

Министерство образования и науки  
Российской Федерации

Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Тихookeанский государственный университет»

**В. Э. Иванов, Г. К. Конопелько**

**Основы проектирования микросистем на базе  
Siemens LOGO! и S7-200**

Утверждено издательско-библиотечным советом университета в качестве  
учебного пособия

Хабаровск  
Издательство ТОГУ  
2016

УДК 681.513.2 (075.8)  
ББК 3972.53-02я7  
И201

**Р е ц е н з е н т ы :**

А. И. Годяев, д-р техн. наук, проф. заведующий кафедрой автоматики и телемеханики  
Дальневосточного государственного университета путей сообщения  
И. А. Кривошеев, д-р техн. наук, проф. заведующий лабораторией информационных  
технологий вычислительного центра ДВО РАН

**Н а у ч н ы й р е д а к т о р**

Чье Ен Ун, д-р. техн. наук, проф. кафедры «Автоматика и системотехника»  
Тихоокеанского государственного университета

**Иванов, В. Э.**

И201 Основы проектирования микросистем на базе SIEMENS LOGO! и S7-200 / В.Э. Иванов, Г.К. Конопелько. – Хабаровск : Изд-во Тихоокеан. гос. ун-та, 2016. – 160 с.

ISBN 978-5-7389-2103-2

В учебном пособии рассмотрены основы программирования и ввода в эксплуатацию микросистем на базе промышленных логических контроллеров LOGO! и S7-200. Описаны способы адресации и типы данных, используемых в промышленной автоматике, основы промышленных языков программирования LAD и FBD, форматы данных и способы обработки аналоговых величин.

Учебное пособие предназначено для студентов, изучающих дисциплины «Проектирование информационно-управляющих систем», «Автоматизированные информационно-управляющие системы», «Вычислительные машины, системы и сети».

УДК 681.513.2 (075.8)  
ББК 3972.53-02я7

**ISBN 978-5-7389-2103-2**

© Иванов В.Э., Конопелько Г.К., 2016  
© Тихоокеанский государственный  
университет, 2016

# ОГЛАВЛЕНИЕ

<b>ВВЕДЕНИЕ.....</b>	5
<b>1. АРХИТЕКТУРА ПЛК LOGO!.....</b>	8
1.1. Подключение к LOGO! внешних цепей.....	12
1.2. Адресация входов и выходов.....	13
<b>2. ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ LOGO!.....</b>	15
2.1. Способы и средства программирования LOGO!.....	15
2.2. Действия LOGO! при сбросе и восстановлении питания.....	16
2.3. Обзор меню LOGO!.....	18
2.4. Переход от принципиальной схемы к блок-диаграмме.....	20
2.5. Порядок ввода блок-диаграммы с помощью встроенного дисплея...	22
2.6. Программирование с помощью LOGO! Soft Comfort.....	25
2.7. Варианты заданий.....	28
<b>3. ВВОД-ВЫВОД И БАЗОВЫЕ ФУНКЦИИ LOGO!.....</b>	30
3.1. Базовые функции и их особенности.....	30
3.2. Вывод текстовых сообщений.....	33
3.3. Пример использования базовых функций.....	37
3.4. Варианты заданий.....	39
<b>4. СПЕЦИАЛЬНЫЕ ФУНКЦИИ LOGO!.....</b>	41
4.1. Специальные функции и их применение.....	41
4.2. Пример использования специальных функций.....	48
4.3. Варианты заданий.....	52
<b>5. ФУНКЦИИ РАБОТЫ С АНАЛОГОВЫМИ ВХОДАМИ LOGO!.....</b>	57
5.1. Подключение аналоговых датчиков к LOGO!.....	57
5.2. Функции работы с аналоговыми входами.....	58
5.3. Масштабирование аналоговых величин.....	60
5.4. Варианты заданий.....	64
<b>6. АРХИТЕКТУРА И ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ S7-200...70</b>	70
6.1. Особенности архитектуры S7-200.....	70

6.2. Технология программирования S7-200.....	77
6.3. Создание простой программы ПЛК и ее отладка.....	83
<b>7. ДОСТУП К ДАННЫМ ПЛК.....</b>	<b>93</b>
7.1. Типы данных и способы их размещения в памяти.....	93
7.2. Структура программы обработки данных на языке LAD.....	99
7.3. Команды преобразования и перемещения данных.....	104
7.4. Варианты заданий.....	108
<b>8. АНАЛОГОВЫЕ МОДУЛИ .....</b>	<b>110</b>
8.1. Область применения аналоговых модулей.....	110
8.2. Способы подключения источников тока и напряжения.....	113
8.3. Способы представления аналоговых величин.....	117
8.4. Подключение и диагностика аналоговых модулей.....	118
8.5. Адресация аналоговых входов и выходов.....	121
8.6. Формат данных аналоговых входов и выходов.....	124
8.7. Пример программы обработки аналоговых величин.....	125
8.8. Варианты заданий.....	128
<b>9. ИЗМЕРЕНИЕ ТЕМПЕРАТУРЫ.....</b>	<b>132</b>
9.1. Измерительные преобразователи «температура-ток».....	132
9.2. Термометры сопротивления.....	139
9.3. Формат данных и схема подключения модуля EM 231 RTD.....	143
9.4. Варианты заданий.....	147
<b>БИБЛИОГРАФИЯ.....</b>	<b>150</b>
<b>ПРИЛОЖЕНИЕ 1.....</b>	<b>151</b>
<b>ПРИЛОЖЕНИЕ 2.....</b>	<b>152</b>
<b>ПРИЛОЖЕНИЕ 3.....</b>	<b>154</b>
<b>ПРИЛОЖЕНИЕ 4.....</b>	<b>156</b>
<b>ПРИЛОЖЕНИЕ 5.....</b>	<b>157</b>
<b>ПРИЛОЖЕНИЕ 6.....</b>	<b>159</b>

## **ВВЕДЕНИЕ**

Микросистемы SIMATIC – это спектр программных и аппаратных продуктов, обладающих невысокой стоимостью и ориентированных на построение наиболее простых устройств и систем автоматического управления различного назначения. Они находят применение в системах промышленной автоматизации, а также в системах автоматизации зданий. Микросистемы SIMATIC объединяют в своем составе две линейки продуктов:

- Логические модули LOGO! для построения наиболее простых программируемых устройств автоматического управления.
- Программируемые контроллеры S7-200 для построения относительно простых систем автоматического управления, работающих автономно или интегрируемых в более сложные системы автоматизации.

Универсальные логические модули LOGO! являются компактными функционально законченными изделиями, предназначенными для решения наиболее простых задач автоматизации с логической обработкой информации. Семейство включает в свой состав универсальные логические модули LOGO! Basic и LOGO! Pure, модули ввода-вывода дискретных и аналоговых сигналов, коммуникационные модули, модули блоков питания LOGO! Power и другие модули. Система ввода-вывода логических модулей LOGO! Basic и LOGO! Pure легко адаптируется к требованиям решаемой задачи. Максимальная конфигурация позволяет обслуживать до 24 дискретных входов, до 16 дискретных выходов, до 8 аналоговых входов и до 2 аналоговых выходов.

Программирование логических модулей LOGO! может выполняться:

- Непосредственно с помощью встроенной клавиатуры (в LOGO! Basic).
- С компьютера, оснащенного программным обеспечением LOGO! Soft Comfort.
- Установкой запрограммированного модуля памяти.

Дисплей модуля используется как на этапе программирования, так и на этапе его эксплуатации. В процессе эксплуатации на дисплей могут выводиться простейшие оперативные сообщения (русский язык поддерживается только в версиях модулей 0BA6 и выше). При необходимости к логическому модулю LOGO! (от версии 0BA6 и выше) может подключаться внешний текстовый дисплей LOGO! TD. Модули LOGO! Pure являются полными функциональными аналогами модулей LOGO! Basic, но не имеют встроенной клавиатуры и дисплея.

Логические модули LOGO! имеют функциональные аналоги в семействе SIPLUS, которые способны сохранять работоспособность:

- В диапазоне температур от -25 до +70 °C или от -40 до +70 °C.
- В условиях появления конденсата и обледенения печатных плат.
- При наличии агрессивных примесей в атмосфере.
- В условиях более сильных вибрационных и ударных нагрузок.

Программируемые контроллеры SIMATIC S7-200 предназначены для построения относительно простых и недорогих систем автоматического управления и могут использоваться для замены существующих релейно-контактных систем. Семейство включает в свой состав модули центральных процессоров, модули ввода-вывода дискретных и аналоговых сигналов, технологические, коммуникационные и другие модули. Контроллеры поддерживают мощную систему команд и способны выполнять логические операции, математические операции с фиксированной и плавающей точкой, поддерживать алгоритмы ПИД регулирования и позиционирования и т.д. Одна система на основе S7-200 способна обслуживать до 256 дискретных и до 32 аналоговых каналов ввода-вывода. Контроллеры способны поддерживать обмен данными через промышленные сети PPI (Point to Point Interface), MPI (MultiPoint Interface), Industrial Ethernet, а также через Internet и системы модемной связи. Кроме того, они способны работать в системах распределенного ввода-вывода на основе сетей AS-Interface и PROFIBUS DP.

Для решения задач человека-машинного интерфейса с программируемыми контроллерами S7-200 может использоваться целый ряд текстовых дисплеев и панелей операторов. Программирование контроллеров SIMATIC S7-200, а также конфигурирование текстовых дисплеев SIMATIC TD 200, TD100C, TD 200C и TD 400C выполняется с помощью пакета STEP 7-Micro/WIN. Разработка программ может выполняться на языках LAD, FBD и STL. Для расширения функциональных возможностей пакета STEP 7-Micro/WIN можно использовать целый ряд дополнительных пакетов промышленного программного обеспечения.

В данный момент линейка микросистем существенно переработана. Семейство ПЛК S7-200 пополнилось новыми моделями ПЛК S7-1200, отличающимися от прежних количеством интерфейсов, общедоступных протоколов и составом модулей расширения. Изменился и подход к проектированию – программирование и отладка проектов осуществляется в программном пакете TIA Portal, который является общим интегратором для всех продуктов Siemens. Семейство LOGO! также непрерывно совершенствуется в направлении наращивания функциональных возможностей.

Несмотря на появление новых конкурентоспособных моделей ПЛК, микросистемы на базе LOGO! И S7-200 не теряют актуальности. В данном учебном пособии рассмотрены базовые основы программирования и ввода в эксплуатацию наиболее часто используемых моделей LOGO! И S7-200. Освоение архитектуры ПЛК, основ программирования на промышленных языках LAD и FBD – это базовые знания для инженеров-разработчиков современных АСУ ТП.

# **1. АРХИТЕКТУРА ПЛК LOGO!**

Логические модули LOGO! являются компактными функционально законченными универсальными изделиями. Они предназначены для построения простейших устройств автоматики с логической обработкой информации. Примерами могут служить реле времени, системы ограничения доступа, управления воротами и т. д. Алгоритм функционирования модулей задается программой, составленной из набора встроенных функций. Программирование модулей LOGO! может производиться с их клавиатуры без использования дополнительного программного обеспечения. Стоимостные показатели модулей настолько низки, что их применение может оказаться экономически целесообразным даже в случае замены устройств, включающих в свой состав 2 многофункциональных реле времени или 2 таймера и 3-4 промежуточных реле.

Таким образом, модуль LOGO!, оснащенный по необходимости дополнительными модулями расширения, представляет собой интеллектуальное реле (или группу реле, связанных друг с другом), логическая функция которого может быть изменена путем перепрограммирования. При этом программу LOGO! можно сравнить с одной из схем, созданных в среде моделирования и отладки электронных устройств (например, MicroCap, LabView), работающей в реальном времени и с физическими входами и выходами. Такое решение экономит время на разработку, монтаж и отладку оборудования, так как алгоритм функционирования системы может быть эмулирован в реальном времени даже без использования подключенного оборудования. Кроме того, использование LOGO! вместо стандартных схем релейной автоматики существенно повышает надежность системы в целом и делает ее более компактной.

С помощью LOGO! можно решить ряд технических задач. Микросистемы на основе LOGO! находят применение при разработке

автоматизированных систем электрооборудования жилых помещений (например, освещение лестничных клеток, внешнее освещение, тенты, жалюзи, освещение витрин магазинов и т. д.), в коммутационных шкафах, в управлении машинами и аппаратами (например, системы управления воротами, вентиляционные системы или насосы для хозяйственной воды и многое другое). LOGO! можно использовать также для специальных систем управления в оранжереях и теплицах, для предварительной обработки сигналов управления и при подключении коммуникационного модуля (например, AS-i) для децентрализованного управления машинами и процессами на месте. Имеются специальные варианты без блока управления и индикации для серийных приложений в микромашиностроении, аппаратостроении и шкафах управления.

Основным отличием LOGO! от других типов ПЛК является возможность работы исключительно с данными булевого типа. Это исключает в системе команд LOGO! функции математической обработки входных величин. Исключением являются функции работы с аналоговыми входами, где под обработкой понимается масштабирование входной величины для отображения ее на встроенном дисплее. Однако значение аналоговой величины используется исключительно для операций сравнения (например, в версии 0BA3) с использованием булевого результата.

В зависимости от конкретных требований к системе управления фирма SIEMENS предлагает различные варианты исполнения LOGO!. Так, например, имеются модули со встроенным или отдельным блоком питания, со встроенным индикатором для программирования и без него, наличием или отсутствием встроенного модуля связи, аналоговыми входами и т. д. Это позволяет найти оптимальное решение для конкретной задачи управления. Отличаются LOGO! также и поколением производства – версия того или конкретного устройства маркируется условным обозначением 0BAx. (например, 0BA0 – 0BA7). Как правило, следующее поколение LOGO! отличается от предыдущего расширенными ресурсами, добавлением новых

функций, изменением и дополнением существующих. В табл. 1.1 приведены основные варианты исполнения LOGO!.

Таблица 1.1

Варианты исполнения LOGO!

Обозначение	Напряжение питания	Входы	Выходы	Свойства
LOGO! 12/24RC	=12/24 В	8 цифровых	4 релейных 230В, 10А	
LOGO! 24	=24 В	»	4 релейных 24В, 0,3А	без часов
LOGO! 24RC	~ 24 В	»	4 релейных 230В, 10А	
LOGO! 230RC	~/= 115...240 В	»	»	
LOGO! 12/24RCо	= 12/24 В	»	»	без дисплея и клавиатуры
LOGO! 24RCо	~ 24 В	»	»	без дисплея и клавиатуры
LOGO! 230RCо	~/=115...240 В	»	»	без дисплея и клавиатуры

Обозначение LOGO! содержит информацию о его различных встроенных характеристиках:

- 12 – вариант на 12 В постоянного тока;
- 24 – вариант на 24 В постоянного тока;
- 230 – вариант на 115/240 В переменного тока;
- R – релейные выходы (без R: транзисторные выходы);
- С – встроенный часовой выключатель на 7 дней;
- О – вариант без дисплея;
- DM – цифровой модуль;
- АМ – аналоговый модуль;
- FM – функциональный модуль (например, ASi).

Для расширения функциональных возможностей базового модуля предусмотрены модули ввода дополнительных дискретных сигналов, модули AS-интерфейса, аналоговых входов и выходов. Подключение дополнительного модуля не требует затрат системных ресурсов и изменения программы – наличие ссылки на ресурс модуля уже предусмотрено в программе. В табл. 1.2 показаны основные возможности и обозначения модулей расширения.

Таблица 1.2

Модули расширения для LOGO!

Обозначение	Напряжение питания	Входы	Выходы
LOGO! DM 8 12/24 R	= 12/24 В	4 цифровых	4 релейных
LOGO! DM 8 24	= 24 В	4 цифровых	4 транзисторных
LOGO! DM 8 230R	~/= 115...240 В	4 цифровых	4 релейных
		2 аналоговых	
LOGO! AM 2	= 12/24 В	0-10 В или 0-20mA	нет

Монтаж LOGO производится на стандартную DIN-рейку. Модули расширения имеют тот же дизайн и подключаются базовому модулю через специальный боковой разъем, предназначенный для обмена данными и подаче на модуль напряжения питания. Внешний вид модуля LOGO! с подключенными модулями расширения на примере LOGO!24RC 0BAZ показан на рис.1.1.

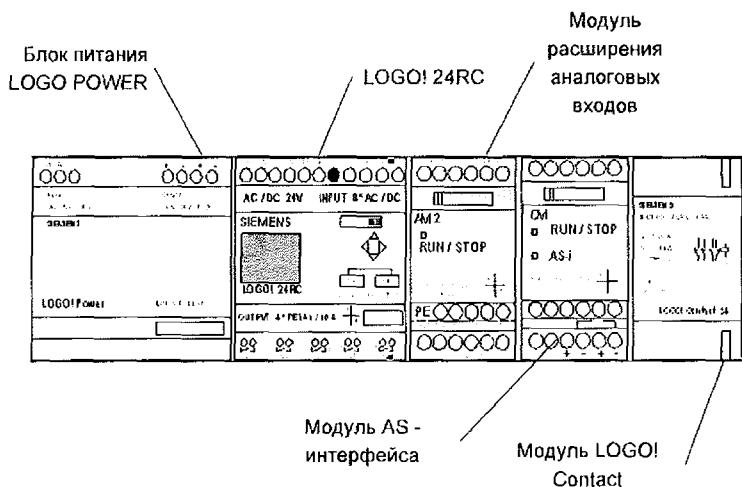


Рис.1.1. Внешний вид LOGO!24RC с дополнительными модулями

## 1.1. Подключение к LOGO! внешних цепей

Любой модуль LOGO! оперирует с данными, представленными исключительно в битовом формате (не считая аналоговых входов). Это обусловлено спецификой этого устройства, как простейшего управляющего контроллера. Поэтому основными устройствами ввода для него являются контактные датчики, выключатели, кнопки. Выходное управляющее воздействие для LOGO! – это сигнал на включение или выключение исполнительного устройства, поэтому выходами являются нормально разомкнутые контакты реле (или транзисторные ключи). На рис. 1.2 показана типовая схема подключения контактных датчиков и нагрузки к LOGO!.

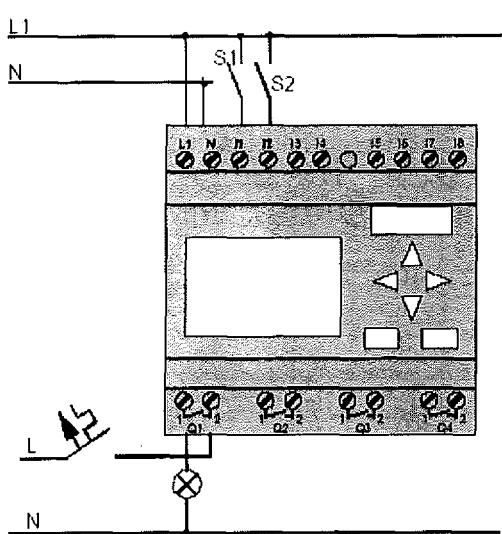


Рис 1.2. Пример подключения датчиков и нагрузки к LOGO!

Обозначение L1, как правило, соответствует выводу источника напряжения питания, который не связан с корпусной землей, N – общий провод включены между полюсом L1 и клеммой цифрового входа. Никаких мер по согласованию уровней напряжения и устраниению дребезга не принимается, так как согласующее устройство входит в состав LOGO!.

Нагрузка подключена через контакты реле, входящее в состав контроллера, последовательно с устройством защиты. Такое соединение не имеет особенностей и считается типовым. При этом источник питания нагрузки может быть гальванически развязан от схемы контроллера. Например, при напряжении питания LOGO! 24 В нагрузка может быть подключена через контакт реле к источнику питания промышленной сети 220 В.

## 1.2. Адресация входов и выходов

Модуль LOGO! поддерживает до 24 цифровых входов, 8 аналоговых входов, 16 цифровых выходов и 2 аналоговых выхода. На этапе программирования LOGO! необходимо назначить соответствие номеру блока в программе пользователя порядковому номеру датчика (входа) или нагрузки (выхода), подключенных к контроллеру. Адресация входов и выходов осуществляется последовательно, согласно расположению входов и выходов по отношению к базовому модулю, слева направо. Максимальная конфигурация с использованием всего адресного пространства может быть достигнута различными способами. На рис. 1.3 (а-в) показаны примеры максимальной конфигурации входов и выходов.

I1, I2, I3 .. I6, I7, I8 AI3, AI4, AI1, AI2  LOGO! Basic  Q1..Q4	I9..I12  LOGO! DM 8  Q5..Q8	I13..I16  LOGO! DM 8  Q9..Q12	I17..I20  LOGO! DM 8  Q13..Q16	I21..I24  LOGO! DM 8	AI5, AI6  LOGO! AM 2	AI7, AI8  LOGO! AM 2		AQ1, AQ2  LOGO! AM2 AQ
---	---	---	--	----------------------------	----------------------------	----------------------------	--	------------------------------

а

I1, I2, I3 .. I6, I7, I8 AI1, AI2  LOGO! Basic  Q1..Q4	I9..I12  LOGO! DM 8  Q5..Q8	I13..I16  LOGO! DM 8  Q9..Q12	I17..I20  LOGO! DM 8  Q13..Q16	I21..I24  LOGO! DM 8	AI3, AI4  LOGO! AM 2	AI5, AI6  LOGO! AM 2	AI7, AI8  LOGO! AM 2	LOGO! AM2 AQ  AQ1, AQ2
---	---	---	--	----------------------------	----------------------------	----------------------------	----------------------------	------------------------------

б

I1 .. . . . . I8  LOGO! Basic  Q1..Q4	I9..I12  LOGO! DM 8  Q5..Q8	I13..I16  LOGO! DM 8  Q9..Q12	I17..I20  LOGO! DM 8  Q13..Q16	I21..I24  LOGO! DM 8	AI1, AI2  LOGO! AM 2	AI3, AI4  LOGO! AM 2	AI5, AI6  LOGO! AM 2	AI7, AI8  LOGO! AM 2	LOGO! AM2 AQ  AQ1, AQ2
---	---	---	--	----------------------------	----------------------------	----------------------------	----------------------------	----------------------------	------------------------------

в

Рис. 1.3. Максимальная конфигурация входов и выходов: а – 4 цифровых и 3 аналоговых модуля, б – 4 цифровых и 4 аналоговых модуля, в – 4 цифровых и 5 аналоговых модулей

Выходы I1-I8 находятся на базовом модуле, остальные входы нумеруются по порядку подключения. Нумерация выходов производится аналогичным образом. Очередность подключения модулей не нарушает способ адресации – так, например, аналоговый модуль может быть включен между двумя цифровыми.

Для модулей LOGO! 12/24 RC/RCo и LOGO! 24/240 можно настроить использование модулем двух или четырех из четырех доступных аналоговых входов. Аналоговые входы (AI) нумеруются последовательно в зависимости от числа настроенных входов, используемых в базовом модуле. Если настроено использование двух входов, они нумеруются AI1 и AI2, при этом эти входы соответствуют входным клеммам I7 и I8. Для последующих модулей расширения с аналоговыми входами (AI) нумерация будет начинаться с AI3. Если настроено использование четырех входов, они нумеруются AI1, AI2, AI3 и AI4, при этом эти входы соответствуют входным клеммам I7, I8, I1 и I2 в указанном порядке. Для последующих модулей расширения с аналоговыми входами AI нумерация будет начинаться с AI5.

Кроме аналоговых и цифровых модулей, в линейку средств расширения входит модуль LOGO! CM, предназначенный для подключения LOGO! к сети AS-i в качестве Slave-устройства. Данный модуль предоставляет возможность организовать 4 виртуальных входа и 4 виртуальных выхода, включаемые в адресное пространство Master-устройства. Передавая данные на виртуальные входы и получая данные от виртуальных выходов, Master-устройство (например, ПЛК S7-200 или S7-300) может осуществлять обмен с LOGO!.

Более подробную информацию о способах подключения, характеристиках входов и выходов, адресации модулей, можно получить в [1].

## **2. ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ LOGO!**

### **2.1. Способы и средства программирования LOGO!**

Создание или изменение программы в LOGO! осуществляется тремя способами:

- При помощи кнопок на блоке управления (действия отображаются на дисплее, для версий LOGO! с наличием дисплея).
- Создание программы на ПК с помощью среды отладки LOGO! Soft Comfort и загрузки ее в контроллер.
- Копирование ранее созданной программы в LOGO! при помощи внешней карты памяти.

Программа, загруженная в LOGO! тем или иным способом, сохраняется в долговременной FLASH-памяти контроллера и хранится неограниченно долго после выключения питания. После сброса питания программа автоматически загружается в ОЗУ контроллера и выполняется. Такой вариант работы применим в тех случаях, когда программа изменяется редко, либо вообще не изменяется пользователем.

Программирование с помощью встроенной клавиатуры и дисплея применяется в тех случаях, когда объем программы невелик, либо необходима незначительная коррекция уже существующей программы (например, изменение величины задержки в программе управления задвижкой).

Программирование с помощью LOGO! Soft Comfort - основной способ изменения или создания программы. Данная среда позволяет создавать программы на языках LAD и FBD, производить их отладку и эмуляцию.

Иногда, для оперативной перестройки оборудования на другой алгоритм работы, необходима быстрая смена программы без участия ПК. Для этих целей для LOGO! разработаны специальные модули памяти,

маркированные желтым и красным цветом. Возможные варианты работы с модулями памяти следующие:

- если в модуле находится программа для LOGO!, то при установке модуля в разъем интерфейса происходит автоматическое копирование программы из модуля памяти во внутреннюю FLASH-память контроллера и при сбросе питания LOGO! выполняет новую программу;
- при необходимости можно сохранить существующую программу из FLASH-памяти в съемный модуль, посредством меню LOGO!;
- при необходимости можно сохранить программу из РС в модуль памяти посредством меню LOGO!.

Различие красного и желтого модулей памяти заключается в ограничении прав доступа на операции копирования программы.

## 2.2. Действия LOGO! при сбросе и восстановлении питания

Так как у LOGO! нет выключателя питания, то многие действия выполняются автоматически. Если на модуль подано питание, то контроллер может находиться в двух режимах (рис. 2.1)

STOP	RUN
<ul style="list-style-type: none"><li>• На дисплее отображается: «Нет прогр.» (кроме модулей LOGO!...o)</li><li>• Переключение модуля LOGO! в режим программирования (кроме модулей LOGO!...o)</li><li>• Светодиод светится красным цветом (только модули LOGO!...o)</li></ul>	<ul style="list-style-type: none"><li>• Дисплей: экранная маска для контроля входов/выходов и сообщений (после выбора «ПУСК» в главном меню) (кроме модулей LOGO!...o)</li><li>• Переключение модуля LOGO! в режим ввода параметров (кроме модулей LOGO!...o)</li><li>• Светодиод светится зеленым цветом (только модули LOGO!...o)</li></ul>
<p>Действия модуля LOGO!</p> <ul style="list-style-type: none"><li>• Входные данные нечитываются.</li><li>• Коммутационная программа не выполняется.</li><li>• Релейные контакты постоянно разомкнуты; бесконтактные выходы отключены.</li></ul>	<p>Действия модуля LOGO!</p> <ul style="list-style-type: none"><li>• Модуль LOGO! считывает состояние входов.</li><li>• Модуль LOGO! использует коммутационную программу для вычисления состояний выходов.</li><li>• Модуль LOGO! включает и отключает релейные и бесконтактные выходы.</li></ul>

Рис. 2.1. Режимы работы LOGO!

Режим работы модулей расширения определяется по цвету свечения индикаторного светодиода (рис. 2.2)

Цвет свечения светодиода (RUN/STOP)		
Зеленый (RUN)	Красный (STOP)	Оранжевый / желтый
Модуль расширения обменивается данными с устройством, расположенным слева.	Модуль расширения не обменивается данными с устройством, расположенным слева.	Фаза инициализации модуля расширения

*a*

Цвет свечения светодиода модуля AS-i		
Зеленый	Красный	Красный / желтый
Обмен данными по интерфейсу AS выполняется.	Сбой обмена данными по интерфейсу AS.	Ведомое устройство имеет нулевой адрес.

*b*

Рис. 2.2. Режимы работы модулей расширения, *a* – для всех модулей, *b* – дополнительный индикатор модуля AS-i

При сбросе или включении питания LOGO! выполняет следующие действия:

- если в LOGO! или в установленном программном модуле нет программы, то LOGO! (с дисплеем) отображает сообщение: «No Program. Press ESC» (Нет программы. Нажмите ESC);
- если в программном модуле есть программа, она автоматически копируется в LOGO!. Коммутационная программа, находящаяся в LOGO!, заменяется.
- если в LOGO! или в программном модуле есть коммутационная программа, то LOGO! принимает рабочий режим, который у него был до выключения питания. Если используется вариант без дисплея (LOGO! ... o), он автоматически переходит из STOP в RUN (светодиод переключается с красного на зеленый).

Более подробно процедуры загрузки контроллера описаны в [1].

## 2.3. Обзор меню LOGO!

Меню LOGO! условно можно разделить на два режима – режим программирования и режим параметризации. Режим программирования – начальный режим, используемый при вводе программы, либо при организации связи между LOGO! и ПК. Режим параметризации действует при работе программы и предназначен для контроля ее выполнения, а также для отладки в реальном времени. На рис. 2.3 показан вид основных окон меню, а также способы навигации.

В режиме «*Program..*» можно осуществить следующие операции:

- *EDIT PRG* – режим редактирования и ввода программы;
- *PRG NAME* – задание имени программы;
- *CLEAR PRG* – очистить ПЗУ контроллера;
- *PASSWORD* – задать пароль для ограничения доступа к программе.

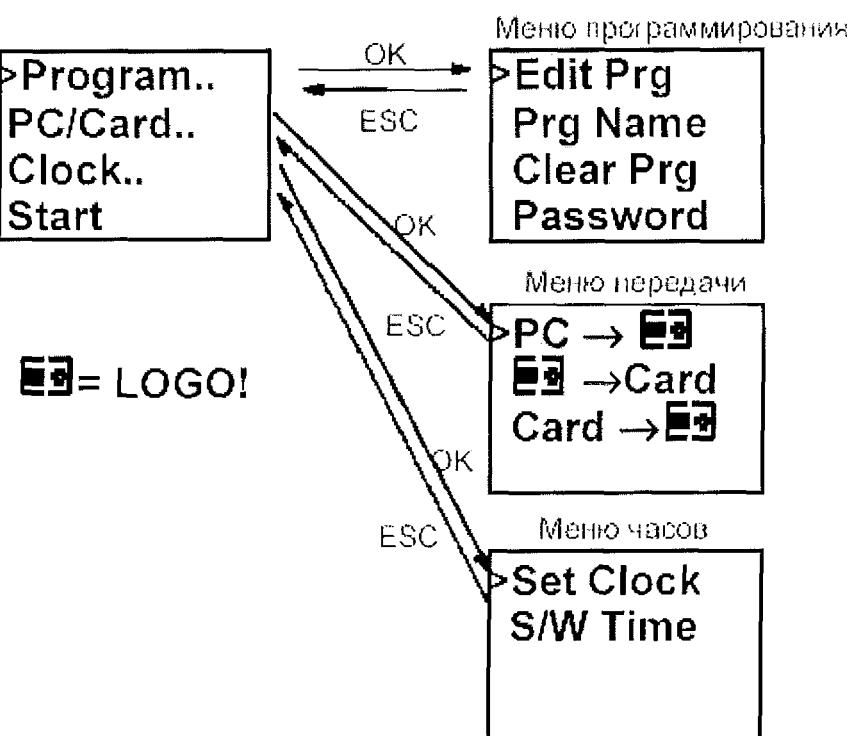


Рис. 2.3. Основные функции меню в режиме программирования

В режиме *PC/CARD* пользователь управляет загрузкой программы со стороны ПК, либо осуществляет операции по копированию программы в сменный модуль памяти:

1. **PC → Card** - режим обмена между LOGO! и ПК, посредством интерфейса связи под управлением LogoSoftComfort;
2. **Card → Card** - режим копирования программы из памяти LOGO! в сменный модуль памяти;
3. **Card → PC** - загрузка программы из сменного модуля в ПЗУ LOGO!.

В режиме *CLOCK* производится установка часов реального времени для правильной работы функций, связанных с обработкой календарных событий. Встроенные часы LOGO! получают питание от источника контроллера, но буферизированы конденсатором большой емкости, что защищает их от кратковременных сбоев питания. При отключении питания часы сохраняют работоспособность около суток.

Функция *START* переводит LOGO! в режим выполнения программы, если она присутствует в памяти, в противном случае настроенном дисплее отображается сообщение «No program».

Режим параметризации показан на рис. 2.4. В этот режим пользователь может войти по нажатию «ESC» на клавиатуре LOGO! в режиме выполнения программы.

В режиме параметризации пользователю доступны следующие функции:

- *STOP* – останов программы пользователя;
- *SET PARAM* – установка параметров блоков.

Например, можно установить новое значение в блоке задержки включения выхода, изменить порог срабатывания аналогового компаратора, и т.д. — то есть у тех блоков, которые предусматривают начальную настройку. Параметры блоков,

Меню параметризации

>Stop  
Set Param  
Set Clock  
Prg Name

Рис. 2.4. Режим

параметризации LOGO!

Параметры блоков, устанавливаемые с помощью меню, будут описаны ниже в соответствующих разделах;

- *SET CLOCK* – установка, либо коррекция часов реального времени (аналогичный режим существует в режиме программирования);
- *PRG NAME* – ввод имени программы.

## 2.4. Переход от принципиальной схемы к блок-диаграмме

Программа для LOGO! отличается от большинства программ других контроллеров тем, что представлена в графическом виде. В этом смысле представление программы сходно с представлениями о моделях систем, реализованных в некоторых программных продуктах, например: в системе MATLAB в пакете Simulink, в системах схемотехнического моделирования MicroCap, Electronic Workbench, LabView, и др.

