ. *Data dynamic exchange*) – динамический обмен данными. Механизм обмена данными между приложениями ОС MS Windows.

OLE (от англ. *Object linking and embedding*) – программная технология внедрения объектов-программ в другие документы и программы ОС MS Windows.

СУБД (Система управления базой данных) – специализированная программа управления данными в базе данных. Передача команд управления осуществляется оператором с использованием SQL-запросов.

SQL (от англ. *Structured query language*) – язык структурированных запросов. Универсальный информационно-логический язык создания и модификации команд управления данными в реляционных базах данных.

Учебное издание

МЕЗЕНЦЕВ Антон Алексеевич
ПАВЛОВ Вадим Михайлович

САПР TRACE MODE 6

Учебно-методическое пособие


Научный редактор *доктор технических наук,*
профессор С.Н. Ливенцов
Редактор *И.О. Фамилия*
Компьютерная верстка *И.О. Фамилия*
Дизайн обложки *И.О. Фамилия*

Подписано к печати 05.11.2012. Формат 60x84/16. Бумага «Снегурочка».
Печать XEROX. Усл.печ.л. 9,01. Уч.-изд.л. 8,16.
Заказ . Тираж 100 экз.



Национальный исследовательский Томский политехнический университет
Система менеджмента качества
Издательства Томского политехнического университета сертифицирована
NATIONAL QUALITY ASSURANCE по стандарту BS EN ISO 9001:2008



ИЗДАТЕЛЬСТВО  **ТПУ**. 634050, г. Томск, пр. Ленина, 30
Тел./факс: 8(3822)56-35-35, www.tpu.ru