Так как LOGO! ориентирован на замену релейно-контакторных схем, то программу (или блок-диаграмму) проще всего представить в виде стандартных логических операций, дополненных специализированными типовыми функциями, наиболее часто встречающихся в таких системах. Например, специализированные функции могут выполнять задачи импульсных реле, реле времени с задержкой включения или выключения и т.д. На рис. 2.5 показан один из вариантов контактной схемы, который необходимо перевести в блок-диаграмму LOGO!.

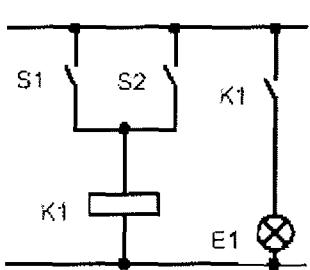


Рис. 2.5. Пример релейно-

контакторной схемы

необходимо выполнить следующие шаги:

Реле K1 срабатывает и замыкает своими контактами цепь лампы E1 в том случае, когда замкнут любой из выключателей S1 и S2. Аналитически это можно записать в виде  $E1 = S1 \vee S2$ , так как параллельное соединение выключателей представляет собой элемент “ИЛИ”.

Для реализации этой функции в LOGO!

- выполнить электрический монтаж;
- определить адреса входов и выходов;
- ввести программу и запустить ее на выполнение.

Электрический монтаж для данной схемы выполняется согласно схеме подключения датчиков и нагрузки к LOGO! (рис.1.2). Датчики S1 и S2 в примере подключены к входам I1 и I2 LOGO!, лампа подключена к источнику питания через контакты реле Q1.

Блок-диаграмма, вводимая в контроллер, может выглядеть так, как показано на рис 2.6а. При этом, блоки, помеченные как «I» соответствуют контактам S1 и S2, блок «B01» – элементу “ИЛИ”, а «Q» – контактам реле. Еще один альтернативный вид отображения программы показан на рис. 2.6б. Блок-диаграмма отображается непосредственно в контактном виде (LAD-представление, используемое как один из языков других контроллеров SIEMENS).

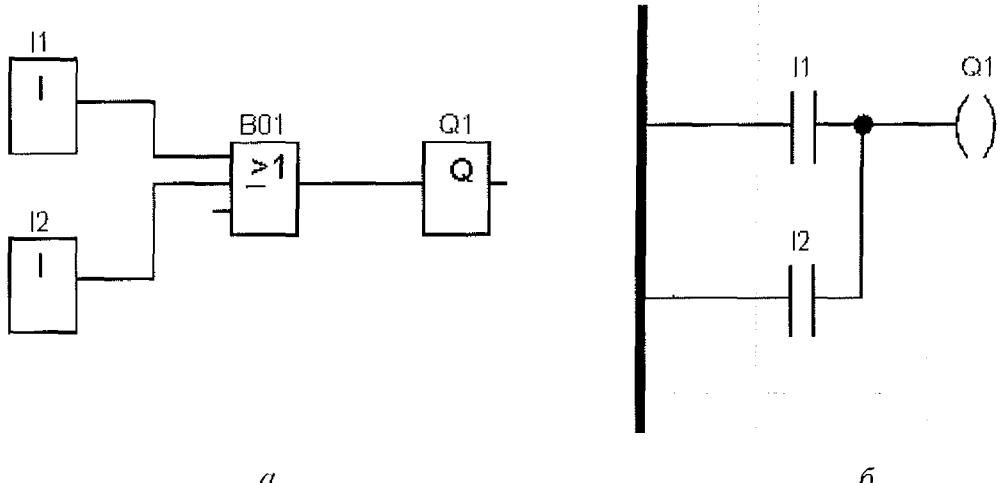


Рис 2.6. Представление электрической схемы в виде: а- блок диаграммы FBD и б - LAD-диаграммы

В каждом конкретном случае пользователь выбирает наиболее удобный способ представления. Вид представления FBD (Functional Block Diagramm) используется в тех случаях, когда в схеме имеется большое число

обратных связей, тогда как представление LAD (Ladder Diagramm) применяется в простых алгоритмах на основе релейной логики.

## 2.5. Порядок ввода блок-диаграммы с помощью встроенного дисплея

На примере схемы, показанной на рис. 2.5, рассмотрим общий порядок ввода схем в LOGO! с помощью встроенной клавиатуры и дисплея.

В меню режима программирования необходимо выбрать **Edit Prg** и нажать **OK**. На дисплее появится первый выход схемы (рис. 2.7). Символ «Q» в Q1 подчеркнут, таким образом курсор показывает текущую позицию в программе. Курсор можно перемещать нажатием клавиш «вверх», «вниз», «влево», «вправо». Выбрав «влево», переместим курсор влево (рис. 2.8).

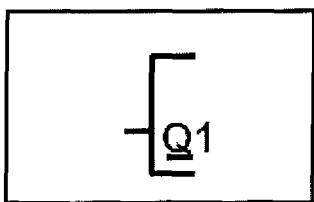


Рис.2.7. Первый выход схемы

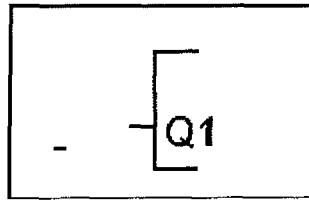
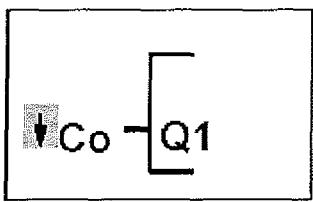


Рис.2.8. Перемещение курсора влево

В этой точке можно ввести первый соответствующий нашей схеме блок.



Для перехода в режим ввода необходимо нажать **OK**, (рис. 2.9). Курсор отображается в виде сплошного прямоугольника, приглашая выбрать соединительный элемент В это время LOGO!

Рис.2.9. Точка выбора соединительного элемента

предлагает различные возможности выбора, путем нажатия клавиши «вниз» можно выбрать нужный блок, например, блок BF (Basic functions - основные функции) и нажать **OK**. Нажимая клавишу «вниз» можно просмотреть

список элементов блока (BF) и выбрать нужный элемент, нажав OK. На рис. 2.10 показан пример с элементом ИЛИ. В верхнем правом углу обозначен номер блока.

Теперь можно соединить входы блока со следующими блоками, соответствующими схеме, нажав клавишу OK. На рис. 2.11 показан дисплей LOGO! с введенным элементом. Выбираем список Со и нажимаем OK, результат показан на рис. 2.12. Первым элементом в списке Со

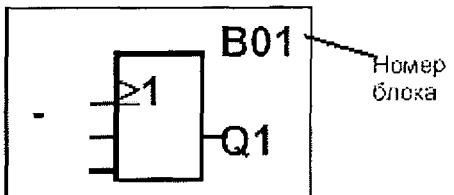


Рис.2.10. Элемент «ИЛИ»

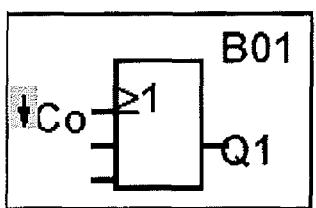


Рис. 2.11. Дисплей LOGO! с элементом «ИЛИ»

является символ, показывающий, что вход не используется, т.е. «X». Нажимая клавиши «вверх», «вниз» выбираем номер входа LOGO! (например I1) (рис. 2.13). Если к элементу ИЛИ нужно подключить еще один элемент, то вместо списка Со выбираем нужный список, а затем элемент аналогично выше

описанным действиям.

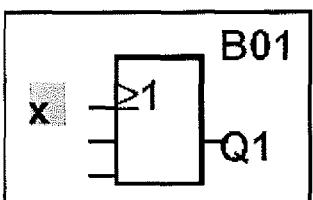


Рис.2.12. Выбор элемента

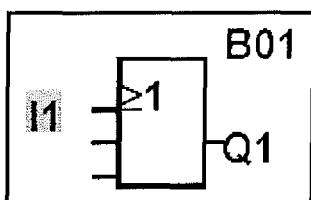


Рис.2.13. Выбор номера

После нажатия на OK, вход LOGO! I1 соединен с элементом ИЛИ (рис. 2.14). Соедините вход I2 с элементом ИЛИ. Если 3-й вывод элемента ИЛИ не нужен, выбираем элемент «X». Если схема

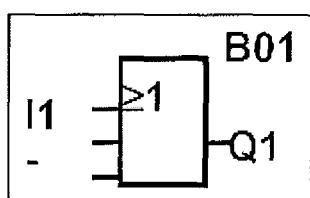


Рис.2.14. Дисплей LOGO! с элементом «ИЛИ»5

введена полностью, нажимаем OK, после чего она примет следующий вид (рис. 2.15).

Для просмотра программы можно выполнять действия клавиш «влево», «вправо». Далее, необходимо перейти в режим программирования. Если

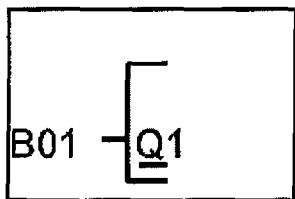


Рис.2.15.

Законченная программа  
программа  
введенной коммутационной схемы. Для контроля выполнения программы можно воспользоваться просмотром состояния входов и выходов LOGO!. Вид дисплея LOGO! в режиме контроля показан на рис. 2.16.

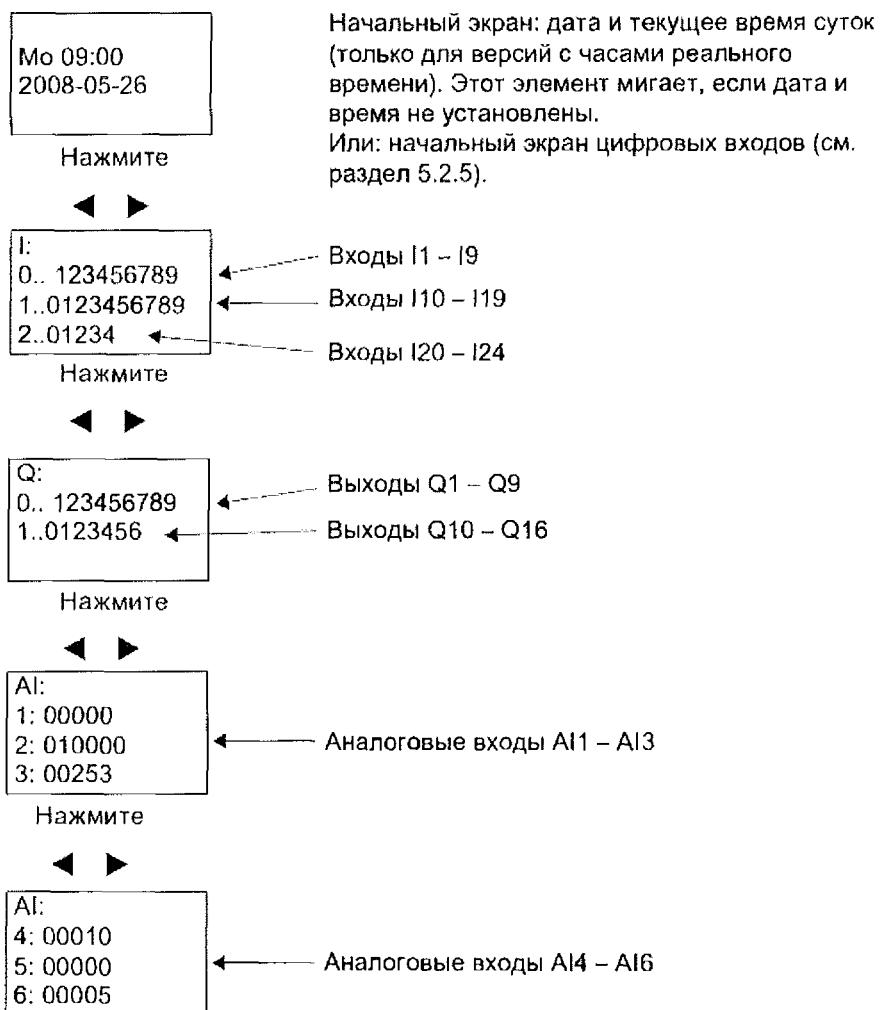


Рис. 2.16. Контроль состояния входов и выходов

При изменении состояния входов и выходов их значения индицируются в виде «1» и «0», значения аналоговых входов – в виде целых значений в диапазоне 0...1000.

## 2.6. Программирование с помощью LOGO! Soft Comfort

Программирование LOGO! посредством LOGO! Soft Comfort требует соединения LOGO! с ПК специальным интерфейсным кабелем. Соединение со стороны LOGO! производится через разъем сменного модуля памяти, а со стороны ПК – через один из свободных COM – портов.

Основные функции LOGO! Soft Comfort:

- создание программ в виде блок-диаграмм, или в LAD-представлении;
- редактирование программы;
- отладка программы в режиме эмуляции;
- установка системных часов LOGO!, очистка памяти;
- загрузка программы в LOGO! и чтение ранее введенной программы.

На рис. 2.17 показан вид основного окна LOGO! Soft Comfort.

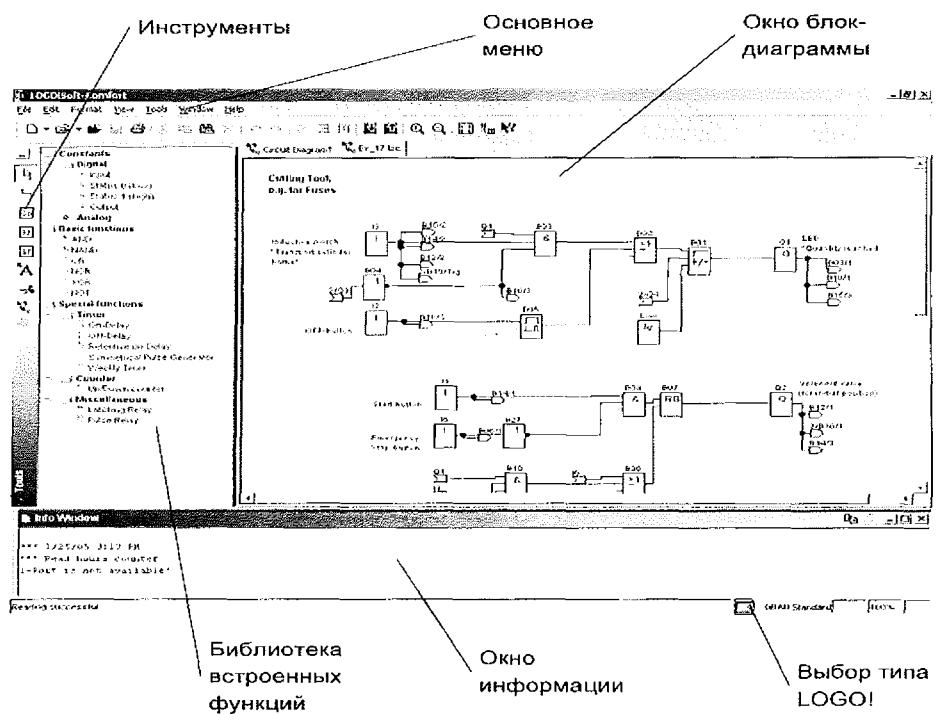


Рис. 2.17. Вид окна LOGO!SoftComfort

Программа вводится в виде блок-диаграммы (или в виде LAD – выбирается в основном меню). Элементы, из которых состоит программа, выбираются из списка функций справа от программного окна. Список функций зависит от типа используемого LOGO!, который должен быть задан в нижнем правом углу окна, либо через основное меню. Для ввода программы используются соединительные элементы, а также стандартные операции копирования и удаления, выбор которых производится в окне «инструменты». В окне «информация» отображаются основные действия пользователя при работе с LOGO!.

Краткое описание основного меню:

1. *FILE* – операции открытия и сохранения программы на диске, защита проекта паролем, выбор вида представления программы, управление печатью;
2. *EDIT* – операции копирования и вставки, а также назначения символьических имен переменным, соответствующих входам и выходам;
3. *FORMAT* – выбор шрифта, отображение сетки, выравнивание;
4. *VIEW* – настройка вида окон, масштабирование;
5. *TOOLS* – инструменты:
  - a) *TRANSFER* – управление загрузкой программ и данных:
    - *PC ->LOGO!* – загрузка программы в LOGO!;
    - *LOGO! ->PC* – чтение программы из LOGO!;
    - *SET CLOCK* – установка часов реального времени в LOGO!;
    - *SUMMER TIME* – установка разрешения перехода с зимнего на летнее время и наоборот;
    - *HOUR COUNTER* – чтение содержимого счетчика часов наработки LOGO!
  - б) *SELECT HARDWARE* – определение типа LOGO! вручную;

- в) *DETERMINE LOGO!* – автоматическое определение типа подключенного LOGO!;
- г) *SIMULATION* – запуск симулятора;
- д) *OPTIONS* – настройка цветов экрана, адреса COM-порта.

Для создания проекта с помощью программной среды необходимо придерживаться последовательности действий:

1. Запустить LOGO! Soft Comfort, создать новый проект, выбрать вид представления (блок-диаграмма или LAD).
2. Выбрать тип подключенного LOGO! через меню *SELECT HARDWARE* (в примере – 0BA3).
3. Ввести проект, используя стандартные функции, доступные для этого вида LOGO!.
4. Сохранить проект на диске с именем.
5. Установить соединение с LOGO!, выбрав правильный адрес COM-порта через меню *OPTIONS*.
6. Установить текущую дату и время, если это необходимо (в программе есть функции, оперирующие с датой и временем).
7. Запустить программу в режиме эмуляции. Вид панели управления в режиме эмуляции показан на рис. 2.18.

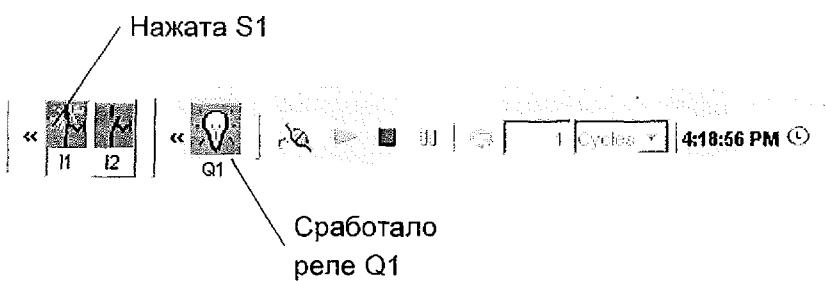


Рис.2.18. Режим эмуляции программы

8. Установить на LOGO! режим PC<-> CARD и загрузить программу в контроллер, используя команду меню TRANSFER/PC ->LOGO!.
9. После окончания загрузки можно запустить программу в LOGO! командой START.

Встроенный эмулятор предоставляет пользователю ресурсы LOGO!, используемые в программе, – в данном примере два дискретных входа и реле.

## 2.7. Варианты заданий

1. На учебном стенде (прил. 1) ввести программу в LOGO! с помощью встроенного дисплея, согласно варианту задания (табл. 2.1). Убедиться в правильности работы, проконтролировать правильность работы программы, анализируя значения входов и выходов.
2. Настроить среду LOGO! Soft Comfort, произвести загрузку введенной программы. Выполнить программу в режиме эмуляции.
3. Добавить в существующую программу инверсию входов, загрузить ее в LOGO! и проверить правильность работы.

Таблица 2.1

Варианты заданий

Вариант	Схема
1	$Y=((X1 + \overline{X1}) * (X3 + \overline{X4} + X1 * \overline{X4}) + (X2 + (X2 * \overline{X3} * X4)) * X1 + X3 * X1)$
2	$Y=((X3 + X2) * (X1 + X4) + X1) + X2) * (X4 + \overline{X4} + X1) + X3$
3	$Y=(X2 * X1 + X3 * \overline{X1} + X2 * X3 * \overline{X4}) * (X1 + X2) + \overline{X4} * X4 * X1$
4	$Y=(\overline{X4 + X3 + X1 + X4 + X2}) * (X4 + \overline{X1}) * (X2 + \overline{X2}) + X4 * X1$
5	$Y=(X1 * \overline{X2} * X4 + X3 * (X4 + X1 + \overline{X3})) + \overline{X1} * (X4 + \overline{X3} + X2)$

Окончание таблицы 2.1

Вариант	Схема
6	$Y = \overline{(X_4 * X_1 + X_3 * X_1 + X_2 * X_3) * (X_4 + X_3) * (X_1 + X_2) + X_3 + X_2 * X_4}$
7	$Y = (X_1 * X_4) + ((\overline{X_4} + X_1 + \overline{X_2 * (X_3 * X_1)}) * X_4) * X_1 + X_2 + \overline{X_3}$
8	$Y = ((X_4 * \overline{X_1} * X_3) + (X_1 * X_3)) * X_1 + (\overline{X_2 + X_1 + X_4} + (\overline{X_3 + X_4}) + X_1)$
9	$Y = (X_4 + X_1) * X_2 + X_3 * X_2 + (X_1 * X_3 + X_2 * X_3) * X_1 * X_2 + X_4 + X_2$
10	$Y = (X_1 + X_2 + X_3) * (X_3 + X_1 * X_2) + (X_4 * X_1 * X_3 + X_4) * X_2 + X_3$

### 3. ВВОД-ВЫВОД И БАЗОВЫЕ ФУНКЦИИ LOGO!

#### 3.1. Базовые функции и их особенности

Контроллер LOGO! оперирует исключительно с битовыми данными, так как основная сфера его применения – устройства двоичной логики. Поэтому в списке основных функций ввода-вывода отсутствуют операции чтения байт, слов, и т. д. Каждому входу и выходу LOGO! поставлен в соответствие уникальный адрес бита входа (I), и адрес бита выхода (Q).

Состояние входов и выходов являются булевыми переменными для стандартных базовых функций. В распоряжение пользователя предоставляются дополнительные возможности:

- *M* (меркер) – бит памяти, который можно установить или сбросить, являющийся внутренней переменной, не имеющей выхода.
- *Hi* – высокий уровень. Значение этой переменной всегда равно «1».
- *Lo* – низкий уровень. Значение этой переменной всегда равно «0».

К основным функциям LOGO! относятся:

- *AND* – логическое «И»;
- *OR* – логическое «ИЛИ»;
- *NAND* – «И-НЕ»;
- *NOR* – «ИЛИ-НЕ»;
- *XOR* – «исключающее ИЛИ»;
- *NOT* – инверсия.

Основные логические функции не нуждаются в описании, однако для ряда задач введены дополнительные их модификации – логическое «И по фронту», и «НЕ-И по фронту». Временные диаграммы, иллюстрирующие их работу, показаны на рис. 3.1 и 3.2.

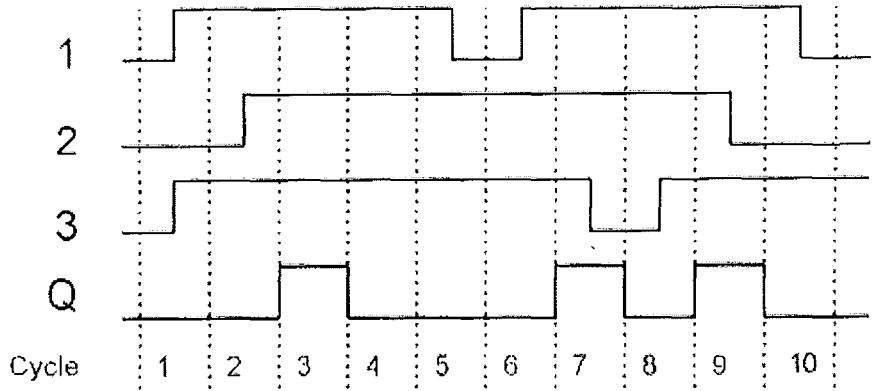


Рис. 3.1. Временная диаграмма работы элемента «И по фронту»

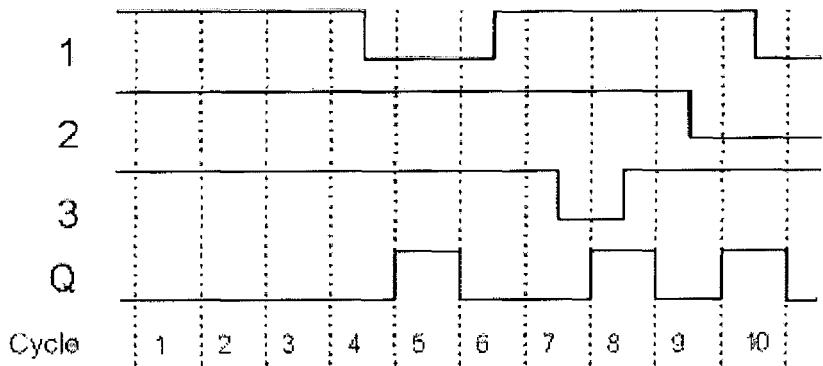


Рис. 3.2. Временная диаграмма работы элемента «НЕ-И по фронту»

Основное отличие этих операций от базовых состоит в том, что результат логической операции (РЛО) присутствует на выходе элемента ограниченное время, равное одному циклу моделирования программы. Необходимость в такой операции важна для фиксации перехода из одного состояния входа в другое, последующее же его состояние не влияет на РЛО. Это часто используется в тех случаях, когда важен момент перехода состояния I в 1 или 0, тогда как последующее состояние не должно влиять на подключенные блоки. Например, если подключить вход II=1 к счетчику без выделения фронта, то в каждом цикле программы будет выполняться приращение значения счетчика. При использовании блока с выделением

фрона такого приращение будет вызвано единственный раз, при переходе I1 из 0 в 1.

Отличительными особенностями всех логических операций является то, что любой элемент имеет не более трех входов (кроме элемента НЕ). Для реализации многовходовых элементов необходимо применять их комбинации, согласно законам алгебры логики. Особый статус имеют также и неиспользуемые входы. Так, например, если на два входа элемента «И» подаются логические уровни с двух входов, а третий вход не подключен, то элемент считает его значение единичным (Hi). Если же вход подключен, то его значение равно РЛО предыдущего блока (например, третьего входа).

Цифровые константы Hi и Lo могут быть использованы в тех случаях, когда необходимо указать явно значение РЛО, например, при включении питания, когда одно из реле должно быть постоянно включено. Также их можно использовать для отладки программы.

*Меркер* – управляемый бит памяти, часто называемый флагом. Меркер не обладает триггерным эффектом. Это означает, что при исчезновении логической «1» на входе меркера, значение его РЛО тоже будет равно «0». Назначение меркерного бита состоит в задержке формирования РЛО на один такт моделирования. Задержка такой величины незаметна, однако позволяет реализовать в программе обратные связи, создание которых затруднено в реальных схемах. На рис. 3.3 показан пример применения меркера в схеме, выполняющей функцию асинхронного RS-триггера.

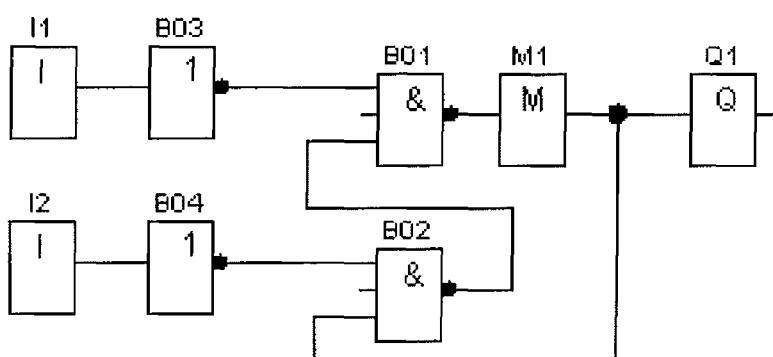


Рис. 3.3. Пример использования меркера

Одна из обратных связей подключена через меркер, что предотвращает эффект «гонки сигналов». В реальной схеме такой триггер устанавливается в произвольное состояние, тогда как в программе LOGO! выход устанавливается в нулевое состояние. Таким образом, использование задержки на один такт повышает надежность и прогнозируемость работы программы.

### 3.2. Вывод текстовых сообщений

Встроенный дисплей LOGO! позволяет выводить текстовые сообщения в процессе исполнения программы. Сообщения могут содержать текст в латинской кодировке, а также внедренные данные. Внедренные данные представляют собой параметры блоков специальных функций, например, при использовании функции «Задержка включения» можно вывести текущее время задержки этого блока. Параметры блоков задаются на этапе ввода программы либо в среде LOGO! Soft Comfort и будут рассмотрены в разделе «специальные функции».

Количество сообщений зависит от модели LOGO!. Так, например, LOGO! 0BA3 поддерживает до 5 текстовых сообщений. Пример вызова простого сообщения показан на рис. 3.4.

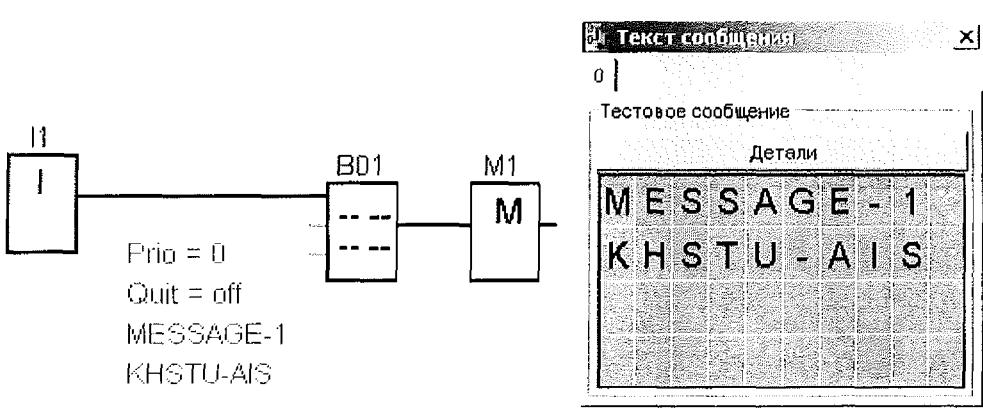


Рис. 3.4. Пример вызова текстового сообщения

В приведенном примере сообщение с текстом «MESSAGE-1 KHSTU-AIS» будет выведено на экран при срабатывании I1. *Наличие меркера или выхода обязательно, иначе вывод будет проигнорирован.*

Конфигурирование сообщения осуществляется через свойство блока «Message Text». Вид окна конфигурации показан на рис. 3.5.

В текстовом поле вводится текст, в дополнительных опциях можно выставить приоритет и требование о подтверждении. В разделе «Функциональный блок» можно выбрать блок, присутствующий в программе, и выбрать параметр, который необходимо внедрить в текстовое сообщение. Например, можно внедрить в сообщение значение аналогового ввода, время задержки включения реле и т.д.

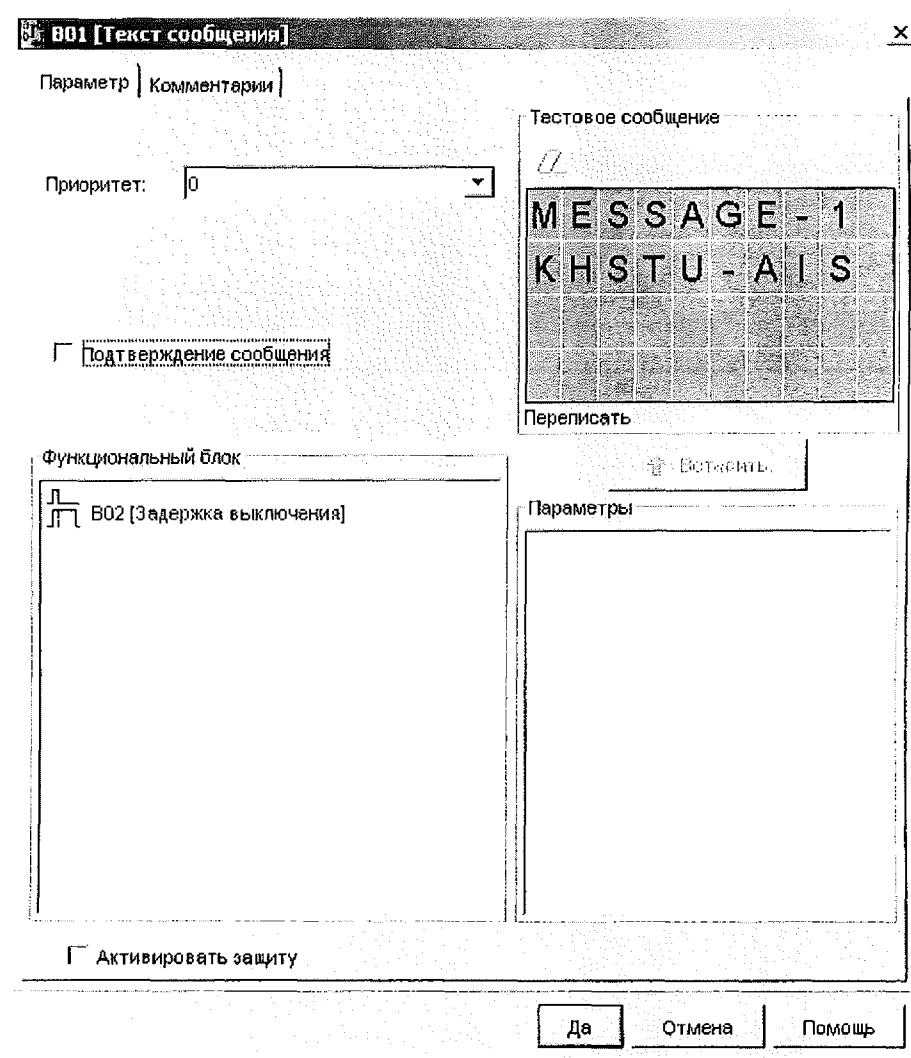


Рис. 3.5. Вид окна конфигурации сообщения

Приоритет сообщения необходим в тех случаях, когда появление сообщений от разных источников возникают одновременно. Так, например, сообщение об аварии должно иметь высший приоритет, а значит, должно появиться на дисплее в любом случае, даже если он занят другим сообщением. Если сообщение об аварии исчезает при исчезновении вызвавшего его условия, то на экране отображается сообщение с приоритетом, на единицу меньше (при условии, что активен вызов этого сообщения).

Если установить опцию «Подтверждение», то любое сообщение будет сохранено на дисплее до тех пор, пока не будет подтверждено оператором нажатием на клавишу «OK» в LOGO!. Сообщение остается на экране даже в том случае, когда исчезает условие его вызова. Однако если при этом возникает вызов сообщения с высшим приоритетом, то оно заменяется последним.

На рис. 3.6 показан пример вызова двух сообщений с разным приоритетом. Сообщение №2 будет отображено на дисплее в любом случае, сообщение №1 – только при отсутствии сообщения №2. Если возникнет сообщение №1, и будет вызвано сообщение №2, то после снятия последнего сообщение №1 будет требовать подтверждения, даже если I1 будет разомкнут.

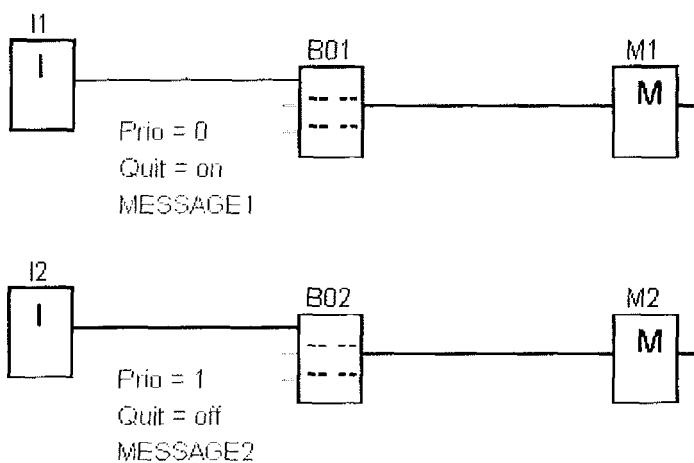


Рис. 3.6. Пример вызова разных сообщений

Для внедрения данных в текст сообщения необходимо выбрать блок из списка, затем выбрать нужный параметр и вставить его в текст. На рис. 3.7 показан пример конфигурации сообщения, содержащего текущее время блока задержки включения.

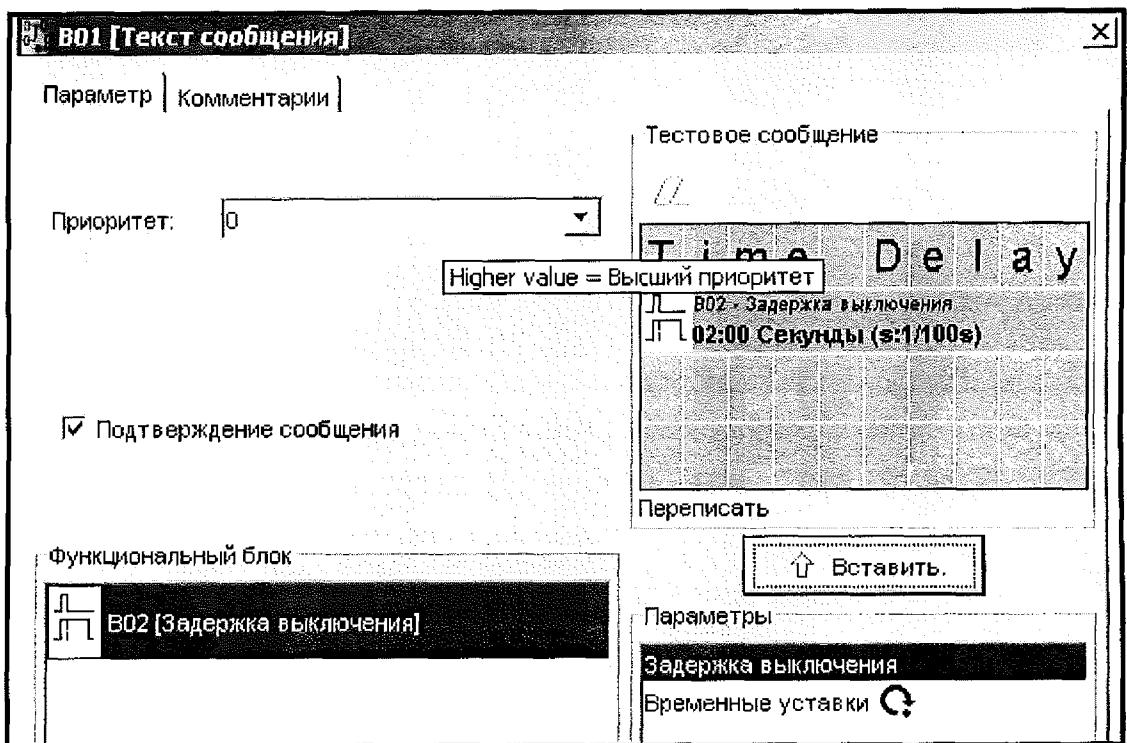


Рис. 3.7. Вывод параметра функционального блока

На рис. 3.8 показан вид дисплея LOGO! с выведенным сообщением. Оно содержит текст пользователя (Time Delay), а также текущее время, отрабатываемое блоком.



Рис. 3.8. Текст сообщения с внедренными данными

Применение базовых функций рассмотрим на примере простой задачи.

### 3.3. Пример использования базовых функций

**Задача.** Технический объект представляет собой три резервуара для хранения сыпучих продуктов. Каждый резервуар оснащен дискретным датчиком-сигнализатором предельного уровня. Датчики подключены к входам I1-I3 LOGO!. К входу I4 подключена кнопка без фиксации. К выходу Q0 подключена сигнальная лампа (HL), к выходу Q1 – звуковая сигнализация (BA).

Алгоритм работы следующий: если сработал хотя бы один сигнализатор, необходимо включить HL, и оставить ее включенной до тех пор, пока уровень хотя бы на одном из сигнализаторов превышает заданное значение. Звуковая сигнализация включается одновременно с лампой, но остается включенной, даже если уровень на всех резервуарах вернулся в норму. Выключение ВА производится кнопкой I4.

Вариант решения задачи приведен на рис. 3.10.

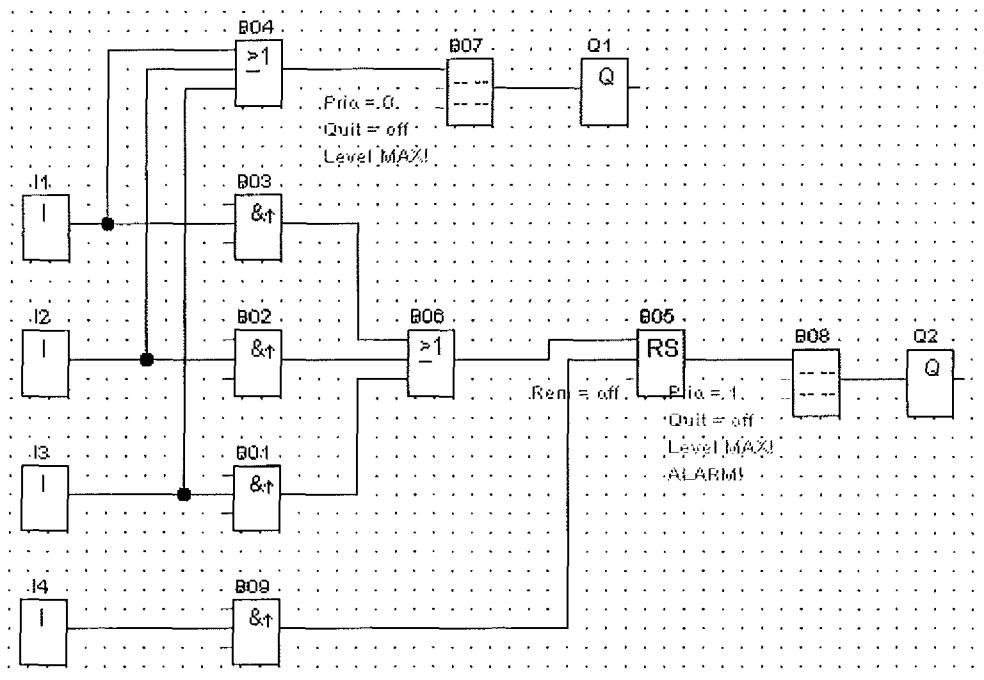


Рис. 3.10. Программа аварийной сигнализации

Управление сигнальной лампой осуществляется от элемента «ИЛИ», на входы которого подаются сигналы от датчиков. Если хотя бы один сработал, элемент «ИЛИ» генерирует на выходе логическую «1» и управляет выходом на лампу через промежуточное текстовое сообщение «Level MAX». Перепад от 0 к 1 от каждого выхода выделяется с помощью элемента «И по фронту» и после операции дизъюнкции поступает на вход S триггера, включая, таким образом, звуковой сигнал. Выключение сигнала производится сбросом триггера от кнопки I4. Для звукового сигнала создано второе сообщение с дополнительным текстом «ALARM». Если возникает сигнал тревоги, вызываются оба сообщения, но второе из них имеет высший приоритет, поэтому отображается «поверх» первого. Если сбросить звуковой сигнал, будет активно первое сообщение без строки «ALARM». На рис. 3.11 показан вид программы в режиме эмуляции.

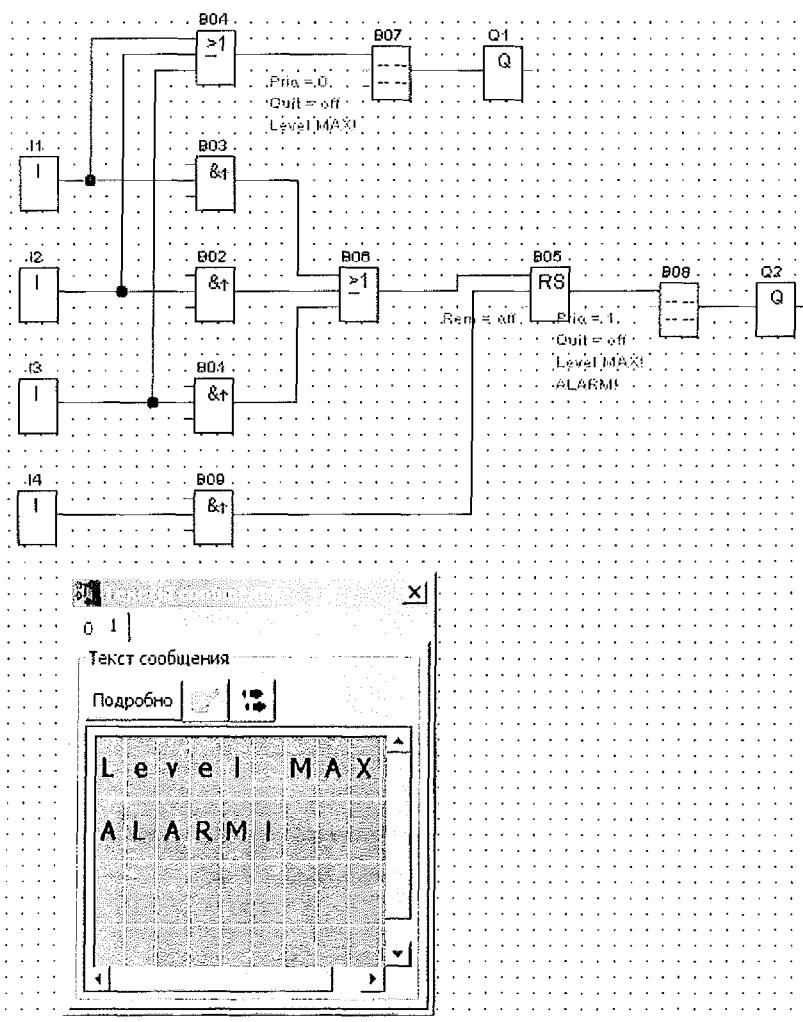


Рис. 3.11. Работа программы в режиме эмуляции

Более подробно особенности использования базовых функций описаны в [1].

### 3.4. Варианты заданий

- Согласно варианту задания выбрать логическую функцию и выводимые данные (табл. 3.1).
- Реализовать полученную функцию в среде LOGO!SoftComfort в любом удобном представлении и проверить ее работу в режиме эмуляции. Перевести программу в другой вид представления.
- Загрузить программу в LOGO! на учебном стенде и выполнить.

Схема подключения внешних цепей приведена в прил. 1.

Таблица 3.1

Варианты заданий

Вариант	S4	S5	S6	Q1	Q2	Q3	Q4	Сообщение	Приор.	Подтв.
1	0	0	0					Норма	0	
	0	0	1		1		1	Предел	2	да
	0	1	0	1		1				
	0	1	1					Нет нагрузки	1	
	1	0	0	1	1	1				
	1	0	1					Нет нагрузки	1	
	1	1	0		1	0	1	Предел	2	да
2	1	1	1	1		1		Тревога	3	да
	0	0	0					Обрыв датчиков	5	да
	0	0	1	1			1	Q1 и Q4 - ON	4	
	0	1	0		1		1	S5 - вкл	0	
	0	1	1	1	1		1			
	1	0	0		1	1		Q2 и Q3 - ON	3	
	1	0	1		1		1			
	1	1	0					Нет нагрузки	2	
	1	1	1	1		1		Все датчики - ON	1	да

Окончание таблицы 3.1.

Вариант	S4	S5	S6	Q1	Q2	Q3	Q4	Сообщение	Приор.	Подтв.
3	0	0	0	1				Q1 включено	0	да
	0	0	1			1		Q3 включено	1	
	0	1	0	1	1		1	Предупреждение	2	
	0	1	1			1				
	1	0	0	1	1		1	Предупреждение	2	да
	1	0	1							
	1	1	0	1	1	1	1	Пределъная нагруз.	3	да
	1	1	1		1					
4	0	0	0	1			1	Обрыв датчиков	3	да
	0	0	1		1					
	0	1	0			1		S5 включен	1	
	0	1	1	1	1		1			
	1	0	0	1	1		1	S4 включен	2	да
	1	0	1			1				
	1	1	0		1			Малая нагрузка	0	да
	1	1	1	1		1				
5	0	0	0		1		1	Q1 и Q4 включены	0	
	0	0	1					S6 включен	2	да
	0	1	0		1		1			
	0	1	1	1	1			Тревога	3	да
	1	0	0			1	1			
	1	0	1			1				
	1	1	0	1	1					
	1	1	1	1			1	Тревога	1	да

## **4. СПЕЦИАЛЬНЫЕ ФУНКЦИИ LOGO!**

### **4.1. Специальные функции и их применение**

Наряду с простейшими логическими операциями LOGO! предоставляет пользователю возможность использования специализированных функций, которые широко используются при решении практических задач в АСУ ТП. Среди многих задач в АСУ ТП возникает необходимость во временных задержках или привязка к реальному календарному времени. Наиболее часто используются задержки включения, выключения, функции формирования тактовых импульсов с заданными параметрами. Комбинируя функции таймеров с базовыми логическими функциями, можно получить достаточно большое число комбинаций, однако это сопряжено с рядом неудобств – увеличение размера программы, сложность проектирования, необходимость в одновременной работе нескольких таймеров и т. д. Поэтому LOGO! имеет в своей библиотеке набор уже готовых функций, который составлен с учетом опыта практических разработок. В этом случае нет необходимости использовать какие-либо таймеры, так как каждая библиотечная функция уже имеет его в своем составе. Таким образом, множество различных специальных функций может работать параллельно, независимо друг от друга.

Типовым примером специальной функции можно считать реле времени с задержкой включения (или выключения). Такая функция выполнена в виде готового блока, величина задержки при этом задается в виде параметра блока. Параметрирование функции можно осуществить с помощью LOGO! Soft Comfort, либо используя дисплей самого LOGO! в режиме редактирования или ввода программы.

На рис. 4.1 приведен пример программы, реализующей функцию светофора, управляемого кнопкой.

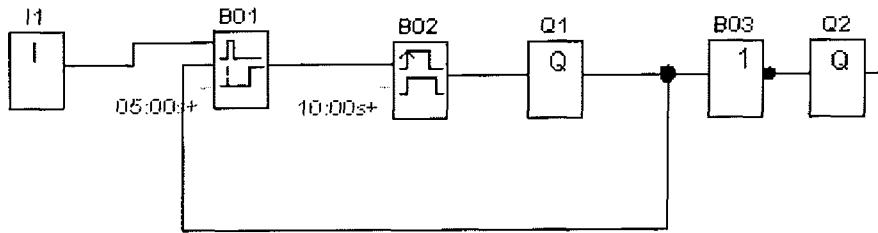


Рис. 4.1. Светофор, управляемый кнопкой

Алгоритм работы программы следующий: в исходном состоянии горит красный свет (реле Q2). При кратковременном нажатии на кнопку I1 выдерживается пауза, равная 5 с, после окончания которой включается зеленый свет (реле Q1). Зеленый свет горит 10 с, после чего вновь переключается на красный, и система переходит в начальное состояние.

В программе использованы две функции. Первая из них (B01) – реле задержки включения с запоминанием. Результат логической операции этой функции равен 1 в том случае, когда время, заданное в параметрах блока, истекло. Запуск счета времени начинается после фронта I1 и после окончания выдержки времени РЛО остается в единичном состоянии (запоминание). Второй блок представляет собой одновибратор, работающий по фронту и переключающий реле Q1 и Q2. Для возврата системы в исходное состояние предусмотрена обратная связь, сбрасывающая значение РЛО B01 в ноль, – для этого в функции предусмотрен вход сброса.

На рис. 4.2 показана времененная диаграмма, иллюстрирующая описанный процесс.

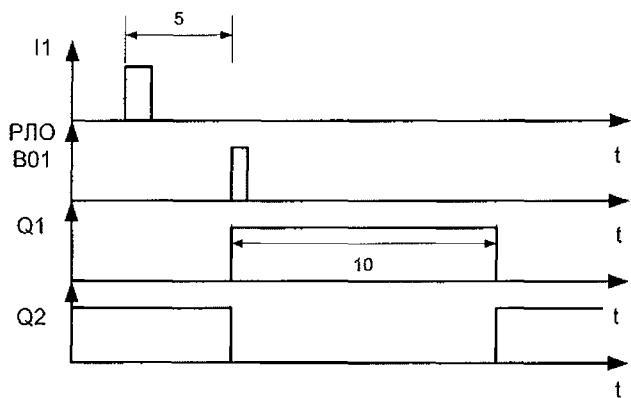


Рис. 4.2. Временная диаграмма работы светофора

Параметрирование функций производится через открытие контекстного меню «свойства блока». Вид окна показан на рис. 4.3.

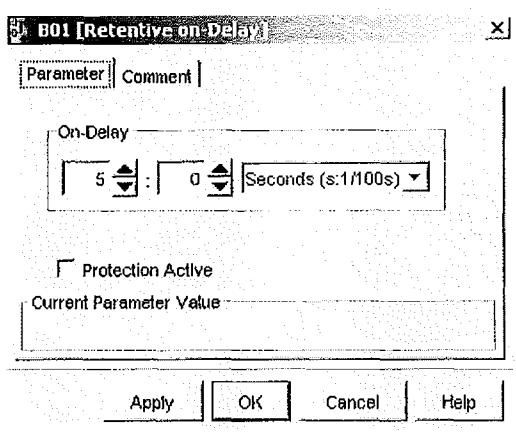


Рис. 4.3. Вид окна параметрирования блока

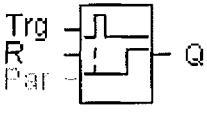
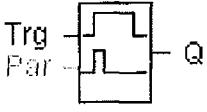
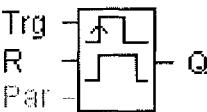
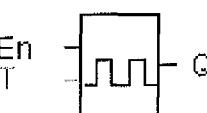
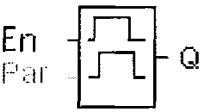
В данном случае задержку можно устанавливать в секундах (с точностью до 100 мс), в минутах и в часах, в зависимости от необходимой продолжительности. Если необходима суточная задержка, то необходимо использовать суточный таймер. Перечень специальных функций LOGO!, их графическое представление и описание приведены в табл. 4.1.

Таблица 4.1

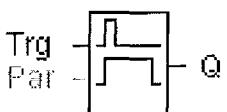
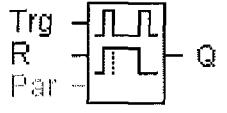
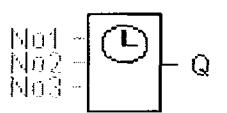
#### Специальные функции LOGO!

Функция	Представление	Описание
Задержка включения		Если возникает условие запуска, то Q включается через заданное время. Если условие включения исчезает, Q выключается.
Задержка выключения		При переходе TRG в "0" выход Q остается включенным на заданное время.
Задержка включения и выключения		Если на входе TRG присутствует "1", выход Q включается через заданное время задержки t1. При переходе TRG в "0" выход Q остается включенным на заданное время задержки t2

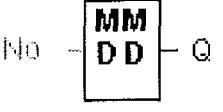
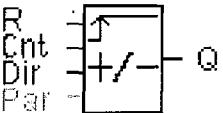
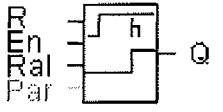
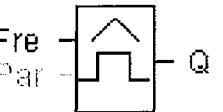
Продолжение таблицы 4.1

Задержка включения с запоминанием		При переходе TRG из “0” в “1” запускается время задержки, по истечении которого включается Q. Отличается от задержки включения тем, что для работы функции нет необходимости удерживать TRG в единичном состоянии
Формирователь импульса		При возникновении “1” на входе TRG на выходе Q формируется импульс заданной длительности. Если за время формирования снять условие на TRG, то формирование импульса досрочно завершится
Интервальное реле, запускаемое фронтом		Отличается от предыдущей функции тем, что для формирования импульса нет необходимости удерживать на TRG значение «1»
Генератор симметричных импульсов		При возникновении условия на EN на выходе Q формируется меандр. При параметрировании задается длительность импульса и паузы. Генерирование начинается с «1» на выходе
Генератор несимметричных импульсов		Отличается от предыдущей функции тем, что длительность паузы и импульса устанавливается независимо
Генератор случайных интервалов включения и выключения		Аналогичен задержке включения и выключения, однако время запаздывания при включении и выключении представляет собой случайную величину в диапазонах, указываемых при конфигурации

Продолжение таблицы 4.1

Выключатель света на лестничной клетке		Перепадом TRG с 1 на 0 запускается внутренний таймер Та. Сигналом TRG = 1 включается выход Q. За 15с до того, как текущее значение таймера достигнет установленного Т (если параметр времени Т установлен в минутах), выход Q будет кратковременно, на 1s, выключен ( $Q=0$ 1s). Если таймер достигнет установленного значения Т, - выход Q будет включен ( $Q=1$ ). При повторном переходе сигнала на входе TRG с 0 на 1 и снова на 0, в течение отсчета времени таймера, текущее значение сбрасывается (возможность перезапуска)
Двухфункциональный выключатель		Если на входе TRG присутствует импульс, то выход включается на заданное время, после чего выключается. Если TRG активен, то выход Q остается включенным и выключается при повторной смене состояния на входе TRG
Недельный таймер		Выход Q включается и выключается в определенное время и в определенные дни недели. Таймер может поддерживать несколько заданных схем включения и выключения в разные дни недели, после окончания недельного интервала цикл повторяется.

Продолжение таблицы 4.1

Годовой таймер		Выход Q включается и выключается в определенное время, выбираемое по календарю. Цикличности нет.
Реверсивный счетчик		В зависимости от направления счета (Dir) текущее значение счетчика инкрементируется или декрементируется при подаче фронта на вход Cnt. При равенстве текущего и заранее заданного значений выход Q включается и остается включенным. Выключение выхода производится подачей на R условия, либо обратным счетом до значения, меньшего предустановленному
Функция	Представление	Описание
Счетчик часов рабочего времени		При подаче на EN разрешающего условия запускается внутренний таймер. Если значение таймера равно предустановленному, включается выход Q. Вход R инициализирует цикл таймера заново, вход Ral сбрасывает значение выхода Q.
Частотный дискриминатор		Если за заданный интервал времени на входе зарегистрировано появление числа импульсов больше заданного N1, то выход Q включается. Если в следующем интервале времени на входе зарегистрировано число импульсов меньше заданного N2, выход выключается.

Окончание таблицы 4.1

RS - триггер		Выход Q включается по импульсу на входе S и выключается по импульсу на входе R.
Реле, управляемое одной кнопкой		Выход Q включается при появлении "1" на входе TRG и выключается при следующем появлении. R и S предназначены для асинхронной установки и сброса без участия TRG.

Специальные функции LOGO!, приведенные в таблице, относятся к варианту исполнения 0ВАЗ. Для других вариантов LOGO! их список может быть расширен или уменьшен. Так, например, в версии 0ВА4 добавлен сдвиговый регистр, а также изменены некоторые характеристики.

Большинство специальных функций оперируют с временными задержками, величина которых задается при параметрировании. При этом пользователю дается возможность задавать задержки в трех диапазонах:

- Секунды : S : MS;
- Минуты: M : S;
- Часы: H : M.

Пользователю предоставляется право выбора одного из трех диапазонов с переменной точностью. В первом случае дискретность установки времени составляет 100 мс, в диапазоне 0...59с, во втором – 1 с, в диапазоне 0...59 мин, в третьем – 1 мин, в диапазоне 0...99 ч. Если необходима длительная задержка и одновременно большая точность, то можно применять несколько специальных функций, работающих в разных временных диапазонах.

Каждая специальная функция может внедрять данные в текстовые сообщения. Информация, как правило, представлена в двух видах – для просмотра доступно как предустановленное значение, так и текущее.

На рис. 4.4 показан вид окна сообщения, содержащего предустановленное и текущее значения реверсивного счетчика.



Рис. 4.4. Вид окна сообщения, содержащего параметры счетчика

На рис. 4.5 показано окно конфигурации этого сообщения. Параметр «Counter Limit» содержит информацию о предустановленном значении, а параметр Counter – текущее значение счетчика.

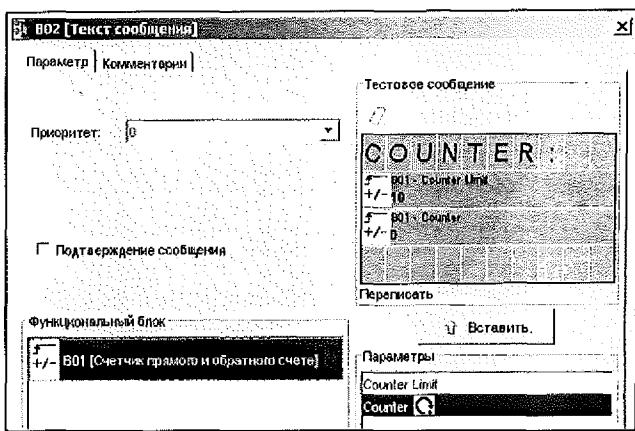


Рис. 4.5. Вид окна конфигурации сообщения

Более подробная информация о специальных функциях, временные диаграммы их работы и примеры использования содержатся в [1].

## 4.2. Пример использования специальных функций

Рассмотрим практический пример применения специальных функций.

Задача. С помощью LOGO! необходимо управлять запуском и остановом двигателя приточного вентилятора. Старт двигателя – по кнопке I1, останов – по I2. В воздуховоде установлен датчик потока, который

срабатывает в том случае, когда начинается устойчивое движение воздуха в нужном направлении. Алгоритм работы следующий: оператор запускает двигатель по кнопке I1. Если через 1 с (или менее), сработал датчик потока, система определяет, что запуск произошел в штатном режиме, о чем сигнализирует сообщение на экране. Останов двигателя осуществляется в штатном режиме по кнопке I2. Если в течение заданного времени датчик не сработал, система отключает двигатель и пытается запустить его автоматически, с периодом запуска 3 с. Если после трех попыток запуска определяется, что поток отсутствует, запуск блокируется и выдается сообщение о тревоге. Возврат к исходному состоянию осуществляется по кнопке I2.

Решение поставленной задачи может быть получено разными способами. Это обусловлено большим разнообразием специальных функций, которые могут быть использованы как в комбинации друг с другом, так и с базовыми. Один из вариантов решения показан на рис. 4.6.

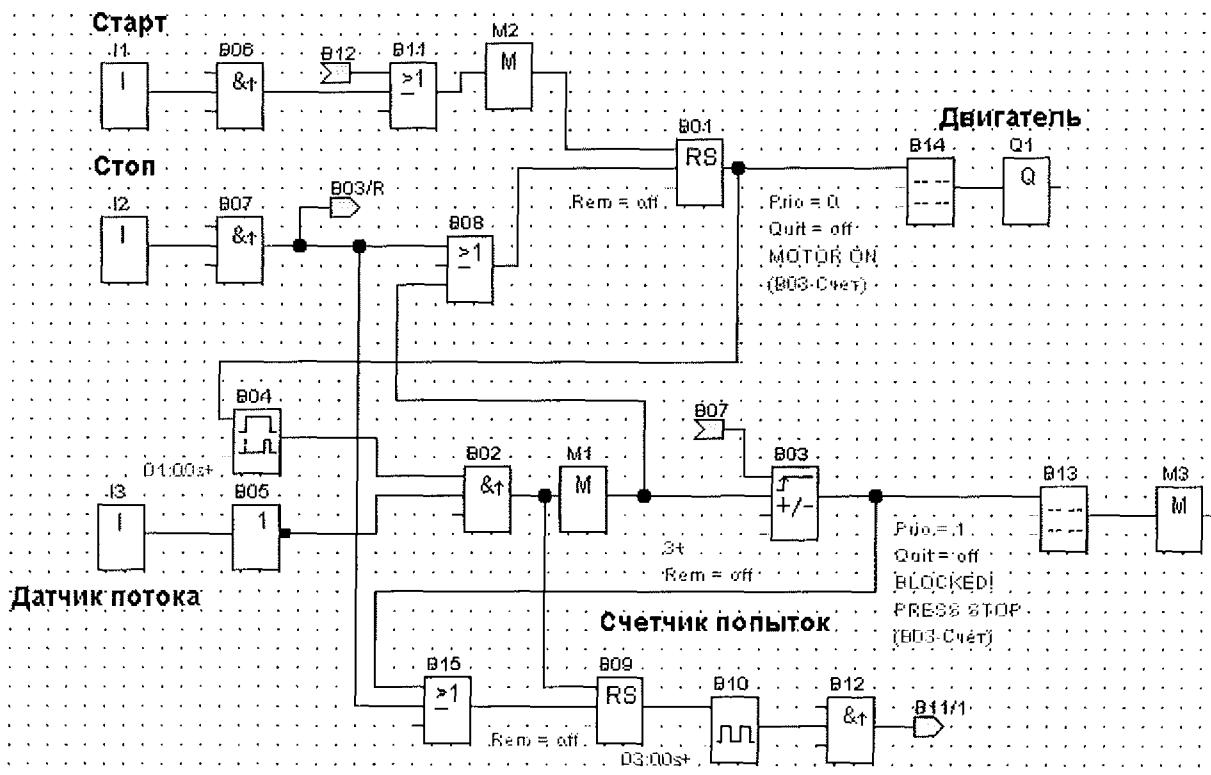


Рис. 4.6. Система управления приточной вентиляцией

Сложные задачи в промышленной автоматике принято решать следующим образом: разбивать основную задачу на несколько простых задач, реализовать каждую из них с помощью средств отладки, связать полученные решения в единый алгоритм, устранив, при необходимости, возникающие противоречия. В данном примере проект можно разделить на следующие задачи: штатный запуск и останов, детектирование аварии при запуске, обеспечение автоматического запуска и подсчет попыток автозапуска. Рассмотрим реализацию каждого фрагмента.

**Штатный запуск и останов.** Данная задача решается с помощью RS-триггера (блок B01), установка и сброс которого производятся по переднему фронту кнопок I1 и I2. Промежуточные элементы на данном этапе можно не принимать в расчет, так как элементы «ИЛИ» и меркер M2 не оказывают влияния на алгоритм запуска и останова.

**Детектирование аварии при запуске.** Задача решена следующим образом – установка триггера запуска B01 запускает таймер B04, на выходе которого формируется логическая «1» (в настройках блока указан интервал 1 с). Если при этом не сработал датчик потока (инфэрсия I2 – «1»), то на выходе элемента «И по фронту» (B02) формируется короткий импульс, который через элемент «ИЛИ» сбрасывает триггер, отключая двигатель. Дополнительно, этим импульсом устанавливается триггер неисправности (B09). Наличие меркера в цепи сброса B01 обусловлено тем, что образовалась перекрестная ссылка, требующая задержки работы на один цикл (меркер введен в схему по подсказке редактора).

**Подсчет попыток автозапуска.** Функция реализована при помощи блока B03 – счетчик событий. В настройках счетчика указано число попыток, равное 3. Если блок B02 выдает 3 импульса (что происходит в том случае, когда возникают 3 попытки неудачного запуска), на выходе счетчика формируется логическая «1», которая вызывает на экран аварийное сообщение. Сброс счетчика осуществляется подачей на его вход R (верхний по схеме) испальса от кнопки сброса.

*Примечание. В данной программе применен способ разрыва связи при помощи ссылок. Любую связь можно разорвать, выполнив соответствующую команду в контекстном меню по данной связи. При этом линия будет разорвана с образованием меток «Источник-приемник».*

**Обеспечение автоматического запуска.** Автозапуск осуществляется следующим образом. Если установлен триггер неисправности B09, включается тактовый генератор с длительностью импульса 3 с (установлен в параметрах) и скважностью 2. Тактовые импульсы через блок B12 поступают на триггер B02 через элемент «ИЛИ» (запуск или пользователем, или автоматический).

Сброс триггера неисправности осуществляется либо переполнением счетчика событий (авария) или с помощью I2, что реализовано при помощи элемента B015.

Переведя редактор в режим эмуляции, произведем штатный запуск, нажав последовательно на I1 и I3 в течение интервала времени 1 с. В результате на экран будет выдано сообщение, показанное на рис. 4.7, а.

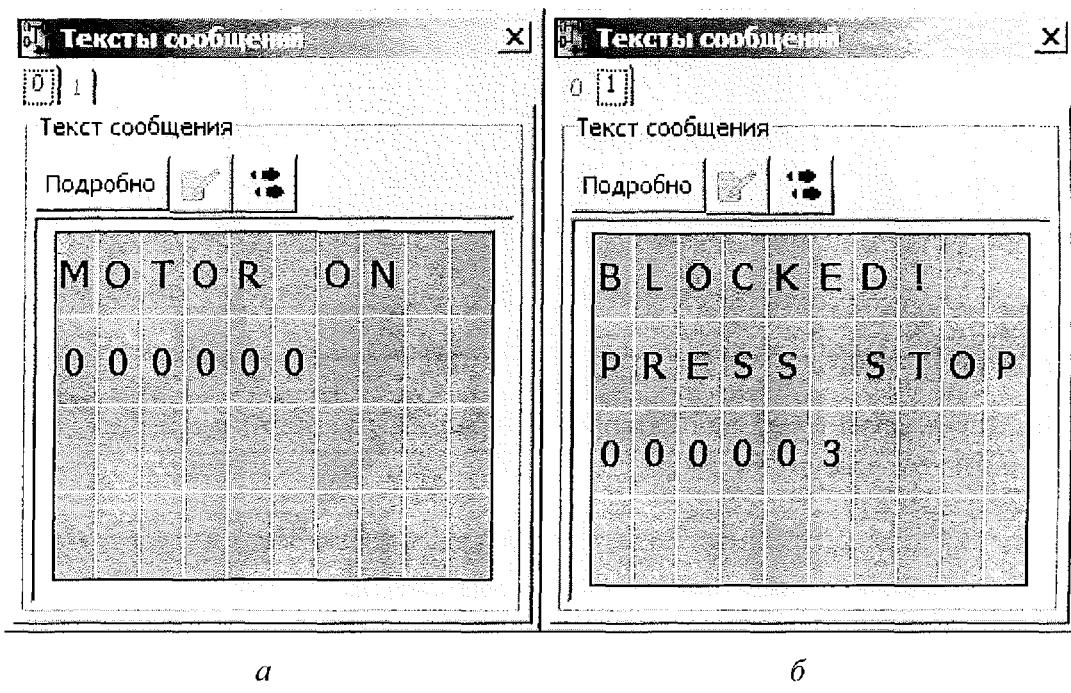


Рис. 4.7. Сообщения о режимах работы установки: а – штатный режим, б – аварийный режим

Выключив двигатель по кнопке I2, вновь нажмем I1 при I3=0. Система установит триггер запуска, включив тактовый генератор. После этого запуск будет произведен 3 раза в автоматическом режиме. Если после первой или второй попыток задействовать датчик потока, двигатель запустится, и на экране будет отображено число попыток до запуска. Если число попыток было превышено, на экран выводится сообщение, показанное на рис. 4.7, б.

Для индикации попыток запуска в сообщения был внедрен параметр «Counter» из блока В03. Загрузив к контроллеру программу, можно убедиться в ее работоспособности, имитируя запуск и останов двигателя.

### 4.3. Варианты заданий

1. Согласно варианту задания, разработать систему управления объектом, используя базовые и специальные функции LOGO!. Для имитации датчиков и органов управления использовать выключатели, подключенные к входам I1-I4, для имитации двигателей и других видов нагрузки использовать индикаторы, подключенные к выходам Q1-Q4. (прил. 1).
2. Произвести отладку программы в режиме эмуляции.
3. Если программа работает правильно, загрузить ее в LOGO! и выполнить.

Схема подключения внешних цепей приведена в прил. 1.

*Примечание:* Вид программы может быть любым (LAD или FBD)

#### *Вариант № 1*

На автостоянке имеется ограниченное число мест (рис. 4.8).

Входной и выходной фотодатчики ФД1 и ФД2 подключены к входам I1 и I2 соответственно. Переключение сигналов светофора осуществляется контактом Q1. Кнопка сброса подключена к входу I3.

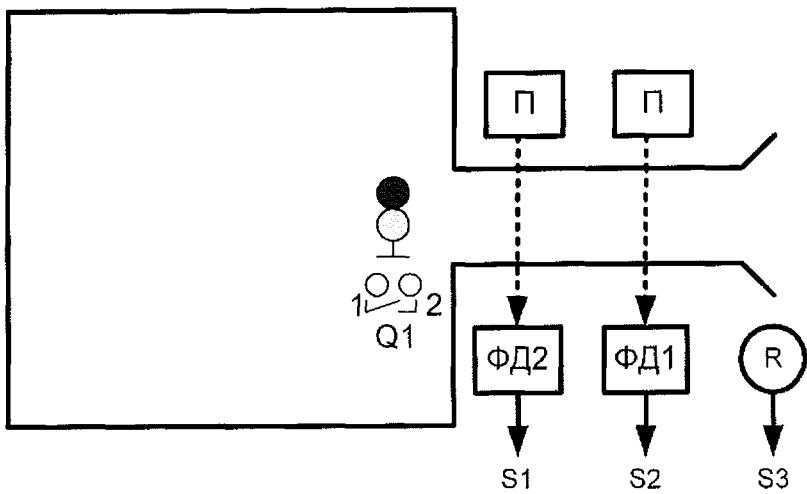


Рис. 4.8. Схема автостоянки

Алгоритм функционирования системы:

1. Входной светофор должен автоматически переключаться с зеленого на красный, когда все места заняты.
2. При освобождении мест включается зеленый.
3. Сброс счетчика производится при нажатии на кнопку сброса.
4. Необходимо выводить сообщение с предельным и текущим количеством машин, а также предупреждение о переполнении.

### *Вариант № 2*

Разработать систему ограничения доступа к объекту (рис. 4.9).

Алгоритм функционирования системы: при нажатии на кнопку S1 система переходит в режим ввода кода. При этом через заданное время загорается лампа, управляемая контактами реле Q1. В момент, когда лампа горит, необходимо ввести первую цифру кода, нажимая на S2 заданное количество раз. Далее лампа гаснет и через заданное время снова включается и вводится вторая цифра кода, после чего цикл повторяется. Количество цифр кода – 3. В случае правильного ввода кода в целом, загорается вторая лампа, которая сигнализирует о том, что код введен правильно. Отслеживание правильности кода производится только после ввода последней цифры.

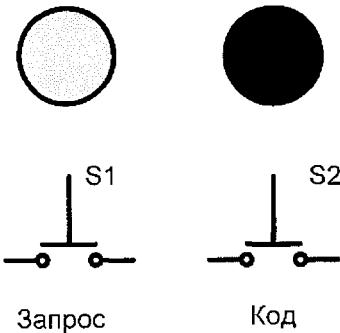
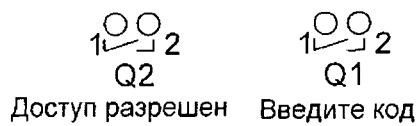


Рис. 4.9. Система ограничения доступа к объекту

### *Вариант № 3*

Имеется три одинаковых потребителя (например, двигатели), которые подключены к сети через контакты реле Q1, Q2, Q3. Два потребителя должны постоянно работать, третий находится в резерве (рис. 4.10).

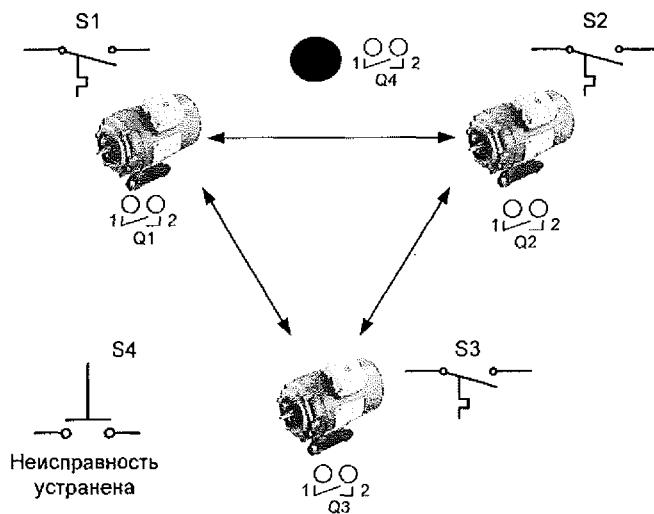


Рис. 4.10. Система автоматического резервирования

Два потребителя работают заданное время (например, Q1 и Q2), после чего включается соседняя пара (Q2 и Q3 и далее Q3 и Q4). Процесс повторяется циклически до тех пор, пока не обнаружится перегрев одного из потребителей. Неисправность имитируется кнопками S1, S2, S3 потребителей

Q1, Q2, Q3 соответственно. В случае перегрева потребитель игнорируется системой, и продолжают работать только два из них, причем резервный занимает место неисправного. Перегрев сигнализируется общим аварийным сигналом (выход Q4). Если перегрев устранен и нажата квитирующая кнопка, подключенная к входу I4, то LOGO! переходит в исходное состояние. Необходимо предусмотреть вывод сообщения о неисправном двигателе.

#### *Вариант № 4*

С помощью кнопки повышения уровня мощности S1 двигатель запускается на уровень 1. Каждым следующим нажатием кнопки двигатель переключается на один уровень мощности выше. Понижение уровня мощности осуществляется кнопкой понижения уровня мощности S2. При изменении уровня мощности подключается другой выход LOGO!, замыкая очередной резистор в цепи питания. На дисплей выводится сообщение о текущей ступени мощности. Схема регулятора представлена рис. 4.11.

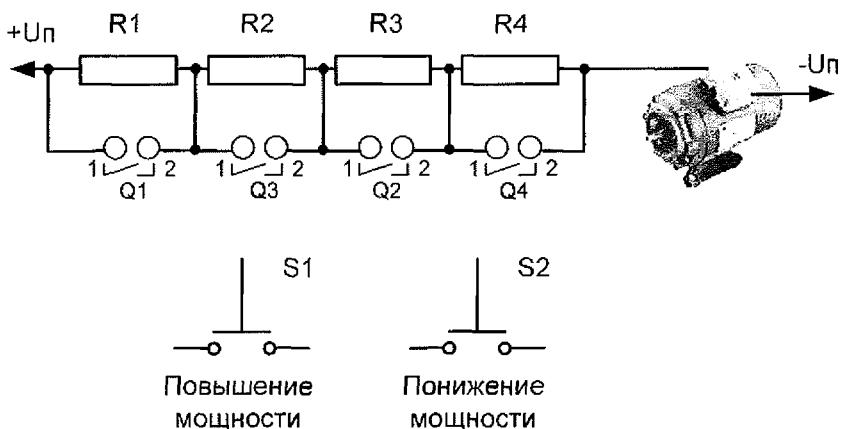


Рис. 4.11. Система ступенчатой регулировки мощности

#### *Вариант № 5*

Ворота управляются двигателем, направление вращения которого задается подачей питания на соответствующую обмотку с помощью контактов Q1 и Q2. Датчики положения S4 и S5 фиксируют положение ворот

(«открыто - закрыто»). Световой индикатор подключен через контакты реле Q3 (рис. 4.12).

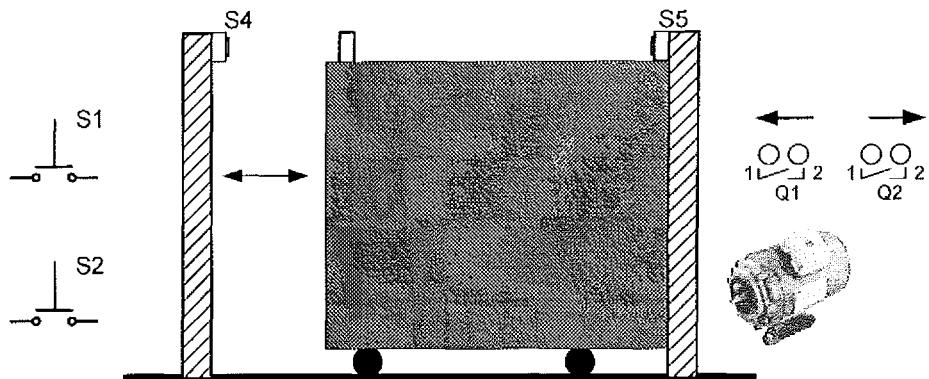


Рис. 4.12. Система управления воротами

Алгоритм функционирования системы:

1. При подаче питания система переходит в ручной режим, управляемый кнопками S1 и S2 без фиксации.
2. Если в закрытом положении нажать кнопку S1, включается двигатель на открытие и система ожидает срабатывания концевого выключателя S5.
3. Если по истечении заданного времени выключатель S5 не сработал, система пытается устранить препятствие, энергичным реверсированием двигателя (заданное количество реверсов), после чего вновь пытается открыть ворота. Если ситуация не изменилась, на панель оператора выводится сообщение о препятствии, мигающий сигнал с различной частотой.
4. Если при закрытии ворот происходит похожая ситуация, система действует аналогичным образом.

## 5. ФУНКЦИИ РАБОТЫ С АНАЛОГОВЫМИ ВХОДАМИ LOGO!

### 5.1. Подключение аналоговых датчиков к LOGO!

Во многих вариантах исполнения LOGO! имеется возможность подключения датчиков с аналоговым выходом. Некоторые модели LOGO! имеют встроенные входы, но для всех версий предусмотрен специальный модуль расширения LOGO! AM2. Основной функцией этого модуля является предоставление пользователю в программе двух точек для ввода аналогового сигнала, который может быть сигналом тока или напряжения.

На рис. 5.1 показана схема подключения источников сигнала к модулю AM2.

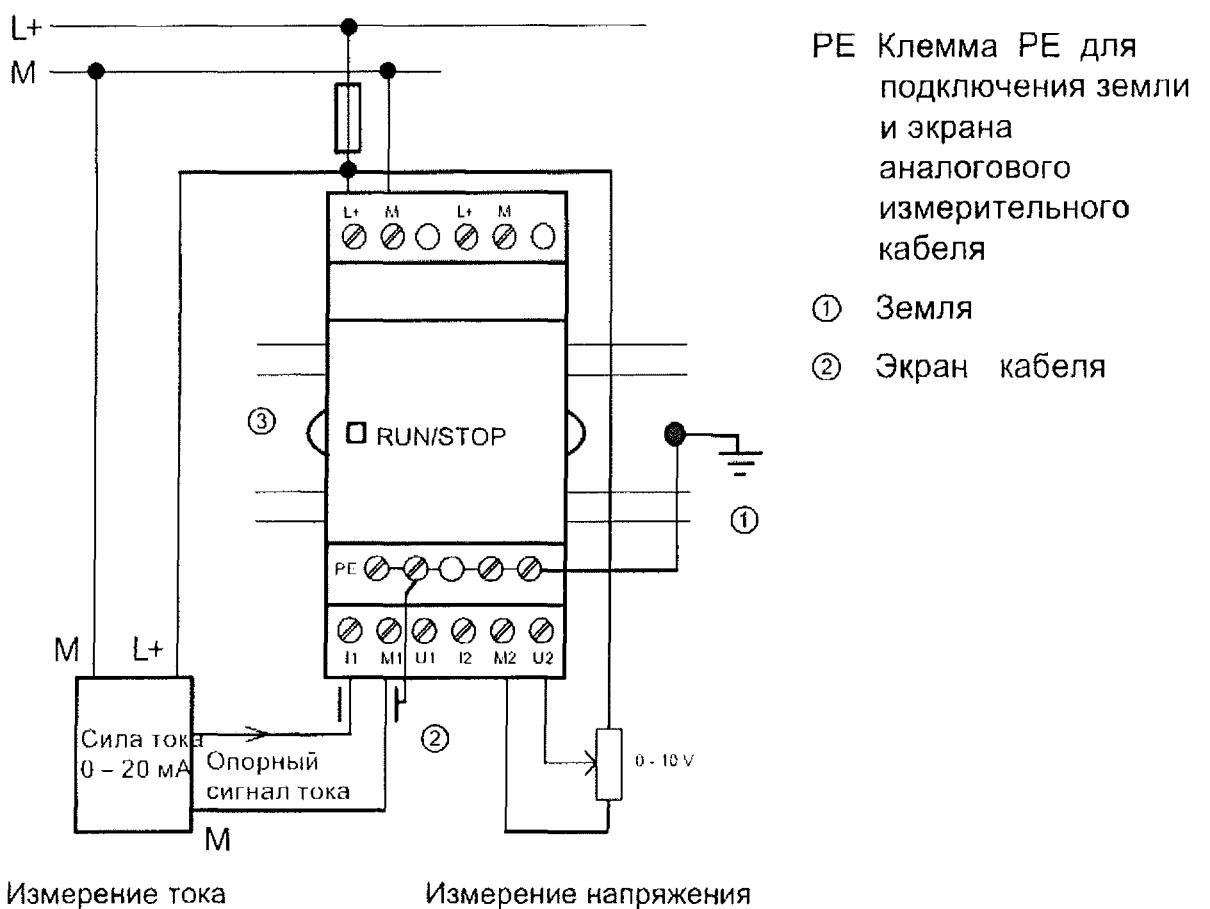


Рис. 5.1. Подключение внешних цепей к аналоговому модулю

Каждый вход имеет три клеммы – М, І и У. При подключении источника тока используются клеммы М и І, при подключении источника напряжения – М и У. Таким образом, каждому входу жестко привязан свой источник сигнала, модуль позволяет настроить тип используемого входа программно.

Токовые входы LOGO! могут использоваться для подключения измерительных преобразователей, выходным сигналом которых является сигнал тока (стандартные диапазоны – 0...+20 мА, +4...+20 мА) – датчики давления, температуры, расхода, уровня и т. п. Входы по напряжению имеют один диапазон 0...+10 В.

## **5.2. Функции работы с аналоговыми входами**

Если в программе пользователя используется аналоговый вход, то его значение представляется в виде слова, разрядность которого зависит от точности представления результата. Например, в ПЛК S7-300 используются 16-разрядные АЦП с диапазоном -32768...+32767, в S7-200 – 12 разрядный АЦП с диапазоном 0...4000. Однако, учитывая специфику программы в LOGO! (отсутствие математической обработки), информация с аналогового входа может быть обработана с помощью некоторой функции, результатом которой является только либо истинное, либо ложное значение. Примером такой функции является компаратор напряжения, сравнивающий значение некоторой уставки с текущим значением, поступающим на вход. Результат логической операции такой функции может быть использован в дальнейшем другими блоками программы. По такому принципу работают все аналоговые функции, имеющиеся в библиотеке LOGO!. В версии 0ВАЗ таких функций две – аналоговый триггер и аналоговый компаратор.

Напряжение от 0 до 10 В на входе AI преобразуется во внутренний диапазон значений от 0 до 1000. В более старших версиях LOGO! Единицы приводятся к диапазону 0...+10 В с точностью 0.1 В. Входное напряжение,

превышающее 10 В, клипируется до величины 1000. Так как у пользователя не всегда имеется возможность представлять входную величину в таких единицах, функции LOGO! предоставляют возможность умножать цифровые значения на коэффициент усиления (Gain), а затем сдвигать нулевую точку диапазона значений (Offset). Это позволяет выводить на дисплее LOGO! аналоговую величину, которая соответствует фактически измеренному значению.

На рис. 5.2 показана временная диаграмма работы аналогового порогового выключателя (триггера). Функция параметрируется двумя значениями: верхним и нижним порогом срабатывания (ON, OFF). Если значение аналогового сигнала ниже верхнего порога, выход Q остается выключенным.

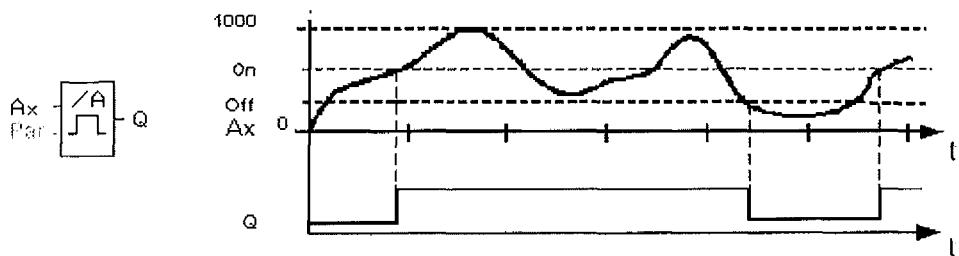


Рис. 5.2. Временная диаграмма работы аналогового компаратора

Если значение входа превышает верхний порог, выход включается и остается включенным до тех пор, пока значение входа не станет ниже порога выключения OFF. Схемотехническим аналогом такой функции является двухпороговый компаратор с гистерезисом, часто используемый в промышленной электронике. На основе таких функций можно строить различные системы, реализующие функции допускового контроля, сигнализаторы предельного уровня, терmostаты, системы поддерживания уровня веществ в емкостях и т. д.

На рис. 5.3 показан внешний вид и временные диаграммы функции аналогового компаратора.

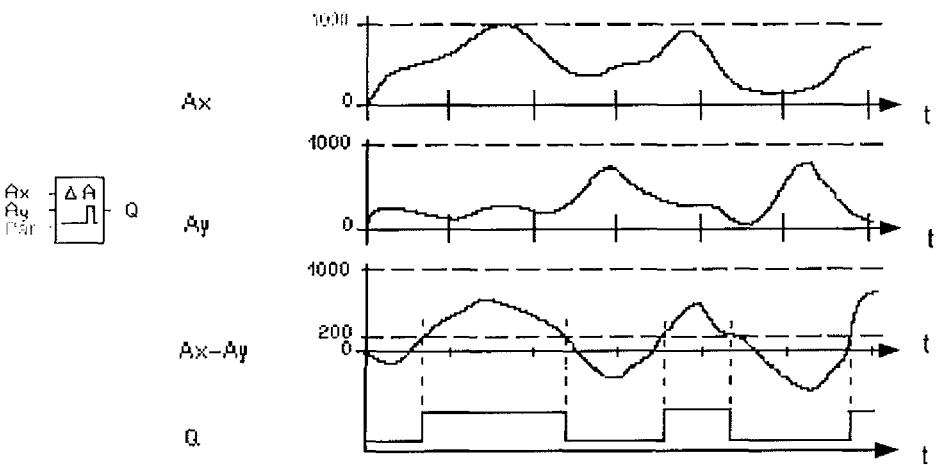


Рис. 5.3. Временные диаграммы работы аналогового компаратора

Работа этой функции отличается от предыдущей тем, что компаратор сравнивает значение аналогового входа не с константой, а со значением другого аналогового входа. Если разность значений  $A_x$  и  $A_y$  превышает некоторый заранее заданный порог (зона нечувствительности), выход  $Q$  включается.

Классическим примером применения этой функции можно назвать использование ее в нелинейной системе автоматического регулирования (релейный регулятор температуры). При этом на вход  $A_x$  подается значение задатчика температуры (например, с потенциометра), а на  $A_y$  подается параметр, представляющий собой напряжение обратной связи. Необходимое значение гистерезиса при этом повышает стабильность работы системы.

В различных версиях LOGO! имеются некоторые дополнительные функции, являющиеся разновидностью рассмотренных выше, однако логика их работы изменяется несущественно.

### 5.3. Масштабирование аналоговых величин

В приведенных выше временных диаграммах все значения аналоговых входов, а также уставки, приведены в условных единицах в диапазоне 0...1000. Данный диапазон может косвенно оценить разрядность АЦП в

LOGO!. Условные единицы облегчают программирование и позволяют в некоторой степени нормировать аналоговые величины. Однако для построения реальных систем необходимо составить уравнение масштабирования. При задании уставки триггера при измерении температуры необходимо иметь соответствие температуры значению диапазона. Например выходное напряжение 0...10 В датчика температуры соответствует реальному диапазону  $T_{min} = -20^{\circ}\text{C}$ ,  $T_{max} = +50^{\circ}\text{C}$ . Также необходимо задавать точность измерения температуры, например, если диапазон условных единиц, заданных в функции, равен 0-1000, при изменении температуры в заданном диапазоне точность будет составлять  $\gamma = (50+20)/1000 = 0,07^{\circ}\text{C}$ , однако такая точность может быть излишней. Иногда необходимо смещать значение аналогового входа на некоторую величину по отношению к горизонтальной оси. Например, при измерении температуры в диапазоне  $-20\dots +100^{\circ}\text{C}$ , напряжение датчика будет изменяться в пределах 0-10 В, для приведения характеристики к диапазону 0...1000 необходимо сместить точку отсчета по шкале температуры на число единиц, соответствующих  $+20^{\circ}\text{C}$ .

Масштабирование аналоговых значений производится через окно свойств аналоговой функции (рис. 5.4).

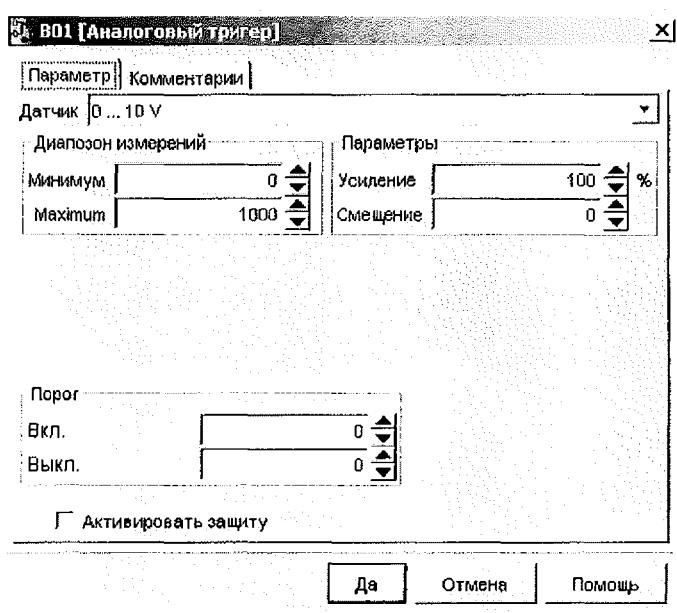


Рис. 5.4. Окно параметрирования аналогового триггера

В опции «Датчик» выбирается тип аналогового входа – токовый или вход по напряжению, в зависимости от подключенного оборудования. Для токового входа дополнительно выбирается диапазон токов.

В опции «Параметры» выбирается усиление (0 – 100 %) и смещение (-999 – +999). Усилинию 100 % соответствует максимальное значение условных единиц равное 1000, таким образом цена деления одной условной единицы равна 0,01 В, при использовании входа по напряжению. При изменении усиления производится автоматический пересчет максимальной величины условных единиц. Так, например, при задании усиления 50 % диапазон условных единиц будет равен 0 – 500, а цена деления составит 0,02 В. Если усиление будет равно 1 %, то диапазон условных единиц составит 0...10, при цене деления 1 В. Величина усиления может лежать в диапазоне 1...1000 %, что соответствует верхним границам диапазона 0...10000, максимальная точность в последнем случае может достигать 0,001 В.

Величина смещения представляет собой константу, которая складывается с нижней и верхней границей диапазона условных единиц. Масштаб этой константы зависит от усиления и рассчитывается следующим образом:  $D = S \cdot K / 100$ , где  $D$  – величина смещения, складываемая с границей диапазона,  $S$  – смещение,  $K$  – усиление в процентах. Например, если усиление составляет 1000 %, а смещение равно 1, то нижняя граница диапазона составит 10, а верхняя – 10010.

Таким образом, расчет диапазона изменения величины в условных единицах можно произвести по формуле

$$N = \left( 10 + \frac{S}{100} \right) \cdot K,$$

где  $N$  – значение в условных единицах;  $K$  – усиление.

Расчет может производиться с погрешностью, так как в LOGO! все параметры не допускают дробных значений. Так как границы диапазона и параметры масштабирования взаимозависимы, то пользователю дается возможность задавать масштаб, изменяя те параметры, которые в том или

иным случае удобно задавать, при этом соседняя пара параметров установится автоматически.

Любой параметр функции, работающей с аналоговым входом, может быть выведен в условных единицах на дисплей LOGO!. Конфигурирование сообщения производится стандартным способом, как и для любой специальной функции.

**Пример.** К аналоговому модулю LOGO! подключен датчик температуры, работающий в диапазоне  $-50 \dots +50^{\circ}\text{C}$ . Необходимо отображать текущее значение температуры. Если значение температуры превышает  $25^{\circ}\text{C}$ , выдать соответствующее сообщение, отключить сообщение при снижении температуры до  $10^{\circ}\text{C}$ . Точность срабатывания реле  $-0,1^{\circ}\text{C}$ . Блок-схема, реализующая данный алгоритм, приведена на рис. 5.5.

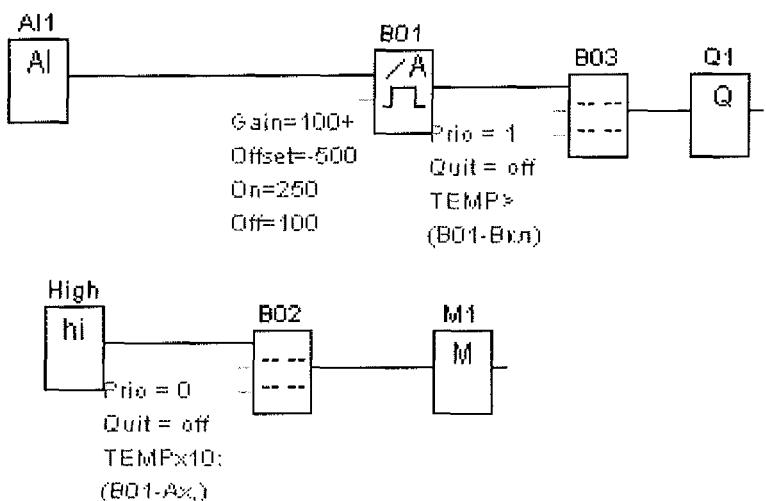


Рис. 5.5. Блок-схема измерения температуры

Вход AI1 подключен к входу аналогового триггера, при срабатывании которого выдается сообщение с высшим приоритетом, а также включается выход Q1. Если триггер не сработал, выдается сообщение с внедренными данными, которые содержат значение входа AI в условных единицах.

Масштабирование аналоговой величины производится следующим образом: так как в полном диапазоне температур  $100^{\circ}\text{C}$  необходима точность  $0,1^{\circ}\text{C}$ , то полный диапазон условных единиц должен быть равен

0...1000. Для корректного вывода данных на дисплей (без десятичной точки) необходимо сместить рабочую точку на -500 единиц. Таким образом, при нижней и верхней границе -500 и + 500 соответственно, значение усиления будет равно 100 %, величина смещения составит -500 единиц.

## 5.4. Варианты заданий

### *Вариант №1*

Разработать систему вытяжной вентиляции для помещения (рис. 5.6).

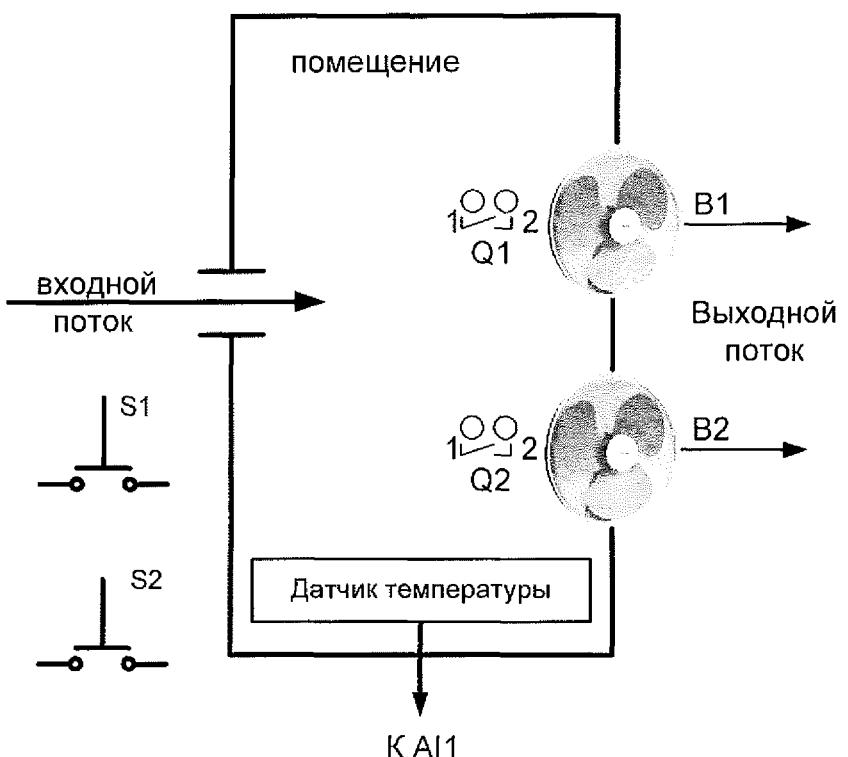


Рис. 5.6. Схема вытяжной вентиляции помещения

В помещении находятся два вентилятора, подключенные к LOGO! через контакты реле Q1 и Q2, датчик температуры с выходом по напряжению, подключенный к аналоговому входу AI1. Кнопки S1 и S2 предназначены для управления системой.

Алгоритм работы системы: если температура в помещении достигает определенного верхнего предела, включается вентилятор B1 и работает

заданное время, по истечению которого выключается. Если за это время температура не снизилась до заданного значения, через определенное время паузы вентилятор вновь включается. Если за определенное количество циклов включения-выключения В1 не произошло снижения температуры, выключается вентилятор В2 и работает непрерывно. Если даже и в этом случае температура не снизилась, выдается сообщение о выходе системы из под контроля (которое должно быть квитировано), и вентиляторы переключаются на ручной режим от S1 и S2. При одновременном нажатии на S1 и S2 система вновь работает в циклическом режиме.

### *Вариант №2*

Разработать систему регулировки жидкости в двух резервуарах (рис.5.7).

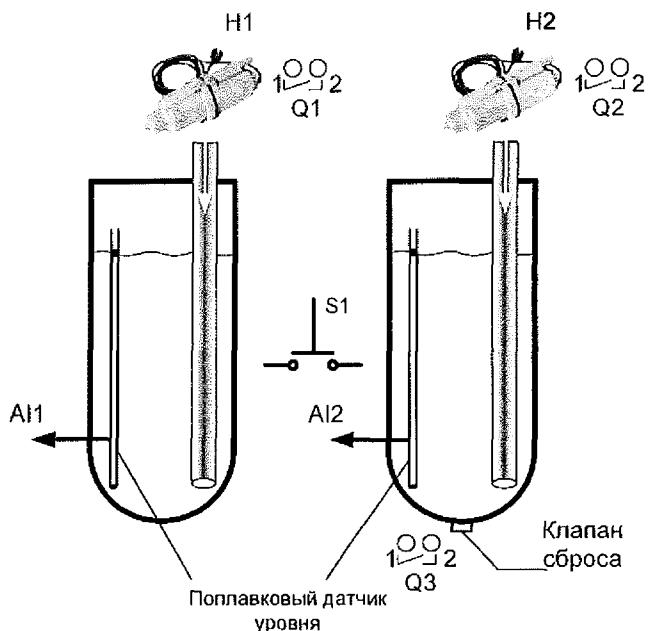


Рис. 5.7. Система регулировки уровня жидкости в двух резервуарах

В каждом из двух одинаковых резервуарах установлены датчики поплавкового типа для измерения уровня жидкости, подключенные к аналоговым входам AI1 и AI2 соответственно. Резервуары наполняются насосами H1 и H2, управляемые контактами реле Q1 и Q2. Контактом Q3

включается клапан слива жидкости из второго резервуара. Кнопкой регулировки уровня S1 регулируется уровень во втором резервуаре.

Алгоритм функционирования системы: система включается при подаче питания. Насос H1 включается (закачивает жидкость) в тот момент, когда уровень жидкости в первом баке достигает заданного нижнего предела, и выключается в момент достижения заданного верхнего предела.

При нажатии на кнопку S1 происходит следующее:

1. Если уровень во втором баке выше, чем в первом, клапан сбрасывает жидкость до момента уравнивания с первым баком.
2. Если уровень во втором баке ниже, чем в первом, H2 закачивает жидкость до момента уравнивания с первым баком.
3. В обоих случаях на дисплее оператора появляется сообщение, которое показывает уровень жидкости в обоих баках.

### *Вариант №3*

Разработать систему управления сушильной камерой (рис. 5.8).

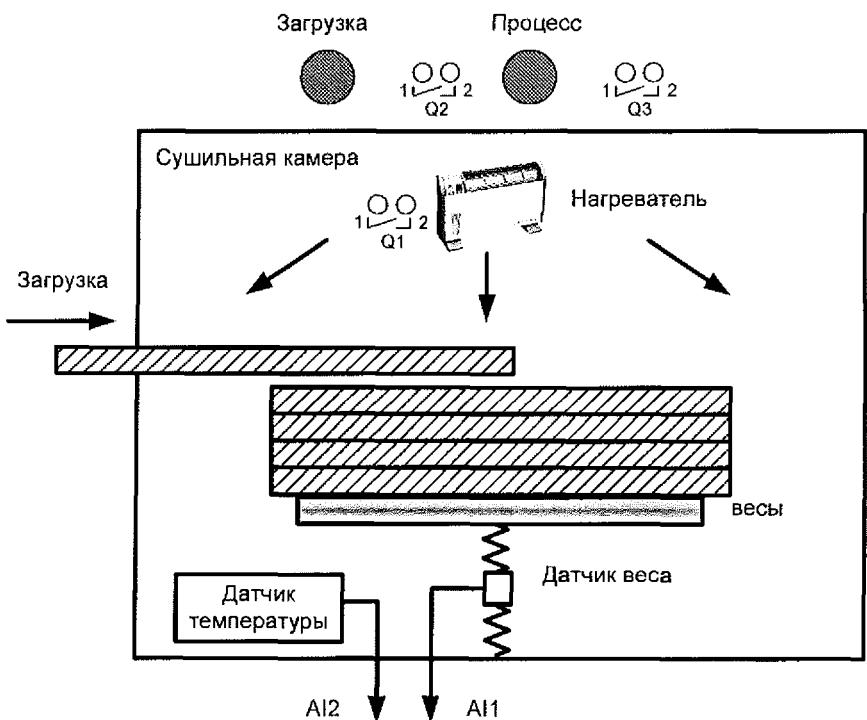


Рис. 5.8. Камера для сушки древесины

Алгоритм функционирования системы: при подаче питания система ожидает загрузки древесины на весы, горит индикатор загрузки (Q2). В системе используется датчик веса, подключенный к аналоговому входу AI1. При достижении максимального уровня загрузки весов система выдает сообщение и автоматически подключает нагреватель, управляемый контактами Q1. Процесс нагрева контролируется датчиком температуры, подключенным к входу AI2. Если температура выше заданного предела, нагреватель выключается, если ниже – включается. Когда вес древесины упадет до заданной величины, система выдает сообщение и нагреватель выключается, индикатор показывает, что система готова для загрузки новой порции.

Процесс сопровождается миганием сигнальной лампы «Процесс» (Q3).

*Примечание: необходимо принять меры для повышения точности весоизмерения в условиях колебания весов при загрузке.*

#### **Вариант №4**

Разработать систему управления рекламной вывеской (рис 5.9).

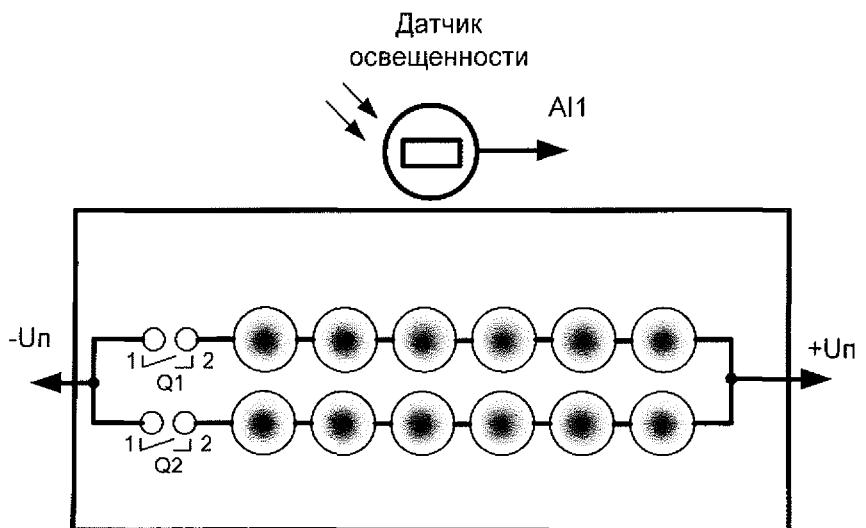


Рис. 5.9. Рекламная вывеска

Рекламная вывеска подсвечивается изнутри двумя группами ламп, управляемых с помощью контактов реле Q1 и Q2. В системе присутствует

датчик внешней освещенности, изолированный от ламп и подключенный к аналоговому входу AI1.

Алгоритм работы системы: вывеска включается спустя некоторое время после падения внешней освещенности на полную мощность. После 24 ч одна из групп ламп гаснет. Выключение вывески происходит по достижению уровня освещенности определенного порога. В следующие сутки процесс повторяется, но после 12 ночи гаснет вторая группа ламп, для обеспечения одинакового времени наработки.

На дисплей выводится состояние групп ламп, уровень освещенности и текущее время.

*Примечание: необходимо предусмотреть защиту от случайного включения или выключения вывески при кратковременном ее освещении или затемнении (например, фарами автомобиля).*

### *Вариант №5*

Разработать систему управления насосной станцией (рис. 5.10).

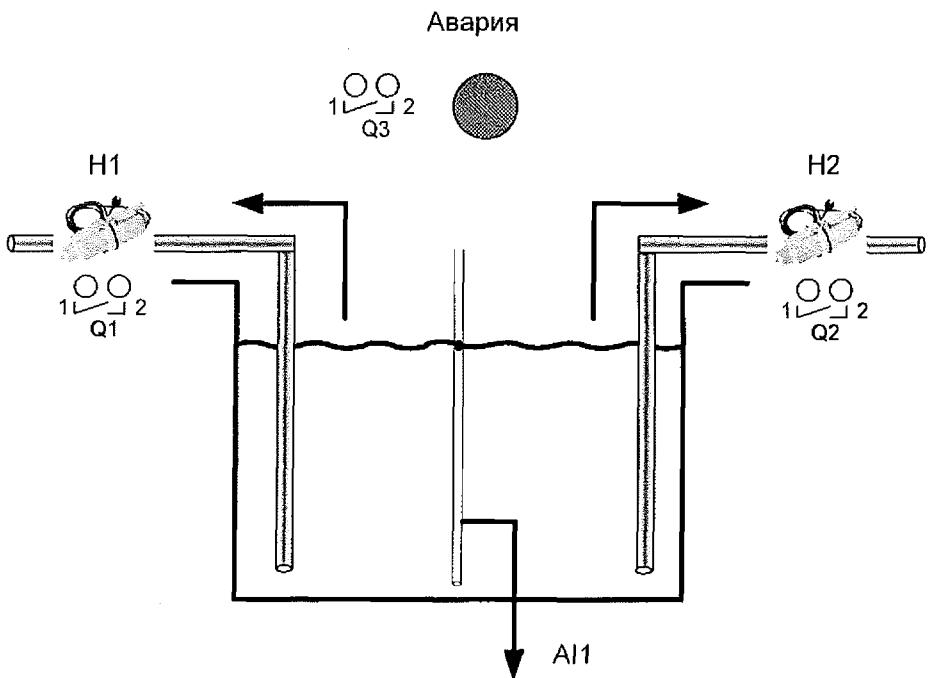


Рис. 5.10. Насосная станция

В системе два насоса Н1 и Н2, управляемые через контакты Q1 и Q2, и поплавковый датчик уровня воды, подключенный к AI1.

Алгоритм функционирования системы: насос Н1 откачивает воду из емкости при достижении уровня верхнего предела. Двигатель отключается при достижении нижнего предела. Если за определенное время уровень не снижается либо остается неизменным, включается второй насос. При включении второго насоса выдается предупреждающее сообщение. Если при работе двух насосов фиксируется повышение уровня воды, делается вывод о выходе системы из под контроля либо неисправности двигателей, выдается аварийное сообщение и мигающий сигнал (Q3), двигатели насосов останавливаются. При превышении уровня на дисплее отображается текущий уровень воды.

## **6. АРХИТЕКТУРА И ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ S7-200**

### **6.1. Особенности архитектуры S7-200**

Программируемые логические контроллеры SIMATIC S7-200 предназначены для построения относительно простых систем автоматического управления, отличающихся минимальными затратами на приобретение аппаратуры и разработку системы. Контроллеры способны работать в реальном масштабе времени и могут быть использованы как для построения узлов локальной автоматики, так и узлов, поддерживающих интенсивный коммуникационный обмен данными через сети Industrial Ethernet, PROFIBUS-DP, MPI, AS-Interface, MPI, PPI, а также через модемы. В общем случае, **ПЛК S7-200** состоит из ряда модулей, главным из которых является центральный процессор (**CPU**). Центральный процессор является базовым модулем, содержащим минимальное количество программируемых входов и выходов, разъем для подключения программатора, разъем питания, а также специальный разъем для подключения внешних модулей расширения. Модули расширения не являются независимыми и предназначены для поддержки встроенных функций центрального процессора, тогда как CPU может работать независимо от модулей расширения с выполнением минимального набора функций. Для S7-200 выпускаются следующие модули:

- модули аналогового ввода-вывода;
- модули дискретного ввода-вывода;
- модули для измерения температуры посредством терморезисторов датчиков РТ;
- модули для измерения температуры посредством термопар;
- модули управления шаговыми электродвигателями;
- GSM-модем для управления объектами SMS-сообщениями;

- аналоговый модем;
- модули сопряжения с промышленными сетями PROFIBUS, Ethernet и AS-i.

Семейство контроллеров S7-200 состоит из ряда моделей, различающихся по объему памяти, количеству входов и выходов, их конфигурации, дополнительным возможностям. В терминологии фирмы Siemens *программируемым логическим контроллером* (ПЛК) называется полная система автоматизации, состоящая из центрального процессора (CPU) и подключенных к нему модулей расширения. При этом одни и те же модули расширения, спроектированные для семейства S7-200, могут быть подключены к разным CPU. В табл. 6.1 приведены основные характеристики двух различных центральных процессоров CPU222 Relay и CPU226XM

Таблица 6.1

Сравнительные характеристики CPU 222 Relay и CPU 226 XM

Свойство	CPU222 Relay	CPU226XM
Напряжение питания	220В	24В
Тип дискретных выходов	Релейные, 220В – 10А	Транзисторные, 24В-0.5А
Память данных	2 Кб	10 Кб
Память программ	4 Кб	16 Кб
Время выполнения инструкции	0.22 мкс	0.22 мкс
Кол-во модулей расширения	Не более 2	Не более 7
Встроенные дискретные входы	8	24
Встроенные дискретные выходы	6	16
Коммуникационный порт	Один порт RS-485	Два порта RS-485
Встроенный потенциометр	один	два

В функции базового модуля CPU входит:

- поддержка системы команд (функций) ПЛК на языке LAD, FBD и STL, включая вычисления с плавающей точкой;

- обработка ошибок и возможность их программной обработки;
- Выполнение коммуникационных задач – например, обмен данными между CPU;
- организация ввода-вывода для дискретных и аналоговых сигналов;
- обработка аппаратных и программных прерываний;
- возможность работы с блоками данных пользователя;
- самодиагностика ПЛК;
- выполнение рабочего цикла программы под управлением операционной системы CPU;
- поддержка систем человеко-машинного интерфейса (панели оператора TD200, TP070, SCADA-системы WINCC Flexible на базе PC).

Программируемый контроллер – это полноценная вычислительная машина, содержащая все основные узлы современных ЭВМ. Однако, в отличие от универсальных и персональных ЭВМ, архитектура S7-200 имеет некоторые отличия, обусловленные областью применения.

Ниже перечислены основные базовые принципы архитектуры S7-200.

**Логическое разделение памяти программ и памяти данных пользователя.** Модификация памяти программ возможна только со стороны инженера-программиста на этапе ее смены либо отладки. Любая программа может обращаться только к заранее известным ресурсам, исключая собственную модификацию.

**Программная модель ресурсов контроллера.** Любые ресурсы контроллера для конечного пользователя представлены как специальные области памяти. Например, существует область памяти входов и выходов, память данных, память аккумуляторов, память специальных переменных и т.д. При этом программа обращается не к физическим входам-выходам (портам), а к соответствующим ячейкам «образа» этих входов. Обновление образа осуществляет операционная система ПЛК в реальном времени.

**Принцип выполнения программы в реальном времени.** Семейство S7-200 предназначено для работы в промышленных системах автоматики, где скорость обработки данных – относительная величина. Например, минимальная частота опроса дискретного контактного датчика зависит от времени переходного процесса при коммутации. Так как в большинстве случаев это время относительно велико (5-20 мс), то выполнение программного цикла в большинстве случаев укладывается в данный интервал. Следовательно, нет необходимости «привязывать» выполнение определенных задачами пользователя процедур к процессу сбора данных.

**Принцип рабочего цикла операционной системы.** Любая программа, написанная пользователем, состоит из конечного числа команд, предоставляемых пользователю операционной системой. Вмешательство пользователя в работу ОС невозможно, как невозможно и низкоуровневое программирование. Вместо этого, ОС контроллера предоставляет пользователю конечный набор инструкций, которые оперируют с заранее определенными ресурсами. Рабочий цикл ОС состоит из следующих последовательных операций:

- обновление образа входов;
- последовательное выполнение инструкций пользователя (программа пользователя);
- выполнение самодиагностики;
- выполнение коммуникационных задач, определяемых командами пользователя в программе;
- обновление образа выходов.

Таким образом, любая программа имеет циклический принцип выполнения, жестко определяемый ОС контроллера. Данный принцип исключает «зависание» программы пользователя, что является полезным свойством для систем АСУТП, работающих в реальном времени.

**Принцип реманентной памяти.** В оперативной памяти (ОЗУ) хранится программа пользователя, которая выполняется в данный момент, а

так же конфигурационные данные ПЛК, память данных (V-память) и память меркеров (M-память, битовая память данных). В постоянной памяти (ПЗУ с электрическим стиранием и записью) хранятся та же программа и данные, но долговременно. При включении питания вся информация из ПЗУ передается в ОЗУ, таким образом, обеспечивается независимая от пользователя загрузка существующей программы и данных проекта. Такая организация обеспечивает высокую надежность и простоту обслуживания ПЛК - программатор используется единожды для загрузки программ в ПЗУ, дальнейшие действия по запуску программы (например, после аварийного отключения питания) берет на себя операционная система ПЛК, которая хранится в специальном ПЗУ, недоступном для пользователя.

**Принцип позиционной адресации внешних модулей.** Внешние модули подключаются к ПЛК посредством специального разъема (шины внешних модулей). При добавлении модуля пользователю становятся доступными ресурсы этого модуля, однако общее адресное пространство ресурсов конечно. Фактические адреса ресурсов модуля становятся известными только на этапе подключения и зависят от позиции модуля по отношению к ПЛК. Например, если общее адресное пространство ПЛК составляет 256 дискретных входов, используется CPU 222 с 8 входами, то первый подключенный модуль имеет адреса входов с 9 по 15.

Обобщенная архитектура ПЛК приведена на рис. 6.1.

Программно-доступная память данных ПЛК организована как набор файловых регистров, каждый из которых имеет собственное назначение и адресацию. Обращение к этим ячейкам производится по идентификатору области. Например, байт VB4 (байт из памяти V) и MB4 (байт из памяти M) находятся в разных областях памяти и могут иметь одинаковое применение, а байт SMB4 (байт специальной памяти) может быть использован только для определенных целей, зависящих от назначения этого байта в таблице специальных функций. Размер областей памяти варьируется в зависимости от конкретного типа CPU.

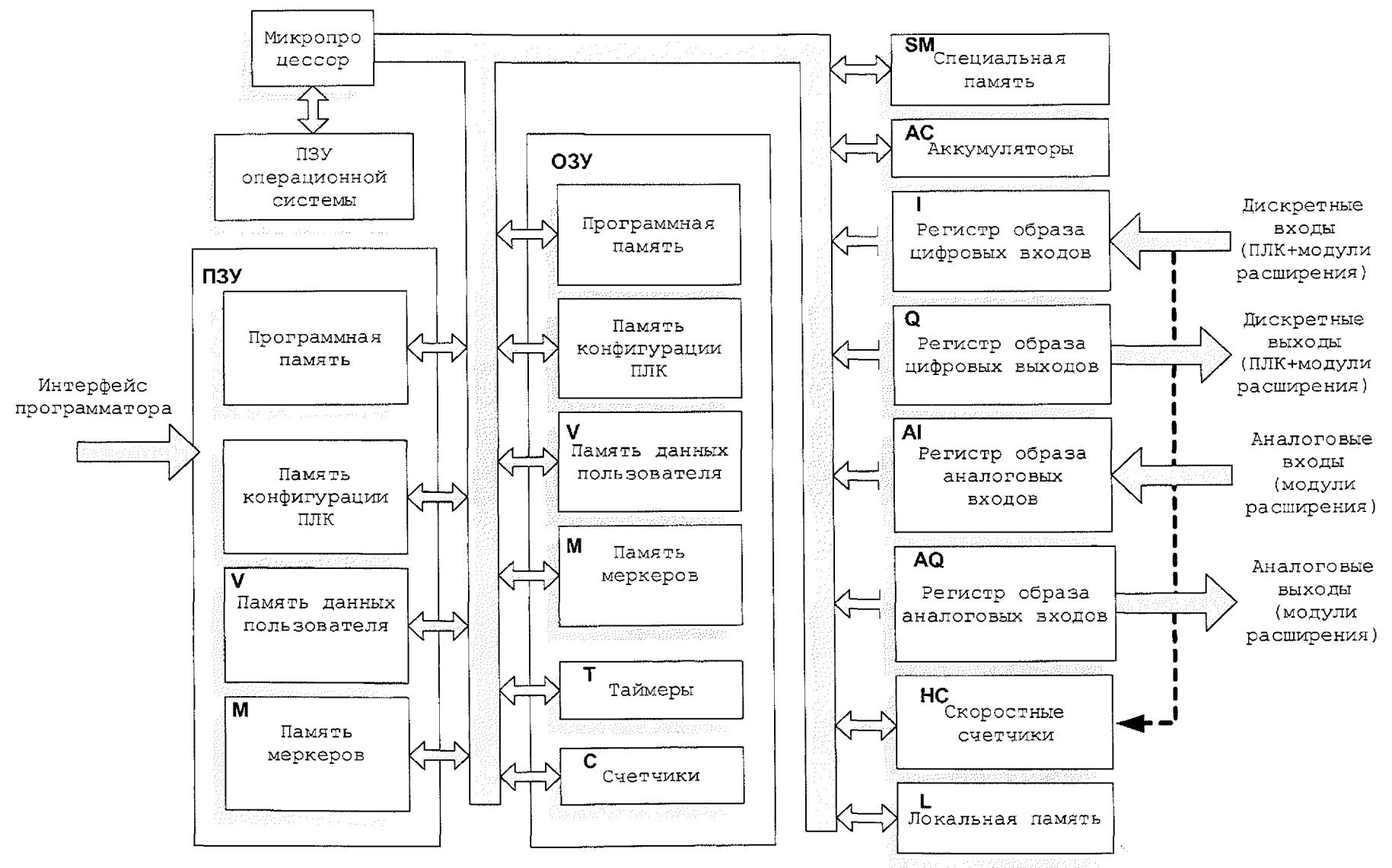


Рис. 6.1. Области памяти S7-200

Пользователю доступны следующие области памяти:

- **V – память.** Основная память данных ПЛК, может хранить промежуточные данные и данные пользователя (табличные значения, рецепты, константы);
- **M-память.** Память меркеров, в основном используется как набор битовых переменных (флаги пользователя);
- **T-память.** Память, содержащая переменные таймеров;
- **C-память.** Память, содержащая переменные счетчиков событий;
- **SM-память.** Представляет собой регистры данных, отражающие системные данные ПЛК (аппаратная диагностика, диагностика ошибок выполнения программы, настройка параметров ПЛК, управление встроенными функциями, дополнительные возможности, предоставляемые операционной системой);
- **AC-память.** Набор регистров-аккумуляторов, могут использоваться как ячейки памяти и регистры operandов для ряда специфических операций;
- **I-память.** Организована как регистры образа входов, отображает значения дискретных входов в текущем программном цикле;
- **Q-память.** Организована как регистры образа выходов, отображает значения дискретных выходов в текущем программном цикле;
- **AI- память.** Организована как набор 16-разрядных регистров аналоговых входов (АЦП);
- **AQ-память.** Организована как набор 16-разрядных регистров аналоговых выходов (ЦАП);
- **HC-память.** Организована как набор регистров, содержащих текущее значение скоростных счетчиков внешних событий, не связанных с выполнением программного цикла ПЛК. Скоростной счетчик HC увеличивает или уменьшает на единицу свое значение, независимо от программы;

- **L-память.** Память локального стека, предназначена для временного хранения промежуточных данных, а так же для передачи параметров в подпрограммы и обработчики прерываний.

Память конфигурации ПЛК и память ОС являются системными. Память ОС недоступна для пользователя, память конфигурации может быть изменена при программировании ПЛК и содержит ряд системных параметров (пароль доступа, сетевой адрес, настройки интерфейса, настройка реманентных областей, и т.д.).

Области памяти V, M и программная память могут быть сохранены в ПЗУ ПЛК и быть восстановлены при перезапуске питания. Память программ и данных сохраняются в ПЗУ по умолчанию на этапе программирования контроллера, M-память сохраняется в том случае, если пользователь определил ее как реманентную (сохраняемую). Более подробно об операциях сохранения и восстановления областей памяти можно узнать в [2].

## 6.2. Технология программирования S7-200

Ввод в эксплуатацию ПЛК начинается с этапа монтажа [2]. При эксплуатации ПЛК может находиться в 4 режимах:

1. Режим выполнения программы (режим «*RUN*»). В этом случае центральный процессор выполняет программу, ранее введенную в память. При включении питания программа автоматически запускается.
2. Режим «*STOP*». В этом режиме CPU останавливает работу программы до ее следующего запуска.
3. Режим программирования. В этом режиме программа пользователя загружается в память программ посредством связи ПК и CPU с использованием соединительного кабеля, имеющего преобразователь интерфейса RS-232 (USB)- RS-485. В этом режиме работа программы

пользователя прекращается и CPU переходит в режим «STOP». В режиме программирования можно прочитать программу, ранее занесенную в память CPU.

4. Режим терминала. («**TERM**»). В этом режиме осуществляется онлайн-связь CPU и ПК, позволяющая запускать и останавливать программу пользователя из среды программирования, производить мониторинг значений переменных в памяти CPU. В этом режиме пользователь может производить отладку введенной программы. Следует отметить, что фирма Siemens принципиально отказалась от разработки симулятора контроллеров семейства S7-200, поэтому отладку можно производить только этим способом — загрузкой программы в память и исполнением ее под управлением пользователя.

Подключение ПК (используемого как программатор) к CPU не является постоянным, так как в функции первого входит только загрузка и тестирование программ, осуществляемых обслуживающим персоналом производства. На рис. 6.2 показан способ подключения ПК к PLC., а на рис. 6.3 – подключение внешних цепей.

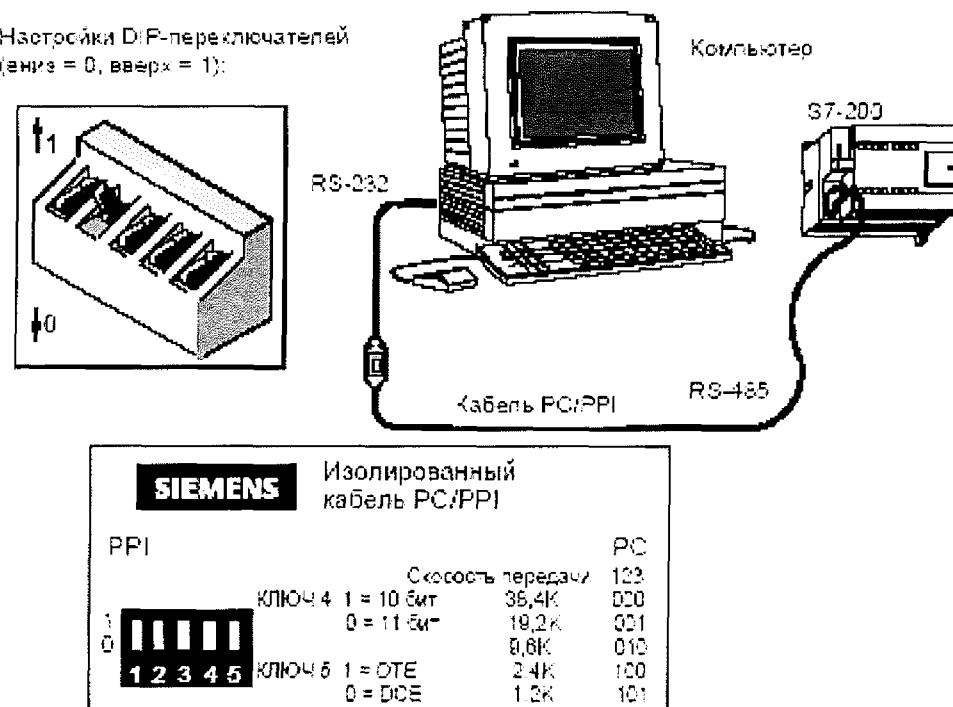


Рис. 6.2. Подключение ПК к S7-200

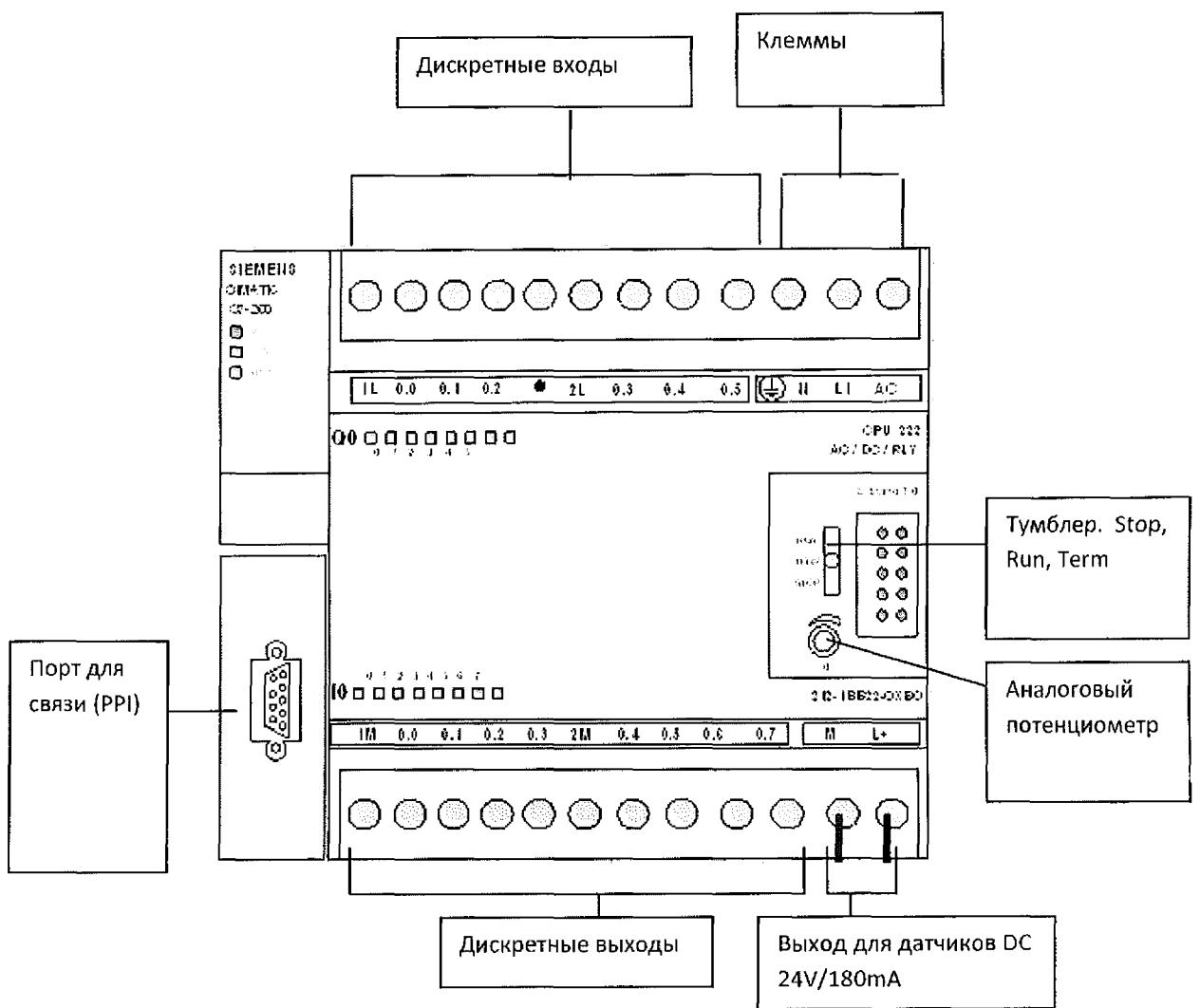


Рис.6.3. Подключение внешних цепей к S7-200

Для разработки программ, их отладки и программирования CPU S7-200 разработан пакет Step 7 MicroWin32. Язык Step 7 – один из наиболее распространенных языков программирования контроллеров, в том числе и S7-300, 400. Step 7 MicroWin 32, в отличие от других пакетов (например, Step 7), ориентирован исключительно на семейство S7-200. Среда программирования предоставляет пользователю несколько стандартных промышленных языков программирования, в частности – язык LAD, FBD и STL.

Общий вид окна среды разработки показан на рис. 6.4.

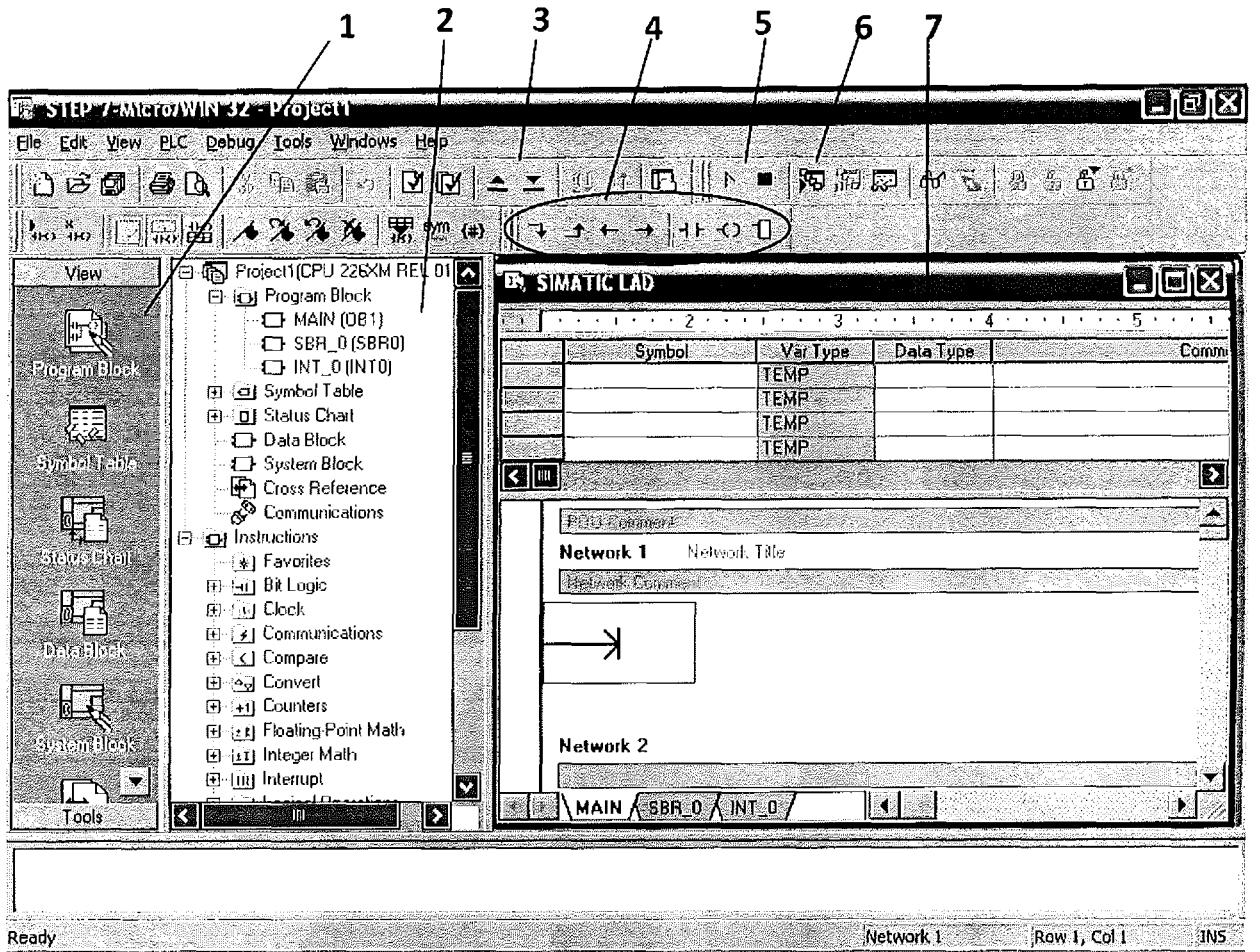


Рис. 6.4. Общий вид окна Step7 MicroWin

Основные компоненты, используемые в приложении STEP7 Micro\Win32:

1 – панель навигации, которая содержит вкладку View, включающую в себя следующие компоненты:

- a) Program Block – редактор текстов программ;
- б) Symbol Table – таблица символьных констант;
- в) Status Chart – таблица состояния;
- г) Data Block – блок данных;
- д) System Block – блок состояния системы (вид для CPU 222);
- е) Cross Reference – просмотр ресурсов и перекрестных ссылок;
- ж) Communications – настройка интерфейса связи;

2 – «дерево» каталогов – содержит папки с проектами, командами, операциями и т. д.;

3 – кнопки: Upload – считывание данных из контроллера, Download – загрузка программы в контроллер;

4 – базовые элементы языка LAD;

5 – кнопки: Run – запуск контроллера на выполнение программы, Stop – остановка работы контроллера;

6 – кнопка: Program Status – наблюдение за переменными в режиме терминала;

7 – окно редактора программ SIMATIC LAD. Окно содержит вкладку “Main” – основная программа, “Sbr0” – подпрограмма (может отсутствовать), “Int0” – обработчик прерывания (может отсутствовать).

Для установления соединения с контроллером и загрузки программы необходимо произвести настройку канала связи RS-232 – RS-485, а также настроить среду для работы с конкретным типом CPU. В системах автоматизации Siemens любое соединение интеллектуальных устройств производится посредством сетевой инфраструктуры, при этом, сетевыми устройствами могут быть несколько CPU, панели визуализации, ПК. В случае соединения ПК и CPU образуется локальная сеть, состоящая из Master-устройства и Slave-устройства. Так как любые сетевые устройства должны иметь разные адреса, то при установке Step 7 MicroWin32 компьютеру присваивается сетевой адрес, равный нулю. Адрес можно изменить через панель управления Windows в свойствах компонента “PG/PC Interface”. Заводские установки CPU назначают адрес контроллера, равный 2. Таким образом, при первом подключении обеспечивается корректное бесконфликтное соединение. Если физические соединения выполнены правильно и к CPU подключено питание, необходимо выбрать пиктограмму «Communications...» на вкладке «View» в левой части приложения STEP7 MicroWin32. Откроется окно, представленное на рис. 6.5. Двойной щелчок по значку обновлений в диалоговом окне. STEP7 MicroWin32 проверит наличие

каких-либо устройств подключенных к программатору (ПК). Если к ПК подключен CPU, ниже появится его значок. Необходимо дважды щелкнуть по значку станции CPU, с которой вы хотите обмениваться данными (если в сети несколько станций). Нужно обратить внимание на то, что коммуникационные параметры в окне «Communication» (Установление связи) отображают параметры выбранной станции (рис. 6.5). Виды параметров: «Local» (Локальный адрес), «Remote» (Удаленный адрес), «PLC Type» (Тип контроллера), «Interface» (Способ соединения), «Protocol» (Протокол), «Mode» (Режим), «Baud Rate» (Скорость передачи).

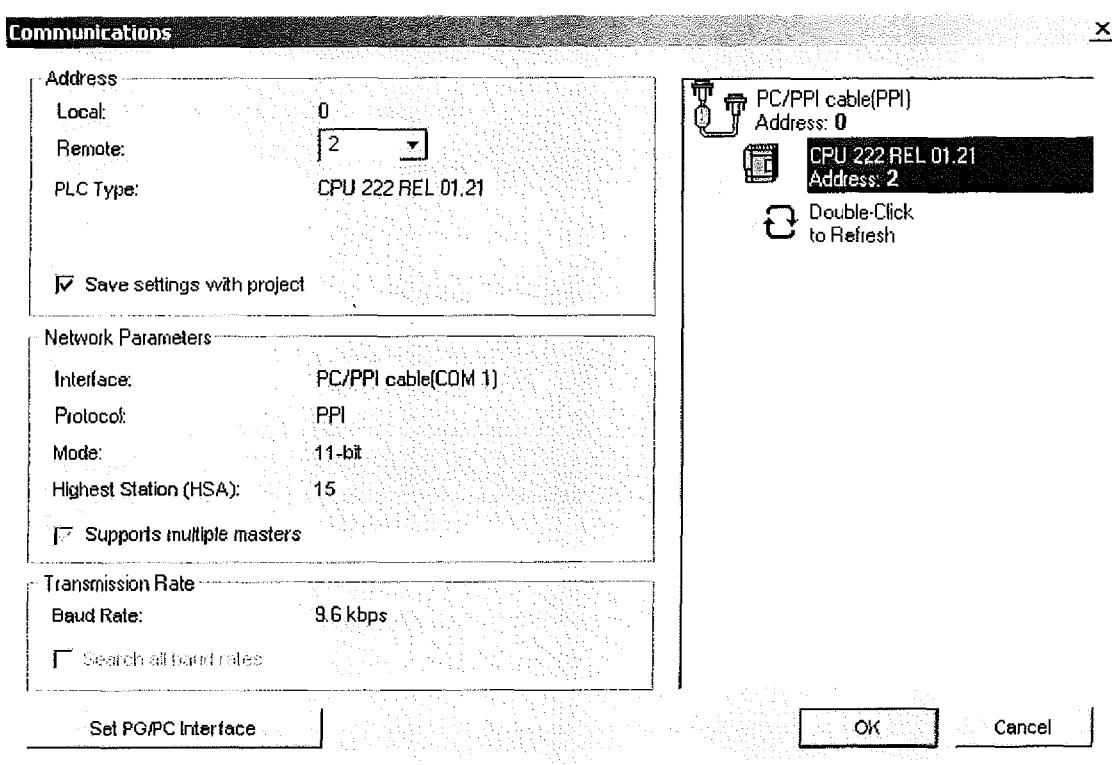


Рис. 6.5. Настройка связи с S7-200

Если соединение прошло успешно, как показано на рис. 6.5, то при нажатии на кнопку «OK» среда программирования автоматически произведет настройку на выбранный CPU. Настройку можно произвести и вручную, через команду меню *View\CPUType*. Данная команда используется в тех случаях, когда пользователь разрабатывает программу без подключенного CPU (в режиме «Offline») или при наличии симулятора других производителей.

### 6.3. Создание простой программы ПЛК и ее отладка

Если соединение с ПЛК успешно установлено, можно переходить к созданию программы, ее загрузки в ПЛК и наблюдению за ее работой.

Один из наиболее простых примеров программы для ПЛК – выполнение заданной логической функции. Например, требуется реализовать следующую функцию: если на заданный вход ПЛК подается высокий уровень напряжения 24 В, необходимо замкнуть два реле на выходе ПЛК. Если на другой заданный вход ПЛК подается высокий уровень напряжения, необходимо замкнуть другое реле. При отсутствии напряжения высокого уровня на этих входах оба реле необходимо разомкнуть.

Очевидно, что данная задача представляет собой 3 логических функции с двумя входными переменными  $X_1$  и  $X_2$  и тремя выходными переменными  $Y_1$ ,  $Y_2$ ,  $Y_3$ . Аналитические выражения для этих функций записывается следующим образом:  $Y_1=X_1$ ,  $Y_2=X_1 \cdot X_2$ ,  $Y_3=X_2$ . Так как ПЛК оперирует с физическими сигналами, подачу напряжений и индикацию  $Y_1$ ,  $Y_2$  и  $Y_3$  можно осуществить с помощью электрической схемы подключения, приведенной на рис. 6.6.

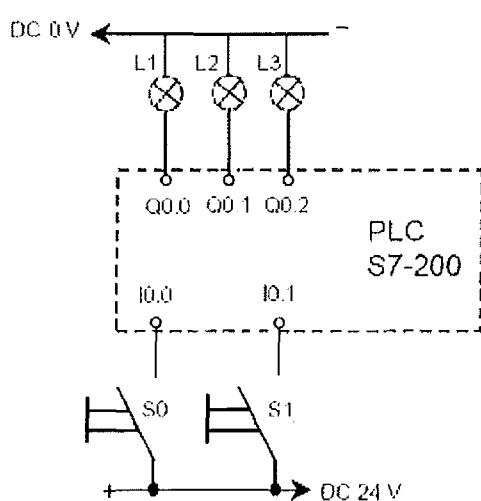


Рис. 6.6. Схема соединения

Следует отметить, что приведенная схема для решения задачи является частным случаем и выбирается в каждой конкретной ситуации в зависимости от типа CPU. В данном случае входы CPU рассчитаны на подключение к контактным датчикам. Замкнутый контакт S воспринимается аппаратурой ПЛК как логическая «1», разомкнутый – как логический «0». Аналогичная ситуация описывает поведение выходов ПЛК. В данном случае логическая «1» – это появление на выходе напряжения 24 В. Как правило, каждый ПЛК имеет встроенную светодиодную индикацию появления на входе и выходе логической «1» и «0». При заданном типе подключения логическая функция записывается следующим образом: Q0.0=Q0.1=I0.0, Q0.2=I0.2, где Q0.0-Q0.2 – адреса памяти дискретных выходов, I0.0-I0.1 – адреса памяти дискретных входов.

Очевидно, что программа пользователя должна оперировать с приведенными переменными. Для логических операций в S7-200 используется специальный язык лестничной логики LAD (Ladder Logic) или KOP (контактный план). Появление данного языка было связано с использованием релейной логики в системах управления, где подобные функции реализовывались с помощью электрических схем. При этом результат логической операции считался истинным, если на выходе блока или элемента присутствовало напряжение. В табл. 6.2 приведены основные обозначения булевых переменных и простейших логических операций для языка LAD.

Например, вход I0.0 с подключенным контактом S соответствует наглядному обозначению цепи с разрывом. Если контакт разомкнут, ток через цепь не течет и результат логической операции (РЛО) равен «0». В противном случае РЛО=1. В связи с этим становится понятна логика простейших логических операций «И» и «ИЛИ». Если два и более контактов включены последовательно, то РЛО=1 только при условии замыкания всех контактов в цепи. Это соответствует логике «И» (многовходовый логический

элемент «И») В случае использования элемента «ИЛИ» РЛО=1 при условии, что будет замкнут хотя бы один контакт из параллельно подключенных.

Таблица 6.2

Булевые переменные в языке LAD

Контактор	Инструкция для PLC в соответствии с функцией
	Опрос: Ток течет? Если да, тогда результат опроса истина. (Результат равен «1»)
	Опрос: Ток не течет? Если да (нет тока), тогда результат опроса истина. (Результат равен «0»)
	Индикатор: Если значение «истина» и ток проходит через индикатор, то он загорается
	Последовательная цепочка: (Логика «И»). Первый ключ И второй ключ должны быть замкнуты для протекания тока
	Параллельная цепочка: (Логика «ИЛИ»). Первый ключ ИЛИ второй ключ должны быть замкнуты для протекания тока
	Этот элемент управляющей программы обозначает конец главной программы

Подобным образом имитируется операция инверсии входной переменной. Если контакт замкнут, это соответствует логическому «0» на входе (оператор не нажал на ключ), но при этом РЛО=1. При нажатии на ключ (на входе «1») цепь размыкается и РЛО=0. Данная операция применима только к входной величине. Если требуется инвертировать РЛО какой-либо другой операции (например, «ИЛИ»), необходимо применить специальную операцию «- | NOT |-».

Используя такие программные элементы и связи между ними, можно создавать достаточно сложные логические функции. Результат логической функции в целом необходимо передать на выход контроллера. Для этого

используется элемент «обмотка реле», на вход которой подается РЛО предыдущего элемента.

Для решения приведенной выше задачи программа будет выглядеть так, как показано на рис. 6.7.

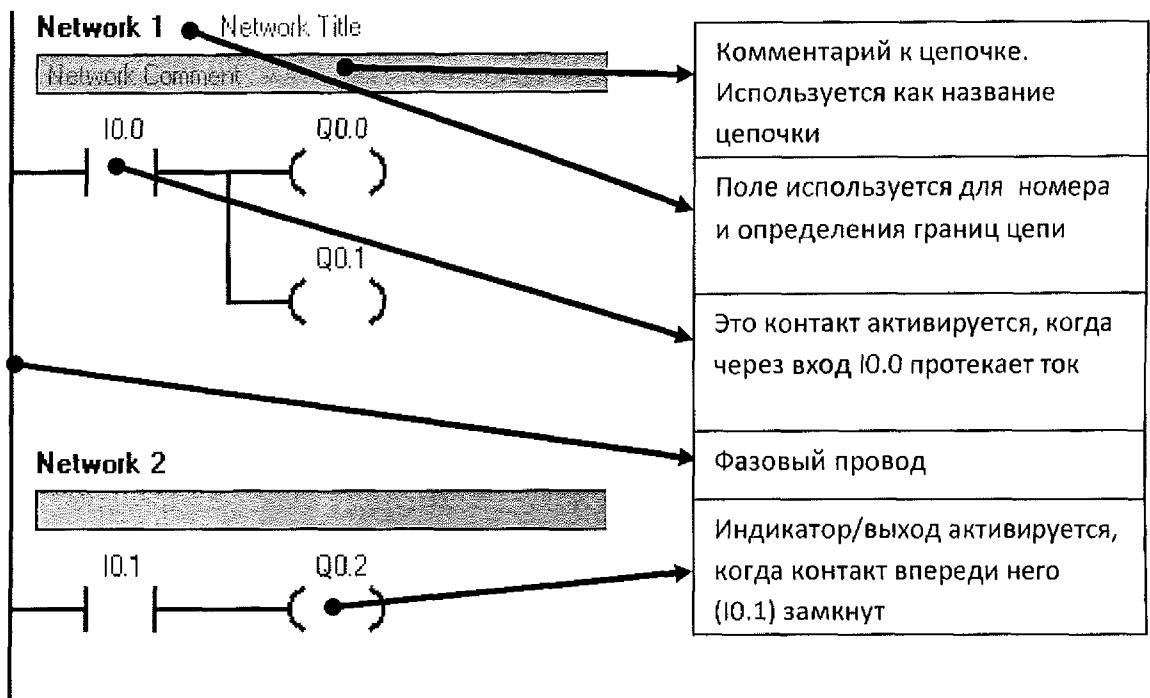


Рис. 6.7. Внешний вид программы на языке LAD

Так как в реальной ситуации используются заданные пользователем входы и выходы, каждый программный элемент имеет свой адрес. Для удобства представления программа разбивается на элементы, называемые цепями (*Network*). Назначение цепей – структурировать программу для более наглядного представления и ввода комментариев. В данном случае протекание тока от «фазного провода» через контакт I0.0 будет вызывать замыкание реле Q0.0 и Q0.1. В отдельной цепи реализована логическая функция  $Q0.2=I0.1$ .

Ввод программы в среде Step7 производится в следующей последовательности:

1. Перейдите на вкладку окна «Main».

2. Щелкните в поле Network 1 на линию со стрелкой. Стрелка должна выделиться рамкой. Стрелка показывает направление выстраиваемой нами цепочки из элементов.
3. Далее нажмите на кнопку «--| |» в панели инструментов и в появившемся контекстном меню выберите нормально открытый контакт вида «--| |», или раскройте вкладку «Bit Logic» в окне с вкладками (рис. 6.4), найдите тот же элемент и двойным щелчком мыши либо перетаскиванием вставьте его в рамку.
4. Добавив контакт в поле Network 1, присвойте ему имя I0.0, которое располагается над элементом. Изменить имя можно, выделив его двойным щелчком мыши.
5. Выделите рамкой стрелку, расположенную за созданным нами элементом. Нажмите на кнопку «--( )» в панели инструментов и в появившемся контекстном меню выберите контакт вида «--( )», или, раскройте вкладку «Bit logic» в окне с вкладками, найдите тот же элемент и двойным щелчком мыши либо перетаскиванием вставьте его в рамку. Заметьте, что при появлении данного элемента исчезла стрелка, а это означает, что наша выстраиваемая цепочка закончилась.
6. Присвойте контакту имя Q0.0.
7. Далее выделите рамкой предыдущий элемент «--| |». Затем нажмите на кнопку со стрелкой вниз ↓ в панели инструментов. При этом в окне редактора в поле Network 1 появится стрелка, идущая от линии между элементами I0.0 и Q0.0 вниз и вправо.
8. Выделите стрелку рамкой. Затем нажмите на кнопку «--( )» в панели инструментов и в появившемся контекстном меню выберите контакт вида «--( )». Присвойте элементу имя Q0.1.
9. Перейдите в поле Network 2 и выделите стрелку рамкой.
10. Нажмите на кнопку «--| |» в панели инструментов, и в появившемся контекстном меню выберите нормально открытый контакт вида «--| |». Присвойте ему имя I0.1.

11. Выделите стрелку, расположенную в этом же поле, рамкой. Затем нажмите на кнопку «--( )» в панели инструментов и в появившемся контекстном меню выберите контакт вида «--( )». Присвойте элементу имя Q0.2.

После ввода программы ее необходимо проверить на наличие ошибок. Проверка осуществляется командой «*Compile All*». Следует отметить, что проверка программ на наличие простых ошибок осуществляется на этапе ввода – например, неправильный формат адреса контакта выделяется красным. Если ошибок не обнаружено, необходимо загрузить программу в ПЛК, нажав на кнопку  . Откроется окно, показанное на рис. 6.8.

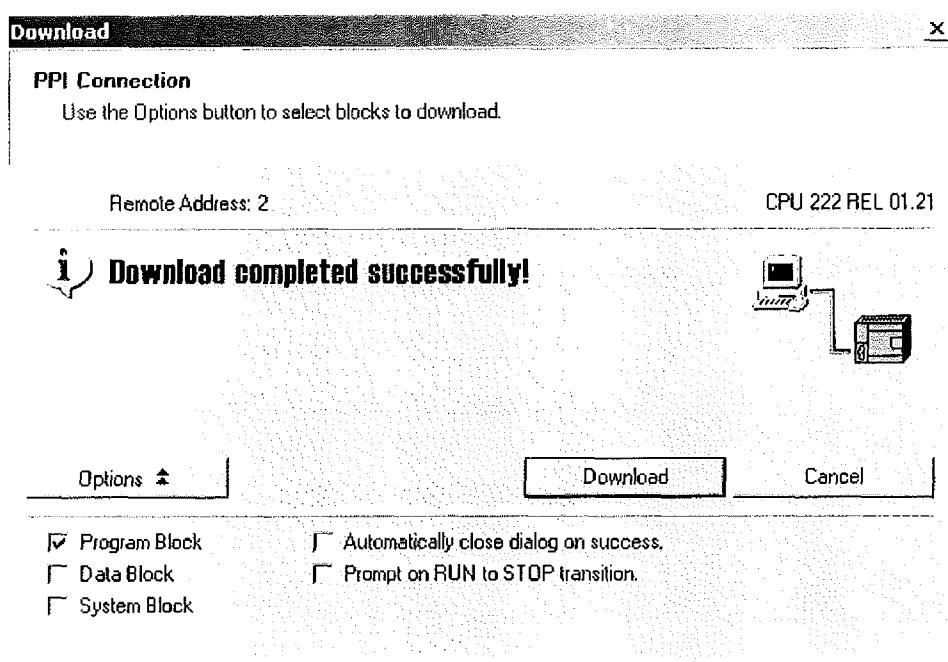


Рис.6. 8. Окно загрузки программы в ПЛК

Перед выполнением загрузки необходимо перевести переключатель режимов работы на лицевой панели ПЛК в режим «Term».

Так как среда программирования уже предварительно настроена на данный тип ПЛК, необходимо выделить, какие именно блоки требуется загрузить. Так как в написанной программе не используются блоки данных и системный блок не изменился пользователем, отмечается только программный блок. После успешной загрузки щелкните мышью на кнопку

¶ для переключения контроллера в режим RUN (загорится индикатор RUN). Загруженная в ПЛК программа начнет работать. При этом операционная система ПЛК циклически выполняет следующие действия:

1. Опрашивает состояние переключателей и заносит их состояние в память I.
2. Выполняет последовательно цепи Network 1 и Network 2, формируя в памяти Q результаты логических операций.
3. Проводит самодиагностику CPU.
4. Выполняет коммуникационные функции (в данном примере - отсутствуют).
5. Обновляет состояние выходов ПЛК путем передачи данных из памяти Q на физические выводы, сохраняя эти значения до следующего обновления.

Таким образом, длительность цикла обновления выходов зависит от количества команд программы. Исполнение программы в ПЛК можно сравнить с системами моделирования Simulink, MicroCap, Multisim, с тем отличием, что выполнение программы производится в реальном (а не модельном) времени, а также тем, что входные и выходные переменные «привязаны» к реальным физическим входам и выходам. Наиболее близким аналогом ПЛК в данном случае является ПК с установленным программным обеспечением LabView и соответствующими модулями ввода-вывода.

В режиме «TERM» возможна отладка программы и наблюдение за текущими значениями переменных. Для выполнения отладки необходимо запустить программу и выполнить команду *Debug/Programm Status*. Пример отображения статуса программы показан на рис. 6.9. В данном режиме все цепи, РЛО которых истинно, выделены синим цветом, сигнализируя прохождение тока в цепях. На рис. 6.9 показан пример, когда замкнут выключатель S2, при этом на дискретный вход подается напряжение высокого уровня, что вызывает прохождение тока в обмотку реле, подключенного к Q0.2.

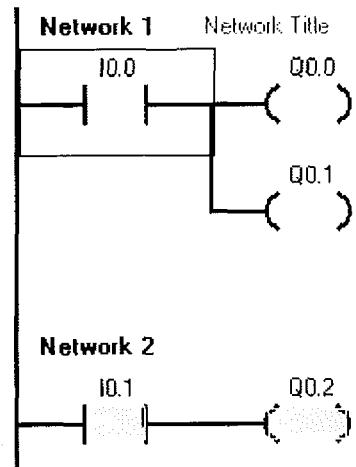


Рис. 6.9. Отладочный режим ПЛК

Следует помнить, что отображение состояний входов и выходов производится не в реальном времени, а с некоторой задержкой. Эта задержка обусловлена конечной скоростью передачи данных по каналу PPI (RS 232), поэтому такой режим можно использовать только для проверки статичных состояний, либо очень медленно изменяющихся величин.

Для более наглядного и удобного представления программы, а также для упрощения программирования, в S7-200 используются символьные таблицы. Назначение таблицы – установить соответствие между функциональным назначением дискретного входа и адресом этого входа. Например, если известно, что кнопка S1 подключена к входу I0.0, гораздо удобнее оперировать с именем входа, а не с его адресом. Данный прием полезен также в тех случаях, когда по тем или иным причинам требуется изменить адрес кнопки. В этом случае требуется лишь изменить соответствие имени и адреса в таблице, в противном случае потребовалось бы изменять адреса входов везде, где они используются в программе.

Выбор символьной таблицы производится через команду «Symbol» Table на панели навигации. Заполнение таблицы для описанного выше примера показано на рис. 6.10.

	Symbol	Address	Comment
1	Кнопка1	I0.0	Кнопка для активации 1-го индикатора
2	Кнопка2	I0.1	
3	Индикатор1	Q0.0	
4	Индикатор2	Q0.1	
5	Индикатор3	Q0.2	
6			

Рис. 6.10. Пример заполнения символьной таблицы

Если таблица заполнена корректно, редактор LAD немедленно заменяет фактические адреса в программе на их символьные представления. На рис. 6.11 показан вид программы после заполнения таблицы.

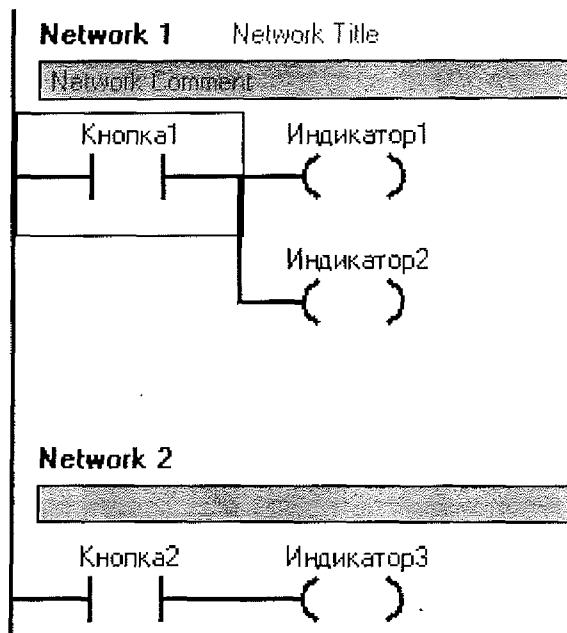


Рис. 6.11. Вид программы с символьными именами

При необходимости в настройках редактора (меню «Вид») можно включить отображение как адреса, так и символьного имени. В этом случае на экране будет показан физический адрес входа или выхода и его символьное представление из созданной ранее таблицы.

*Примечание. Следует иметь в виду, что символьная таблица не загружается в ПЛК вместе с программой. При утрате файла проекта все символьные обозначения будут потеряны.*

Рекомендуемый порядок создания проекта:

1. Реализовать на языке LAD логическую функцию, согласно варианту задания. Необходимо использовать исходную или минимизированную функцию.
2. Включить питание контроллеров на рабочих местах 6-8. На рабочем месте № 6 используется контроллер S7-200 CPU 226 XM, Ра рабочих местах № 7-8 – CPU 222 Relay.
3. Перевести ПЛК в режим «STOP» и «TERM».
4. Запустить среду Step7 MicroWin 32.
5. Произвести настройку ПК для работы с контроллером на рабочем месте.
6. Ввести программу, заполнить символьную таблицу, обозначив входные переменные символами «X1-X4» а выходные – символами «Y1» и «Y2». В качестве имитаторов входных переменных используются тумблеры, подключенные к входам I0.0 – I0.3.
7. Произвести компиляцию программы и загрузку ее в ПЛК.
8. Запустить программу на выполнение. Изменяя положение переключателей, наблюдать за поведением выхода Y (светодиодная индикация), контролируя, таким образом, таблицы истинности реализованных функций.
9. Включить режим «Programm Status». В реальном времени проследить выполнение программы. При возникновении ошибок, локализовать их, используя визуальное отображение РЛО участков цепей.

## 7. ДОСТУП К ДАННЫМ S7-200

### 7.1. Типы данных и способы их размещения в памяти

Программируемый логический контроллер семейства S7-200 поддерживает распространенные типы данных, принятые в вычислительной технике. Кроме этого, в архитектуре S7-200 заложен принцип доступа, именуемый «битовые поля», принятый, в основном, в однокристальных микроконтроллерах. Это обусловлено спецификой применения ПЛК в производстве. Так, например, очень часто в одной программе требуется обрабатывать как данные с плавающей точкой (для вычислений и масштабирования аналоговых величин – температуры, давления и т.д.), так и управлять объектами на уровне битовых величин (включение и выключение механизмов, двигателей и т.д.). Язык LAD, принятый основным для данного семейства ПЛК, поддерживает все типы данных, обеспечивая не только функциональность, характерную для большинства языков программирования высокого уровня, но и наглядность представления программы. В табл. 7.1 приведены основные типы данных S7-200 с указанием их размера (за исключением битовых).

Таблица 7.1

Размеры и типы данных S7-200

Представление	Байт (B)	Слово (W)	Двойное слово (D)
Целое без знака	От 0 до 255 От 0 до FF	От 0 до 65 535 От 0 до FFFF	От 0 до 4 294 967 295 От 0 до FFFF FFFF
Целое со знаком	От -128 до +127 От 80 до 7F	От -32 768 до +32767 От 8000 до 7FFF	От -2 147 483 648 до +2 147 483 647 От 8000 0000 до 7FFF FFFF
Вещественное IEEE 32-битовое с плавающей точкой	Неприменимо	Неприменимо	От +1.175495E-38 до +3.402823E+38 (положительное) От -1.175495E-38 до -3.402823E+38 (отрицательное)

В принятой для S7-200 системе обозначений размер байта (8 бит) обозначается буквой B, слова (16 бит) – буквой W, двойного слова (32 бит) – буквой D. Необходимо обратить внимание, что данные обозначения распространяются на все типы данных – как целые, так и вещественные. Например, для адресации вещественного числа с плавающей точкой необходимо использовать обозначение D, так как размер данного числа в памяти составляет 32 бит (64-битные значения удвоенной точности, принятые в C++ в ПЛК не используются).

Для того чтобы адресовать какое-либо значение в памяти ПЛК, необходимо указать идентификатор области памяти (I, Q, M, V, AC, SM и т.д.), размер данных (B, W, D) и адрес начального байта этого значения. Любая память ПЛК S7-200 (кроме недоступной для адресации программной памяти), организована как байтовый массив ячеек, поэтому адресом любого типа и размера данных является первый байт этого числа. На рис. 7.1 показан принцип адресации различных размеров данных.

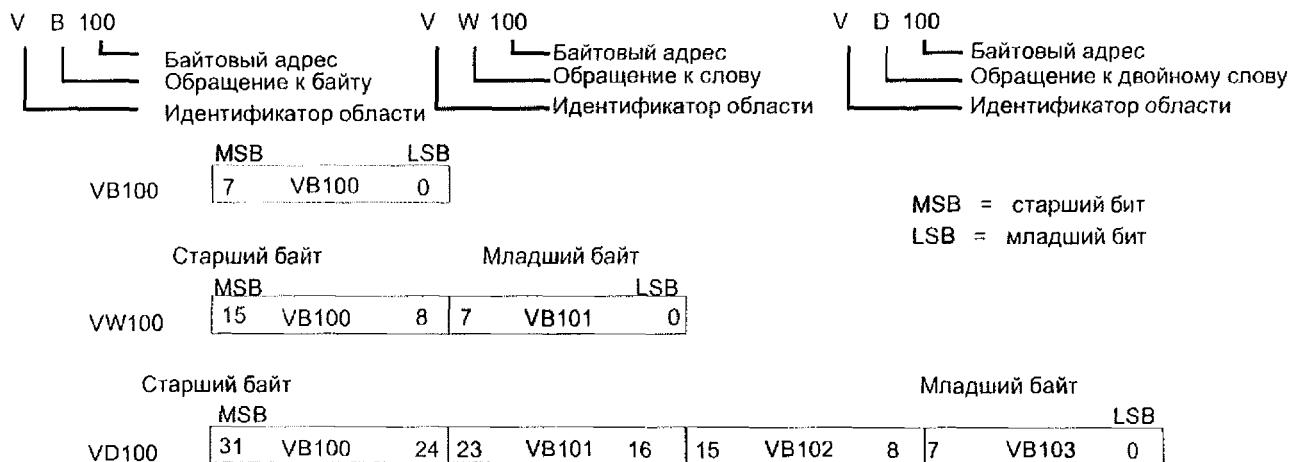


Рис. 7.1. Принцип адресации данных S7-200

В приведенном примере показан фрагмент памяти данных V с адреса 100 по 103. Для того чтобы обратиться к байту по адресу 100, необходимо применить в команде обращения текстовое выражение «VB100» (память V, байт, 100-я ячейка). Если указать синтаксис «VW100», ПЛК прочитает из

памяти слово, состоящее из двух байт по адресам 100 и 101. При обращении к двойному слову VW100 будут использованы 4 байта с адресами 100, 101, 102, 103.

Следует отметить, что в данном примере наглядно иллюстрируется отличие в размещении данных в памяти для S7-200 от принятой фирмой INTEL нотации расположения байт в многобайтных словах для любых носителей данных. Для INTEL-представления по младшему адресу расположен младший байт числа (LSB), по старшему адресу – старший (MSB). Заметно, что для S7-200, как и для других контроллеров SIEMENS, порядок расположения байт обратный.

Пользователю, работающему с большим числом многобайтных слов, часто приходится решать задачу последовательного размещения данных в памяти. S7-200 поддерживает косвенную адресацию данных, однако для большинства задач память V (или M) необходима для временного сохранения промежуточных результатов, к которым может осуществляться прямой доступ из систем верхнего диспетчерского уровня (например, из панелей оператора или SCADA-систем). Очевидно, что зная тип данных и их размер, можно легко вычислить очередной адрес ячейки, где будет храниться следующий фрагмент данных. Например, если требуется сохранить 2 двойных слова, начиная с ячейки 100, то первое слово будет иметь адреса 100-103, а следующее – 104-107. Если же следом за первым двойным словом размещено 16-битное слово, его адрес будет равен 104-105 (при обращении указывается VW104). *Следует отметить, что операционная система ПЛК не отслеживает возможные ошибки и наложения данных, поэтому данная процедура размещения целиком возлагается на пользователя.*

Отдельный способ адресации принят для битовых данных. Так как бит является наименьшей неделимой единицей информации в ПЛК, целесообразнее было бы организовать память как битовый массив. Однако такой способ адресации привел бы к неоправданному усложнению синтаксиса при указании

адреса для многобайтных слов и сложности вычисления последнего. Поэтому, для адресации бит используется другой принцип, показанный на рис. 7.2.

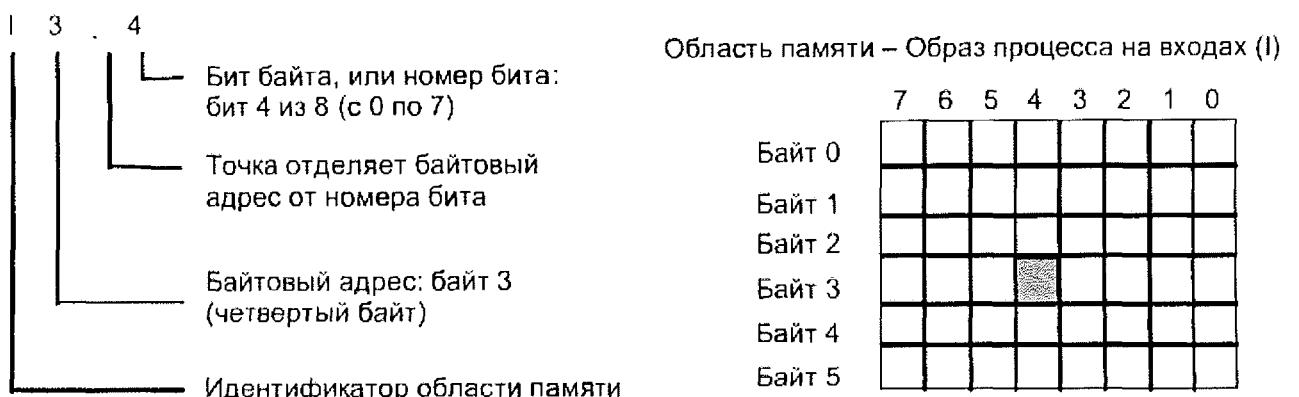


Рис. 7.2. Принцип адресации отдельных бит S7-200

Данный принцип получил название «битовое поле». В синтаксисе команды обращения указывается идентификатор области (в примере I – область данных дискретных входов), номер байта в данной памяти, а затем через точку – номер бита в этом байте, начиная с нулевого. Следует отметить, что при обращении к биту определенного байта идентификатор самого байта (B) не используется. Это связано с тем, что адресация бит в многобайтных словах не применяется. Подобным образом можно обращаться к конкретным битам многобайтных слов, необходимо лишь вычислить адрес байта и адрес бита в этом слове. Например, для слова VD100 (рис. 7.1) необходимо прочитать или установить 19-й бит этого слова. Очевидно, что данный бит размещен в 3-м байте этого слова, начиная с младшего. Следовательно, адрес бита - V101.3.

Адресация отдельных бит наглядно проиллюстрирована в [2] при рассмотрении примера простой программы.

Кроме основных правил доступа к данным, имеются некоторые особенности, связанные со спецификой архитектуры S7-200:

- Как правило, все виды памяти доступны как для записи, так и для чтения, за исключением некоторых отдельных адресов. Например,

биты специальной памяти SM0.0 и SM0.1 представляют собой константы, доступные только для чтения. То же самое можно сказать о некоторых байтах и словах в памяти SM.

- Если в программе осуществляется доступ к логически существующей ячейке, но отсутствующей физически, то при чтении возвращается нулевое значение, при записи – информация не записывается. Такая ситуация может возникнуть тогда, когда производится обращение к входам и выходам, для которых не назначен соответствующий модуль расширения.
- На этапе ввода данных среда Step7 отслеживает некорректные адреса данных в том случае, когда допущена ошибка в синтаксисе адреса, а также при несуществующем логическом адресе (выход за пределы пространства памяти для данного ПЛК). Ошибка индицируется также в том случае, когда размер данных не соответствует типу команды (например, попытка переслать байт данных командой пересылки слова).

Следует также отдельно сказать об адресации строковых данных и формате вещественных чисел. Обычно, в языках программирования (например, в C++) строка не выделяется в отдельный тип данных, а организована как одномерный байтовый массив. При этом, для корректного выполнения команд работы со строками используется нулевой байт как признак конца строки. Размер строки может быть не задан фиксировано, а сама строка описана в виде указателя (например, *char \*Str*). В ПЛК S7-200 строка является отчасти независимым типом данных, так как размер элемента строки всегда представлен байтом, а структура строки выглядит так, как показано на рис 7.3.

Длина	Символ 1	Символ 2	Символ 3	Символ 4	...	Символ 254
Байт 0	Байт 1	Байт 2	Байт 3	Байт 4		Байт 254

Рис. 7.3. Формат строки для S7-200

Длина строки не может превышать 254 байта, первым байтом строки является указанное пользователем число элементов строки. Для корректного использования данного формата в командах обработки строк необходимо позаботиться о том, чтобы значения каждого элемента строки были представлены в стандартном ASCII-коде (за исключением первого байта, представленном в двоичном виде). Строковые типы данных используются в S7-200 для ввода-вывода информации через коммуникационный порт RS-485 при подключении ПЛК к различному внешнему оборудованию (например, к считывателю штрих-кода).

Как показано в табл. 7.1, вещественные числа в S7-200 имеют размер двойного слова. На рис. 7.4 показан формат вещественного числа.

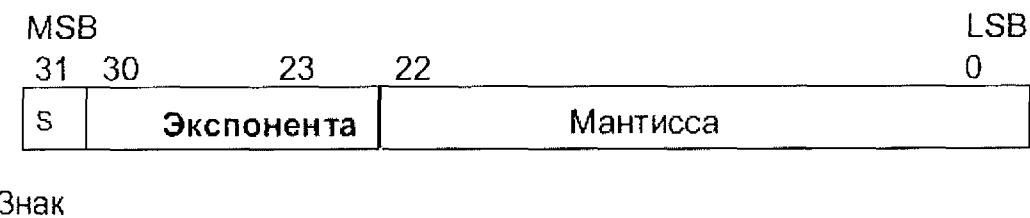


Рис. 7.4. Формат вещественного числа S7-200

Во многих случаях, при написании программы пользователю необходимо задавать константы непосредственно в тексте. Входной аргумент команды, представленный константой, означает, что данное число будет размещено не в памяти ОЗУ (и сохранено в ПЗУ, если данный адрес попадает в область реманентной памяти), а непосредственно в программной памяти ПЛК. Это соответствует описанию типа Const Char a=10, где ключевое слово *const* размещает число 10 в памяти программы и передает пользователю указатель на него в виде переменной *a*. В табл. 7.2 приведены основные правила записи констант для языка LAD.

Данные правила действуют и в том случае, когда пользователь задает блок констант в памяти V.

Таблица 7.2

## Правила записи констант в LAD

Представление	Формат	Пример
Десятичное	[десятичное значение]	20047
Шестнадцатеричное	16#[шестнадцатеричное значение]	16#4E4F
Двоичное	2#[двоичное число]	2#1010_0101_1010_0101
ASCII	'[текст ASCII]'	'A'
Вещественное	ANSI/IEEE 754-1985	+1.175495E-38 (положительное) -1.175495E-38 (отрицательное)
Строка	«[текст строки]»	«ABCD»

**7.2. Структура программы обработки данных на языке LAD**

На рис. 7.5 приведен пример простой программы на языке LAD. Программа выполняет следующие действия: если I0.0=1, производится сложение содержимого ячеек VW0 и VW10 с помещением содержимого в VW20. Если сложение выполнено успешно, то производится вычитание содержимого ячейки VW30 из результата прошлой операции и занесение результата в VW40.

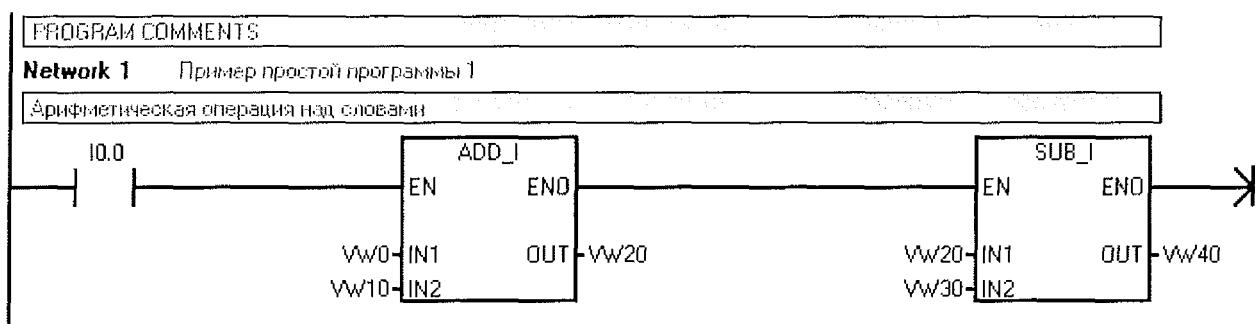


Рис. 7.5. Пример программы на языке LAD

Следует отметить, что логика обработки данных несколько отличается от принятой в других языках программирования. Так, операция сложения является составной и выполняется по условию  $I0.0=1$ . Практически любая операция с данными на языке LAD имеет условие выполнения (*EN*) и условие подтверждения выполнения (*ENO*). Другими словами, операция выполняется, если  $EN=1$ , при этом данные на входе *IN1* и *IN2*, обрабатываются согласно заданной функции с размещением результата в *OUT*. Если операция выполнилась успешно, *ENO* присваивается значение 1 (true). Значение *ENO* можно использовать в тех случаях, когда целесообразно прервать выполнение цепочки операций при возникновении ошибки в одной из них. Возможен и другой вариант этой же программы, показанный на рис. 7.6.

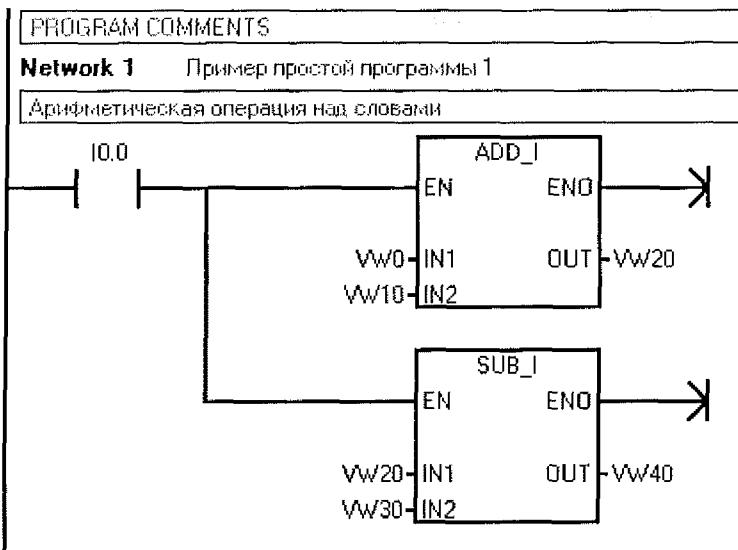


Рис. 7.6. Вариант расположения блоков программы

Логика работы данной программы отличается от предыдущей тем, что функция вычитания будет выполнена в любой случае, независимо от результата сложения. Однако обе эти функции будут выполняться в том случае, если  $I0.0=1$ .

Программы, показанные на рис. 7.5 и 7.6, выполняются циклически, под управлением ОС контроллера. Это означает, что в каждом цикле операционной системы будет проводиться проверка  $I0.0$ , и при его истинном значении будут выполняться функции сложения и вычитания. Так как цикл ОС выполняется

достаточно быстро, то обновления ячеек VW20 и VW40 будут производиться в реальном времени.

Для всех программ на языке LAD выполняется условие – последовательность выполнения команд подчиняется правилу «слева-направо и сверху-вниз». Важно отметить, что такая логика несколько отличается от других систем моделирования (например, MATLAB, LABVIEW), где принято выполнение блоков в зависимости от готовности данных. Аналогом выполнения команд на рис. 7.5 и 7.6 можно считать форму записи на языке C++:

```
if ( I0.0 == true )  
{  
    VW20=VW0+VW10;  
    VW40=VW20-VW30;  
}
```

В приведенном варианте программы используются исходные данные, размещенные в памяти V. Для выполнения данного примера необходимо предварительно занести в ячейки VW0, VW10, VW30 константы. Данная операция может быть выполнена с помощью инициализации DATA-блока ПЛК (вкладка на панели навигации **View\**). Блок данных можно заполнить следующим образом (рис.7.7).

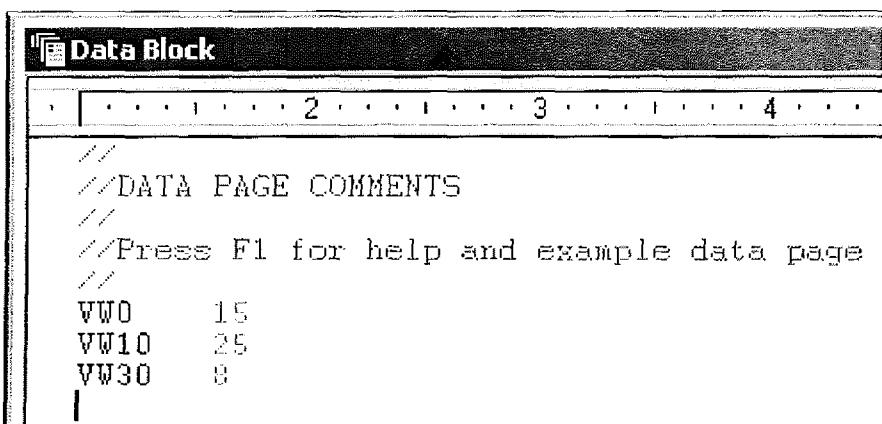


Рис.7.7. Пример заполнения DATA-блока

В дальнейшем при сохранении проекта DATA-блок будет сохранен вместе с программой. При загрузке проекта в контроллер необходимо отметить загрузку DATA-блока. При этом блок будет записан в релейную память V в ПЗУ ПЛК. При включении питания данные из релейной области будут загружены в ОЗУ (память V) и будут доступны из программы.

Если данные неизменны в течение всей работы программы, их можно записать в виде констант непосредственно в программе, сэкономив память V.

Ячейки VW20 и VW40 используются для хранения результатов операций. Так как их значение обновляется в течение работы программы, начальная их инициализация необязательна. Так как ОС контроллера не отслеживает наложение данных, пользователю часто необходимо представлять карту распределения памяти, доступную по вкладке **Cross Reference\Byte Usage**. Карта памяти для приведенного примера показана на рис. 7.8.

Cross Reference											
Byte	9	8	7	6	5	4	3	2	1	0	
V80											W W
V810											W W
V820											W W
V830											W W
V840											W W

Рис. 7.8. Карта распределения памяти V

Из рис. 7.8 видно, что из всей памяти V используются 5 ячеек, каждая из которых имеет стартовый адрес, кратный 10, и представлена 2-байтным словом (W). Очевидно, что для компактного размещения слов в памяти их можно разместить по адресам VW0, VW2, VW4, VW6, VW8.

Для отладки выполнения программы можно воспользоваться главным окном Step7 в режиме **Programm Status** (переключатель на лицевой панели ПЛК должен быть в положении «ТЕРМ»). В этом режиме можно увидеть значения переменных на входах и выходах блоков команд с частотой обновления экрана (зависит от скорости канала связи с ПК). Для программ с большим количеством переменных можно воспользоваться вкладкой **Status**

Chart, где в колонке адреса можно указать необходимые переменные, а в колонке представления – систему счисления. Например, на рис. 7.9 показан вид окна **Status Chart** для приведенного выше примера. Отображаются результаты операций в десятичном и двоичном виде.

	Address	Format	Current Value	New Value
1	VW20	Signed	+40	
2	VW40	Binary	200000_0000_0010_0000	
3		Signed Unsigned Hexadecimal Binary ASCII		
4				
5				

Рис. 7.9. Вид окна Status Chart

Окно Status Chart может быть использовано также в тех случаях, когда необходимо по ряду причин изменить значения переменных при отладке. Например, если переменная VW0 задана в DATA-блоке как константа, можно добавить ее в список Status Chart , задать новое значение (New Value) и выполнить команду **Force Value**. Необходимо помнить, что если попытаться изменить значение переменной, модифицируемой в данный момент программой в каждом цикле ОС, то приоритет изменения будет отдан программе.

Всю систему команд S7-200 можно условно разбить на несколько групп:

1. Битовые команды. В данную группу входят команды опроса бит, установки и сброса бит по условию, установки и сброса с памятью (RS-триггер). Кроме того, в этот список входят команды выделения переднего и заднего фронта битовых событий.
2. Команды доступа к данным и их преобразования, работа с таблицами и строками.
3. Команды арифметических и логических операций над данными, команды сдвига. Арифметические команды разделяются на целочисленные команды и команды с вещественными числами.

4. Коммуникационные команды, предназначенные для работы с встроенным портом RS-485.
5. Команды работы с таймерами, счетчиками и часами реального времени;
6. Команды работы с прерываниями.
7. Команды сравнения.
8. Команды управления ходом программы (циклы, переходы по метке).

Следует отметить, что не все группы команд, а также отдельные функции выполняются на выбранном типе ПЛК. В этом случае неподдерживаемые команды выделяются в списке команд.

### **7.3. Команды преобразования и перемещения данных**

Команды перемещения и преобразования данных можно условно разделить на следующие подгруппы:

- Команды перемещения («Move»).
- Команды преобразования типов («Convert»).
- Команды обработки строк («String»).
- Команды сдвига и вращения («Shift\Rotate»).

**Команды перемещения.** Команды перемещения данных оперируют со всеми видами памяти и предназначены для транспортировки данных из одних областей памяти в другие, учитывая размер данных (слово, двойное слово, байт).

**Команды преобразования типов.** Основные команды преобразования типов данных приведены в табл. 7.3.

Как правило, для каждой команды преобразования существует обратный вариант (например, I\_B), однако существуют исключения. Например, операция преобразования вещественного в целое должна производиться через промежуточную команду отсечения дробной части. Преобразование байта в вещественное также является составной операцией — сначала производится

выравнивание размеров данных (байт – двойное слово через промежуточное преобразование в слово), а затем – непосредственное преобразование типа.

Таблица 7.3

Команды преобразования данных

Команда	Прямая	Обратная
Байт < - > Слово со знаком	<b>B_I</b>	<b>I_B</b>
Слово со знаком < - > Двойное слово со знаком	<b>I_DI</b>	<b>DI_I</b>
Слово со знаком <- > Стока	<b>I_S</b>	<b>S_I</b>
Двойное слово со знаком - > Вещественное	<b>DI_R</b>	-
Двойное слово со знаком < - > Стока	<b>DI_S</b>	<b>S_DI</b>
Вещественное < - > Стока	<b>R_S</b>	<b>S_R</b>
Целое со знаком < - > Двоично-десятичный код	<b>I_BCD</b>	<b>BCD_I</b>
Команда округления	<b>ROUND</b>	-
Команда отсечения дробной части	<b>TRUNK</b>	-
Целое со знаком - > Массив ASCII-кодов	<b>ITA</b>	-
Двойное целое со знаком - > Массив ASCII-кодов	<b>DTA</b>	-
Вещественное - > Массив ASCII-кодов	<b>RTA</b>	-

Некоторые команды преобразования имеют особенности, обусловленные спецификой применения ПЛК. Например, команда **ITA** переводит целое число размером 2 байта в массив ASCII-кодов с дополнительными возможностями. Выходным параметром команды является адрес начального байта массива, имеется дополнительный параметр FMT, формат которого показан на рис. 7.10.



Рис. 7.10. Формат байта FMT

Кроме перевода числа в массив кодов, возможна установка разделительной точки (бит  $c=0$ ) или запятой (бит  $c=1$ ) в позиции результирующего массива, определяемой кодом NNN. Например, если

необходимо сформировать массив кодов для входного числа -12345 с установкой точки после 2-го знака (-12.345), необходимо сформировать FMT=00000011. Подобным образом работают команды **DTA** и **RTA**.

Аналогичным образом производится преобразование чисел в строку, однако, порядок расположения байт в строке иной (рис.7.3).

**Команды обработки строк.** Команды работы со строками представлены в табл. 7.4. Входным параметром каждой команды является указатель на строку. Стока может быть задана как в виде строковой константы, так и с помощью адреса первого байта, с которого она расположена в памяти (рис. 7.3).

Таблица 7.4

Команды обработки строк

Команда	Обозначение
Вычисление длины строки	<b>STR_LEN</b>
Копирование строки	<b>STR_CPY</b>
Копирование части строки из N элементов, начиная с индекса M	<b>SSTR_CPY</b>
Добавление строки (IN) к концу другой строки (OUT)	<b>STR_CAT</b>
Поиск совпадений элемента строки IN2 в строке IN1 с возвратом адреса первого совпавшего символа	<b>STR_FIND</b>
Поиск в строке IN1 символа из шаблона IN2 с возвратом индекса первого совпавшего символа	<b>CHR_FIND</b>

**Команды сдвига и вращения.** Команды сдвига и вращения представлены в табл. 7.5.

Таблица 7.5

Команды сдвига и вращения

Команда	Обозначение
Сдвиг влево байта, слова и двойного слова	<b>SHL_B, SHL_W, SHL_DW</b>
Сдвиг вправо байта, слова и двойного слова	<b>SHR_B, SHR_W, SHR_DW</b>
Циклический сдвиг влево байта, слова и двойного слова	<b>ROL_B, ROL_W, ROL_DW</b>
Циклический сдвиг влево байта, слова и двойного слова	<b>ROR_B, ROR_W, ROR_DW</b>
Сдвиг N-разрядного регистра, начиная с бита S_BIT первого байта регистра	<b>SHRB</b>

Все команды сдвига, независимо от размера данных (байт, слово и двойное слово), имеют общий алгоритм. Слово данных сдвигается в направлении, соответствующем названию команды на N бит. При этом один из битов (старший или младший соответственно) выталкивается за пределы разрядной сетки, он помещается в специальную память SM1.1. Так как сдвиг не является циклическим, то сдвигаемое слово постепенно замещается нулями. Если в результате очередного сдвига число станет равным нулю, выставляется флагок SM1.0. Пример сдвиговых операций показан на рис. 7.11.

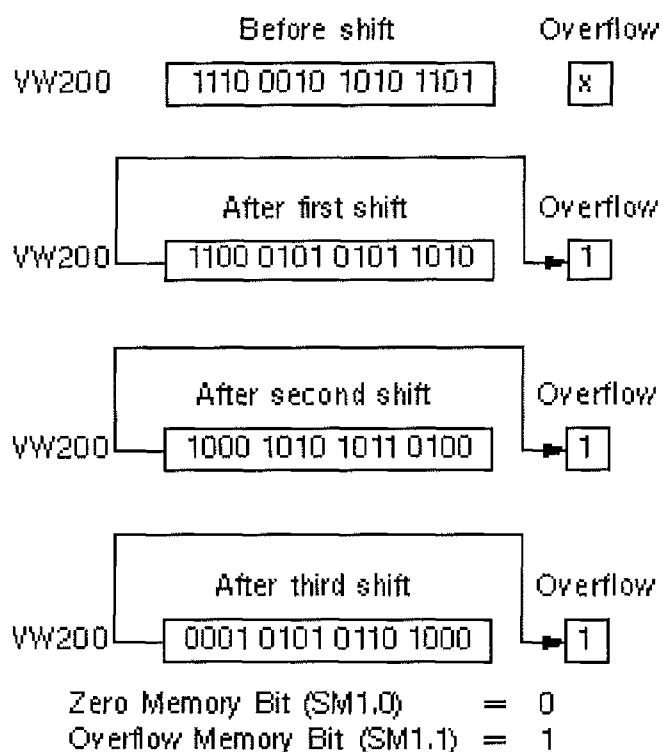


Рис. 7.11. Пример операции сдвига

В примере показан сдвиг двойного слова влево (в сторону старших бит) на 3 разряда ( $N=3$ ). После выполнения операции бит переполнения  $SM1.1=1$ , бит  $SM1.0=0$ .

Команды циклического сдвига отличаются от команд сдвига тем, что по мере выполнения операции регистр замещается не нулями, а «выталкиваемыми» за пределы разрядной сетки битами. Этим способом организуется вращение числа. Пример операции вращения показан на рис. 7.12.

Логика работы флагков SM1.1 и SM1.0 не отличается от логики в операциях сдвига.

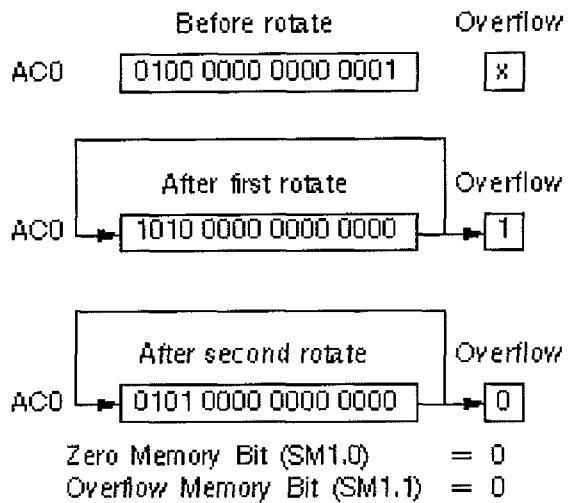


Рис. 7.12. Пример операции циклического сдвига числа

В случае если необходим сдвиг числа, представленного произвольным количеством бит (например, превышающем количество бит двойного слова), предусмотрена команда SHRB. Логика работы команды подробно описана в [2].

#### 7.4. Варианты заданий

В задании необходимо реализовать заданную по варианту в таблице 7.6 последовательность преобразований данных. Исходные данные задаются в виде констант разных типов в DATA-блоке, начиная с адреса, указанном в таблице. Результаты преобразований необходимо размещать в ячейках памяти V так, чтобы обеспечить непрерывное заполнение памяти, без пропусков адресов. Правильность преобразований данных можно отслеживать в ходе работы программы (в режиме терминала). В отчете к выполненной работе необходимо указать:

- задание по варианту;
- текст программы в режиме эмуляции с результатами;
- карту памяти с расположением исходных данных и результатов в памяти.

Таблица 7.6

## Варианты заданий

№	Тип исходных данных					Начальный адрес блока исходных данных	Начальный адрес блока результатов
	BYTE	INT	DINT	REAL	STRING		
1	INT,REAL	REAL,BCD	INT>, REAL	BYTE>, INT	INT<	VB35	VB120
2	STRING, INT	STRING, BYTE>	REAL,BCD	DINT>	INT, REAL	VB61	VB150
3	REAL	REAL,DINT>	BCD, INT<	BYTE>, STRING	DINT, REAL	VB31	VB320
4	INT>, DINT, BCD	DINT>, REAL	STRING, INT	STRING, BYTE>	DINT, REAL	VB70	VB210
5	DINT>, REAL	STRING, REAL	INT, BYTE	DINT<, BCD	INT>, BCD	VB0	VB140
6	STRING, REAL	BCD, DINT	BCD, INT>	DINT<	INT, REAL	VB20	VB170
7	INT, DINT	BCD, DINT<	STRING,INT<	BYTE,STRING	INT, REAL	VB30	VB220

## **8. АНАЛОГОВЫЕ МОДУЛИ И ИХ ПРИМЕНЕНИЕ**

### **8.1. Область применения аналоговых модулей**

Помимо дискретного ввода-вывода в системах автоматизации используются источники и приемники данных, с непрерывным способом представления информации в виде тока, напряжения и сопротивления. Примером аналоговых входных величин могут служить данные температуры, давления, уровня, расхода, массы и т. д., значения которых необходимо измерять с определенной и достаточно высокой точностью. Для выдачи управляющего воздействия на объект также может понадобиться аналоговый сигнал. Примером может служить значение напряжения, от величины которого зависит положение части какого-либо объекта управления, например, положение заслонки в трубопроводе, или скорость вращения двигателя.

С помощью цифровых входов и выходов решаются задачи другого класса, например: включение и выключение какой-либо нагрузки по команде, предусмотренной алгоритмом, опрос контактного датчика, и т. д. Таким образом, управляющее воздействие в данном случае может быть только двоичным («включено», «выключено»), как и входные величины («датчик сработал», «датчик не сработал»).

Контроллеры S7-200 относятся к микросистемам автоматизации, поэтому, как и в ПЛК LOGO!, наличие встроенных аналоговых входов и выходов для них не предусмотрено, однако в составе выносных модулей расширения они присутствуют. При этом максимальное количество входов и выходов (а значит и модулей) зависит от типа PLC. В каждом ПЛК поддержка аналоговых величин организована в виде специальных команд.

Применение модулей аналогового ввода-вывода позволяет дополнять систему ввода-вывода программируемых контроллеров S7-200 требуемым количеством и видом аналоговых каналов и обеспечивает:

- оптимальную адаптацию контроллера к требованиям решаемой задачи за счет подключения необходимого количества модулей расширения;
- возможность непосредственного подключения к входам и выходам контроллера широкой гаммы аналоговых датчиков и исполнительных устройств;
- высокую гибкость: наличие модулей расширения позволяет в любой момент времени расширить функциональные возможности контроллера с внесением соответствующих изменений в программу.

В системах автоматизации на базе S7-200 существует 5 типов аналоговых модулей, различающихся назначением и параметрами и выходных-входных сигналов:

- Модули **ЕМ 231** выполняют аналого-цифровое преобразование входных аналоговых сигналов контроллера. Результат преобразования может быть использован центральным процессором SIMATIC S7-200 в процессе выполнения программы. Модуль позволяет подключить 4 источника аналоговых сигналов по дифференциальной схеме, причем входной сигнал может быть представлен и в виде тока, и в виде напряжения. Диапазон входного сигнала выбирается с помощью DIP-переключателей на корпусе модуля.
- Модули **ЕМ 232** предназначены для цифро-аналогового преобразования внутренних числовых величин контроллера в его внешние аналоговые сигналы. Позволяют подключить 2 приемника аналогового сигнала, выходной сигнал может быть представлен в виде тока и напряжения. Диапазон входного сигнала выбирается с помощью DIP-переключателей на корпусе модуля.

- Модуль **EM 235** выполняет аналого-цифровое преобразование входных аналоговых сигналов контроллера и формирует цифровой результат измеренного значения параметра, а также цифро-аналоговое преобразование внутренних цифровых величин контроллера и формирование его выходных аналоговых сигналов. Результат аналого-цифрового преобразования может быть использован центральным процессором SIMATIC S7-200 в процессе выполнения программы. Позволяет подключить 4 дифференциальных источника аналогового сигнала и один приемник. Данные входов и выходов могут быть представлены в виде токов или напряжений. Диапазон входного сигнала выбирается с помощью DIP-переключателей на корпусе модуля.
- Модуль **EM 231 TC** позволяет производить прецизионное измерение температуры с использованием стандартных термопар, а также измерение сигналов напряжения  $\pm 80$  мВ. В отличие от модулей, перечисленных выше, обладает следующими особенностями: возможностью выбора пределов измерений, возможностью работы с термопарами типов J, K, T, E, R, S или N, контроля обрыва соединительных линий, компенсацией холодного спая. Измерение температуры может выполняться в градусах Цельсия или Фаренгейта.
- Модуль **EM231 RTD** позволяет производить точное измерение температуры с использованием стандартных термопреобразователей сопротивления (термометров сопротивления). В отличие от других модулей, перечисленных выше, обладает следующими особенностями: возможностью выбора пределов измерений в зависимости от типа используемого датчика (Pt 100/ Pt 200/ Pt 500/ Pt 1000/ Pt 10000, Ni 100/ Ni 120/ Ni 1000, Cu 10, 150/ 300/ 600 Ом), измерения температуры в градусах Цельсия или Фаренгейта.

Перечисленные в данном разделе модули могут использоваться в системах расширения CPU 222/ CPU 224/ CPU 226/ CPU 226XM. Технические данные модулей приведены в [3] и прил. 2.

## 8.2. Способы подключения источников тока и напряжения

В системах автоматизации обычно имеют место три величины, способные нести информацию о величине измеряемого физического параметра – напряжение, ток и сопротивление. На рис. 8.1 показана схема дифференциального подключения датчика напряжения к аналоговому модулю.

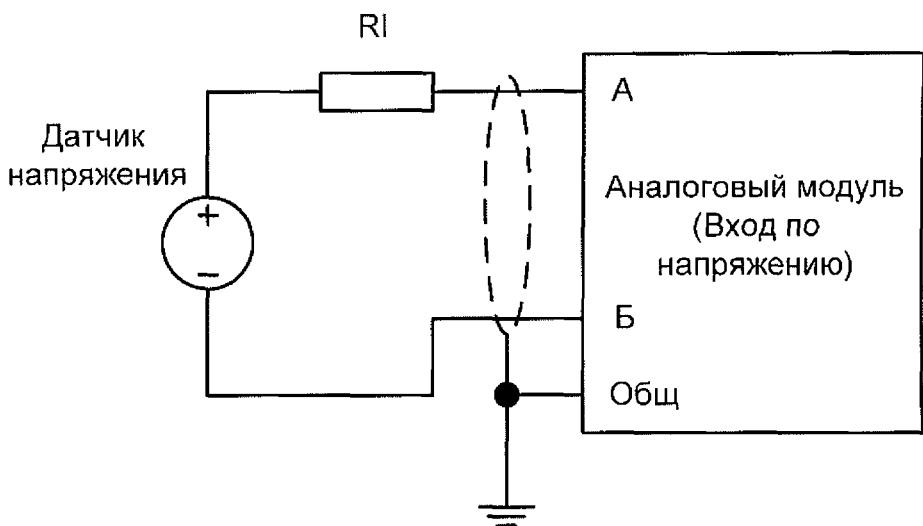


Рис. 8.1. Дифференциальная схема подключения

К достоинствам данной схемы следует отнести ее высокую помехоустойчивость, так как измерению подлежит величина напряжения между точками А и Б (что приводит к подавлению синфазной составляющей), а также возможность измерения величин напряжения в различных точках измерения, не связанных с общим проводом (так называемый «плавающий» источник сигнала). К недостаткам схемы можно отнести необходимость применения трехпроводной линии связи с наличием экрана, назначение которого состоит в снижении уровня наводок на кабель.

На рис. 8.2 показан второй способ подключения датчика напряжения. В этом варианте также использована двухпроводная линия связи, однако особенностью является то, что источник сигнала и аналоговый модуль имеют общую корпусную шину. Такой вариант подключения является единственным возможным в том случае, когда источник сигнала получает питание от того же источника, что и входные цепи измерительной системы.

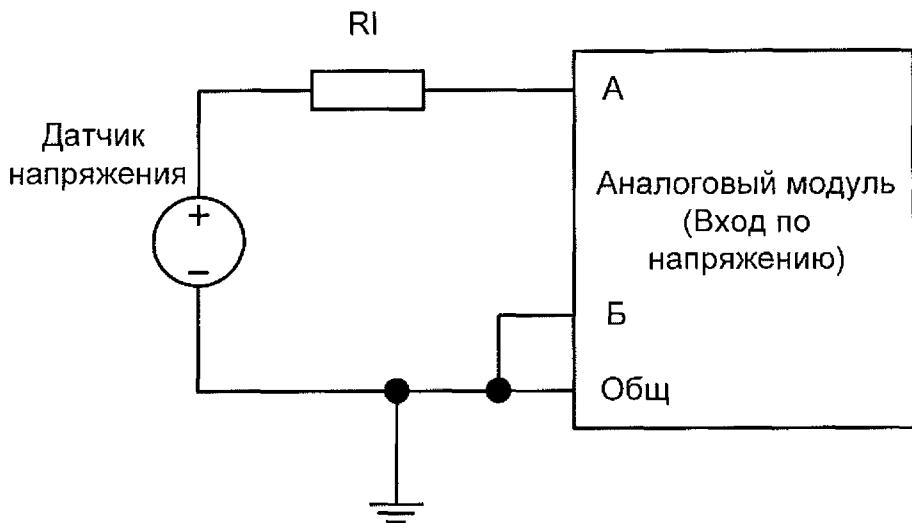


Рис. 8.2. Схема подключения по двухпроводной схеме

Общим недостатком использования входа по напряжению является высокая чувствительность приведенных схем к неизбежным помехам в линии связи. Резистор  $R_1$ , представляющий собой сопротивление линии, является источником напряжения помехи, что приводит к снижению точности измерения вблизи нижней границы диапазона. Кроме этого, при достаточно большой длине линии сопротивление  $R_1$  становится соразмерным с входным сопротивлением модуля. Это приводит к падению напряжения на входе модуля за счет протекания измерительного тока, что увеличивает погрешность измерения.

К достоинствам этих схем подключения можно отнести простоту сопряжения измерительной системы с теми датчиками, величина которых преобразуется непосредственно в напряжение.

Схема подключения источника тока показана на рис. 8.3. В отличие от схем, описанных выше, измеряемым параметром является сила тока в цепи датчика, при этом сам датчик представляет собой полупроводниковый источник тока с большим внутренним сопротивлением. Как правило, преобразователь измеряемого параметра в величину тока (ИП) находится в корпусе датчика и получает питание от измерительной системы либо от внешнего источника. На входе аналогового модуля установлен нагрузочный резистор  $R_t$ , назначение которого – преобразование тока в напряжение. Резистор  $R_t$  обычно входит в состав модуля и подключается к входу на этапе конфигурации. Величина этого резистора, в большинстве случаев, составляет 250 Ом.

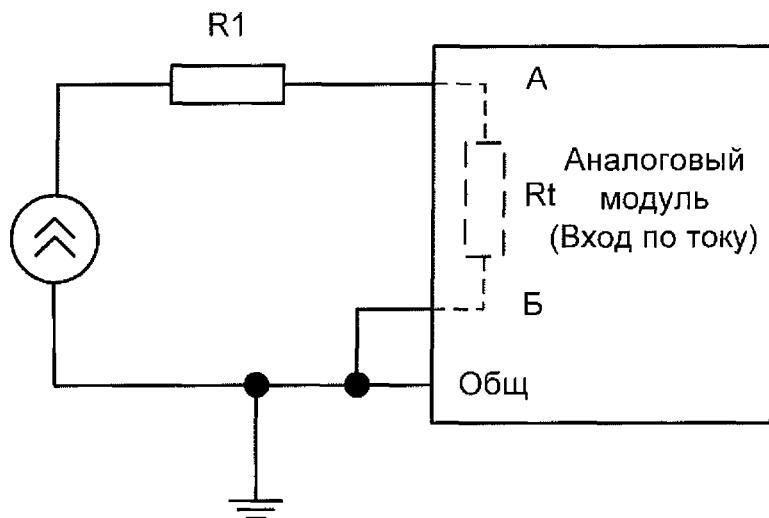


Рис. 8.3. Схема подключения датчика с токовым выходом

Эта схема обладает двумя основными достоинствами – так как источник тока обладает высоким выходным сопротивлением, а сопротивление нагрузки – относительно мало, то значение тока в цепи датчика слабо зависит от сопротивления линии связи, что позволяет удалять датчик от модуля на значительное расстояние (сотни метров). Это имеет большое значение в том случае, когда система автоматизации является распределенной. Вторым достоинством следует считать низкую

чувствительность данной схемы подключения к помехам ввиду малой величины сопротивления  $R_t$ .

Очень часто преобразователь «величина – ток» является двухполюсником, при этом сам преобразователь получает питание по той же линии связи, по которой передается выходной параметр. Это делается для упрощения схемы подключения, так как в этом случае не требуется подавать напряжение питания на измерительный преобразователь по отдельным линиям. На рис. 8.4 показана схема подключения преобразователя «температура-ток» к модулю, имеющему токовый вход.

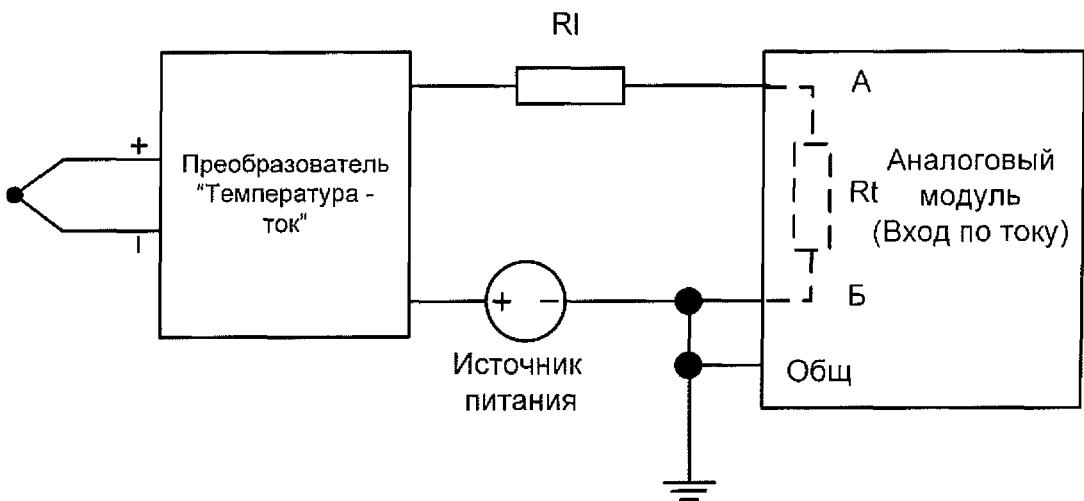


Рис. 8.4. Схема подключения токового преобразователя

В данной схеме источник питания может находиться как на стороне модуля, так и на стороне измерительного преобразователя. В этом случае ИП является модулятором тока, где модулирующим сигналом является величина измеряемого параметра. ИП с двухпроводным подключением имеют более сложную схемотехническую реализацию, однако благодаря универсальному способу подключения используются на практике чаще, чем четырехпроводные.

### **8.3. Способы представления аналоговых величин**

В системах автоматизации используются как униполярные, так и двухполярные величины. При этом значение напряжения может быть как положительным, так и отрицательным, значение сопротивления – только положительным. Диапазон изменений напряжений или токов у разных типов датчиков и исполнительных приводов может быть различным, цифровое же значение преобразованного модулями EM23x сигнала имеет размер 12 бит, что обеспечивает 4096 уровней квантования. Например, при использовании диапазона входного сигнала 0...10 В (или от +5 до -5 В) величина квантования сигнала по уровню будет равна  $10/4096 = 2,44$  мВ. Приведенная погрешность при этом будет составлять  $(2,44 \text{ мВ}/(2 * 10\text{В})) * 100\% = 0,012\%$  (шум квантования). Для входного сигнала 80мВ на данном диапазоне величина этой погрешности будет равна уже 1,5 %. Поэтому для эффективного и точного преобразования различных диапазонов напряжений и токов в цифровой код необходимо использовать масштабирование входной величины, что достигается введением в состав модуля масштабного усилителя, коэффициент усиления которого устанавливается вручную для выбранного диапазона измерения. Модули EM23x рассчитаны на работу в диапазонах, границы которых позволяют подключить большинство существующих на данный момент датчиков с высокой точностью измерения.

Шаг квантования по уровню 1/4096 от максимального значения не очень удобен для представления, так как для многих величин границ диапазонов дает результат деления в виде числа, не кратного двум или пяти. Особенностью модулей EM23x является то, что верхняя граница диапазона измерения приведена к величине 4000, что значительно упрощает вычисления в программах пользователя. Например, в модулях EM235 и EM231 верхней границе диапазона 0...10 В соответствует двоичное число

$111110100000_2$  (вместо  $111111111111_2$ ), что позволяет свести разрешающую способность до величины 2,5 мВ (вместо 2,44). Это упрощает обработку и представление чисел в программе пользователя.

Для модулей, имеющих входы по напряжению (например, EM235), характерен следующий набор диапазонов: 0–50 мВ, 0–100 мВ, 0–500 мВ, 0–1 В, 0–5 В, 0–10 В.

Для датчиков тока имеется один стандартный диапазон – 4–20 мА.

#### 8.4. Подключение и диагностика аналоговых модулей

Внешний вид и расположение цепей подключения на примере EM 235 показан на рис. 8.5.

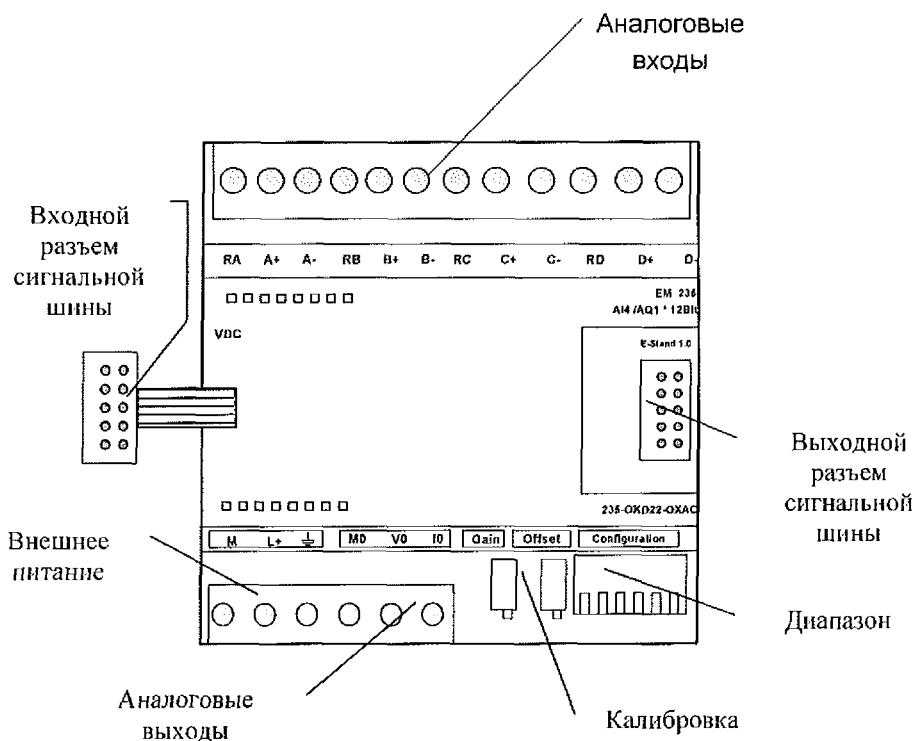


Рис. 8.5. Внешний вид модуля

Для подключения модуля расширения к существующей системе автоматизации на базе S7-200 необходимо выполнить три действия:

- произвести механический монтаж модуля на профильную шину контроллера;
- произвести электрический монтаж соединений, включая внешние датчики и приводы;
- произвести диагностику модуля с помощью программного обеспечения Step7 MicroWin и определить диапазоны входов и выходов подключенного модуля.

Механический монтаж модуля выполняется таким же образом, как и монтаж PLC, так как дизайн модуля совпадает с первым и ориентирован на размещение его на профильнойшине.

На первом этапе электрического монтажа производится соединение шины данных модуля с соответствующим разъемом центрального процессора (сигнальная шина). Если в системе используются несколько модулей расширения, то вновь подключаемый модуль соединяется с разъемом сигнальной шины предыдущего модуля. На втором этапе электрического подключения к модулю подводится внешнее напряжение питания +24 В от центрального процессора либо от внешнего источника питания (например, от стандартного источника SITOP). На третьем этапе производится подключение внешних источников сигнала согласно функциональному назначению модуля и выбор режима с помощью DIP-переключателей (прил. 4).

Диагностика модулей может производиться как на этапе подключения и конфигурирования, так и в процессе работы. Самая общая диагностика производится визуально в процессе работы с помощью светодиодов, расположенных на самом модуле. Состояние модуля доступно и в режиме отладки программного обеспечения в среде Step7MicroWin по команде в основном меню **PLC\Information**. Примеры выполнения данной команды показаны на рис. 8.6 и 8.7.

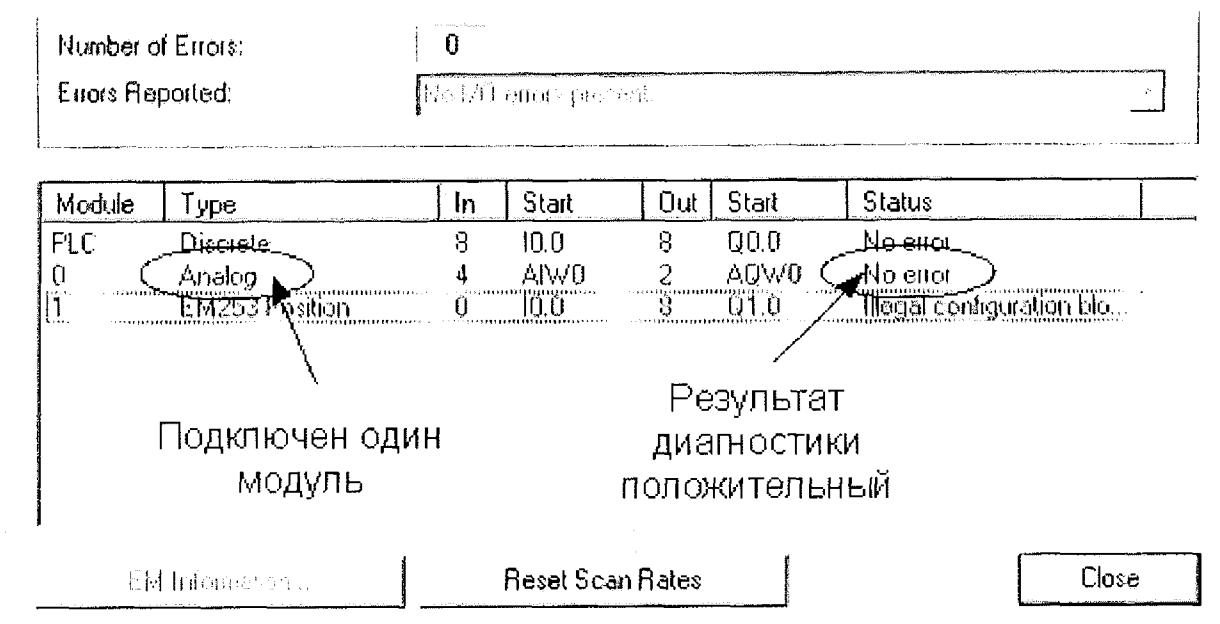


Рис. 8.6. Вид окна диагностики с одним подключенным модулем

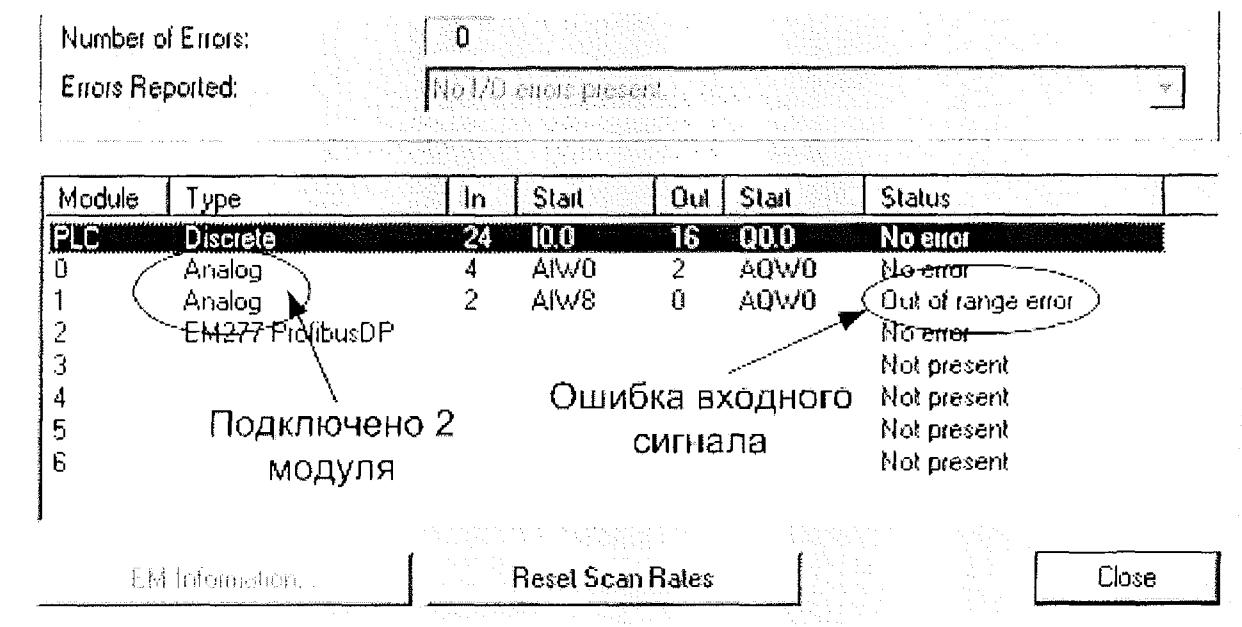


Рис. 8.7. Вид окна диагностики с двумя подключенными модулями

Состояние аналоговых модулей доступно также и в процессе исполнения программы, путем обращения к регистрам специальной памяти SMB8 – SMB21, в которые записывается информация от системы диагностики PLC в каждом цикле выполнения программы. Таким образом,

опрашивая эти значения, можно предусмотреть в алгоритме какие-либо действия, связанные с неисправностью модуля.

## 8.5. Адресация аналоговых входов и выходов

Способ адресации аналоговых входов и выходов оптимизирован с целью упрощения подключения и быстрого ввода модуля в эксплуатацию. Для этого в контроллере используется позиционная адресация входов и выходов. При использовании этого способа в регистре образа цифровых входов и выходов зарезервировано определенное число адресов, количество которых зависит от типа центрального процессора. Это означает, что адрес того или иного аналогового входа или выхода будет зависеть от физического положения модуля по отношению к шине расширения контроллера. Так, например, в CPU 222 Relay имеется 16 точек аналоговых входов и 16 точек аналоговых выходов, при этом адрес входа и выхода может совпадать. Это связано с тем, что обращение к ним производится разными командами. Принято обозначение аналоговых входов как **AIWx** (аналоговый вход) и **AQWx** (аналоговый выход), причем число **x** может принимать только четные значения. Это объясняется тем, что аналоговый вход или выход занимает в регистре образа два байта, причем нумерация начинается с нулевого. При подключении модуля к контроллеру производится автоматическое назначение адресов входам, расположенным на модуле. Диапазон адресов зависит от того, сколько входов содержит подключаемый модуль, а также его расположение по отношению к контроллеру.

На рис. 8.8 показан пример адресации аналогового модуля EM 235 при подключении к контроллеру S7-200 CPU 226XM. Модуль EM235, имеющий в своем составе 4 точки аналогового входа и 2 точки аналогового выхода, занимает адреса AIW0 – AIW6 (соответствуют именам входов A, B, C и D на модуле ) и адреса выходов AQW0, AQW2.

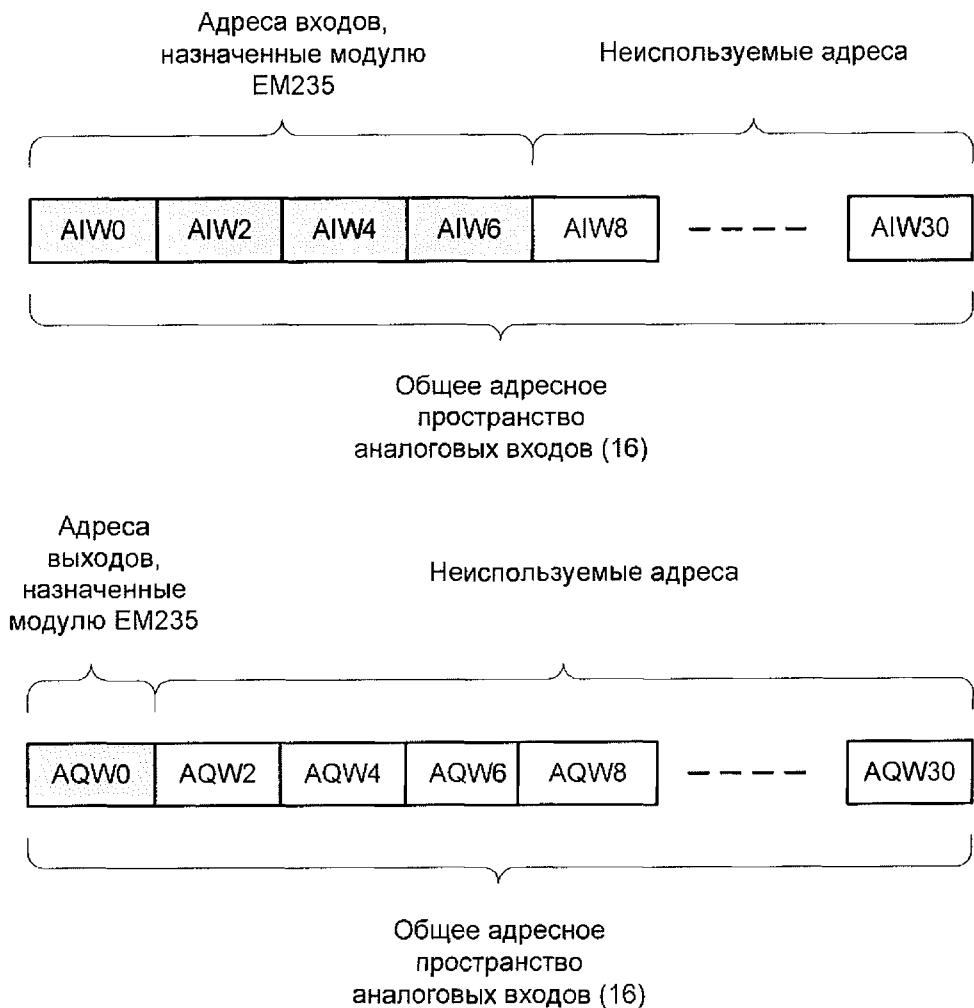


Рис. 8.8. Пример адресации входов EM235

Этот вариант конфигурации действителен в том случае, когда модуль подключен непосредственно к контроллеру, и в этой ситуации нумерация входов начинается с нулевого адреса. На этапе диагностики модуля в среде Step7MicroWin32 можно увидеть назначенные адреса входов (рис. 8.6, колонка Start). Заметно, что этот модуль установлен в слот «0» (т. е. подключен непосредственно к PLC). Начальные адреса входов и выходов — нулевые. Переназначение этих адресов пользователем невозможно. Если к контроллеру подключено несколько модулей, то адреса назначаются по порядку следования. На рис. 8.9. показан пример адресации двух аналоговых модулей EM235 и EM231 RTD при подключении к контроллеру CPU226XM.

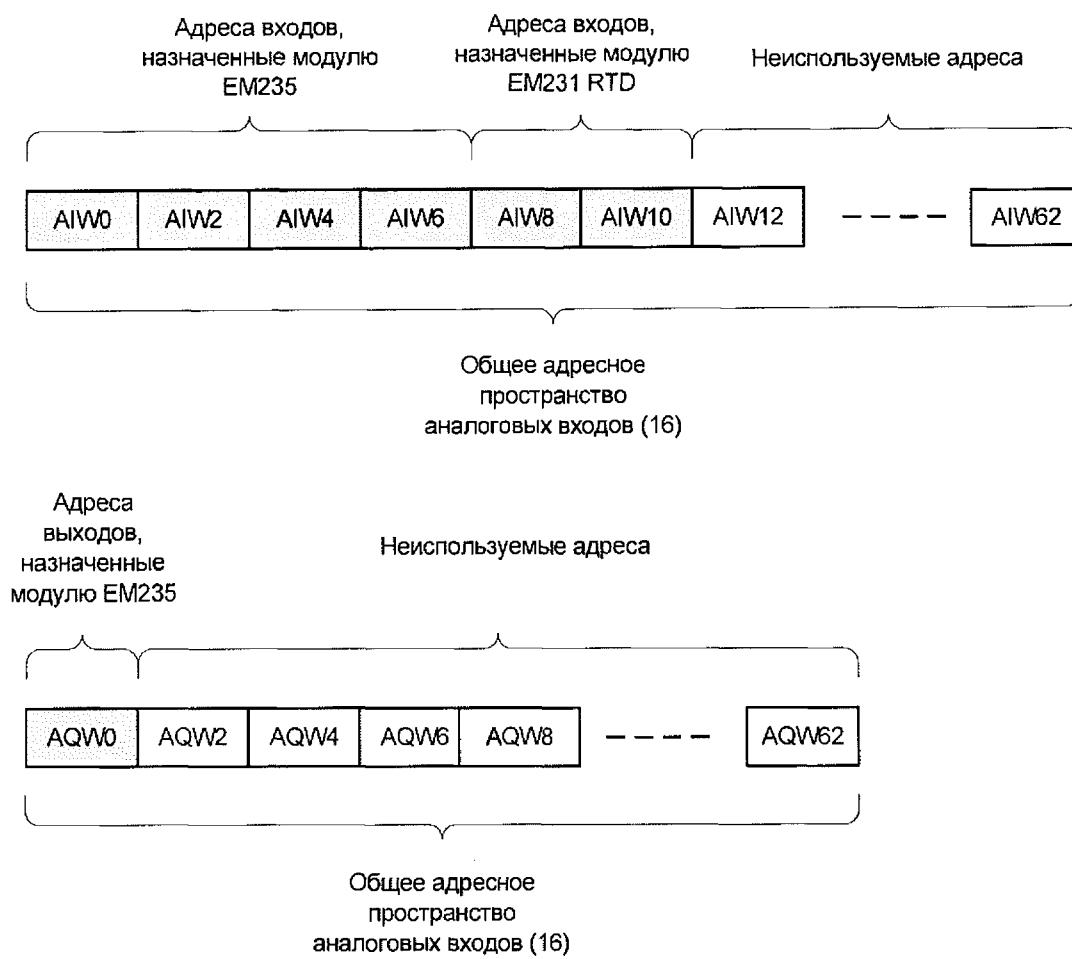


Рис. 8.9. Пример адресации двух модулей

Общее адресное пространство входов выходов составляет 32 входа и 32 выхода. При подключении модулей EM235 и EM231 RTD (в этом порядке следования) контроллер назначает адреса AIW0 – AIW6 модулю EM235, и входы AIW8, AIW10 модулю EM231, так как последний имеет в своем составе два аналоговых входа. Адрес выхода AQW0 соответствует выходу модуля EM235. Последующие адреса выходов остаются неиспользованными, так как модуль EM231 не имеет аналогового выхода.

Порядок назначения адресов операционной системой ПЛК имеет свои особенности. Назначение контроллером адреса входа и выхода осуществляется с шагом 2, это означает, например, что если к системе будет подключен еще один модуль EM235, то его входам будут назначены адреса AQW14 – AQW18, а выходу – AQW4. Данный способ назначения адресов

обусловлен тем, что в ПЛК S7-200 традиционно принята байтовая адресация всех ресурсов, кроме битовых данных. Этот способ исключает неверное толкование адреса для различных типов и размеров данных, расположенных в смежных ячейках памяти.

Следует отметить, что в ПЛК других производителей принятые другие способы адресации. Например, в ПЛК ОВЕН принята технология выравнивания, где адресом фрагмента данных является число, кратное размеру этого фрагмента.

## 8.6. Формат данных аналоговых входов и выходов

При обращении к аналоговому входу или выходу по определенному адресу пользователь фактически обращается не к самому модулю, а к регистру образа этого входа или выхода, расположенного в памяти контроллера, при этом регистр образа обновляется в каждом цикле операционной системы. Формат регистра образа для аналоговых входов показан на рис. 8.10. 12-разрядное слово данных выравнивается по левому краю (в сторону старших бит). Если модуль сконфигурирован на биполярный входной диапазон, то старший разряд слова является знаковым, т.е. модуль числа определяется 11 разрядами. В случае униполярного сигнала знаковый бит не используется (равен нулю), и число определяется 12 разрядами.

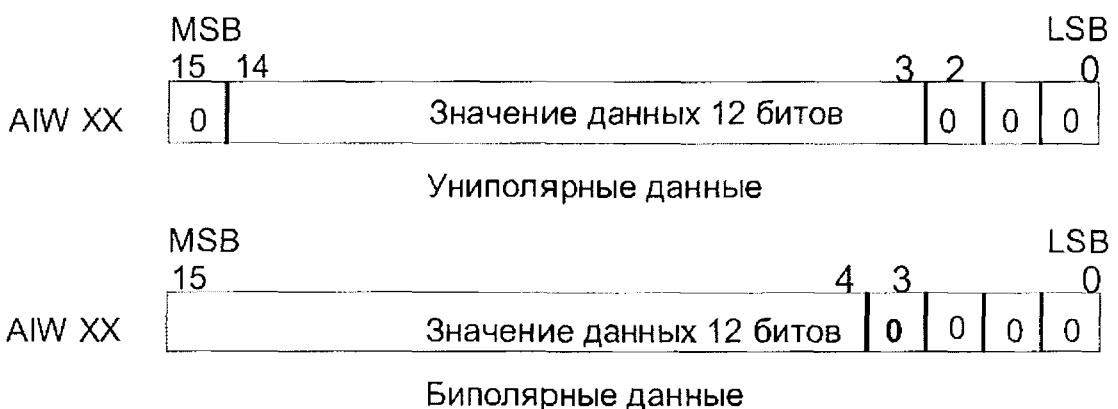


Рис. 8.10. Формат слова аналоговых входов

На рис. 8.11 показан формат слова аналоговых выходов. Выход модулей, как правило, не конфигурируется, т. е. параметры выходного сигнала жестко установлены.

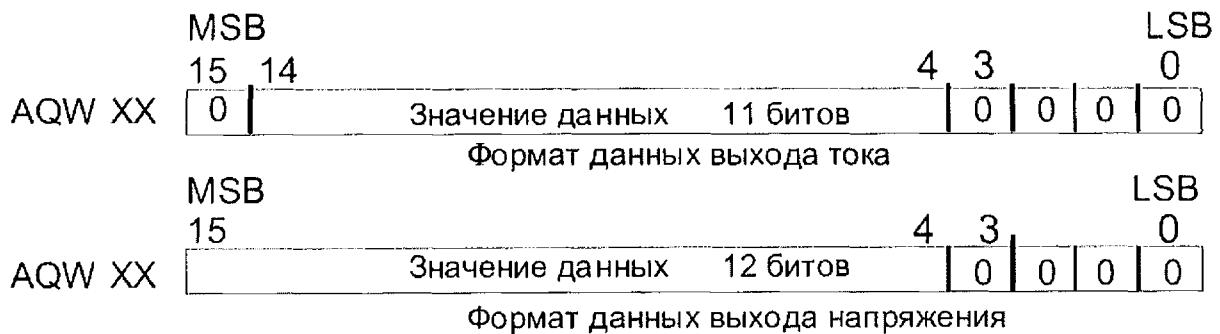


Рис. 8.11. Формат слова аналоговых выходов

Значение выходного тока может лежать в диапазоне 0–20 мА, выходного напряжения – от –10 до +10 В. Например, в модуле EM235 существует только один адрес выхода, однако физических выходов 2 – по току и напряжению. Это означает, что значение, передаваемое по адресу выхода, физически выдается в виде тока и напряжения на двух физических выходах. Формат выхода тока аналогичен униполярному значению, формат выхода напряжения – биполярному.

## 8.7. Пример программы обработки аналоговых величин

Приведем пример простого проекта по работе с аналоговыми модулями. На рис. 8.12 приведена схема подключения имитаторов аналоговых сигналов для модуля EM 235, который подключен к ПЛК S7-200 CPU 222.

Произведем конфигурирование диапазона входных сигналов для входов модуля, согласно приложению В. Установим тип входа «Униполярный», входной диапазон 0–10 В.

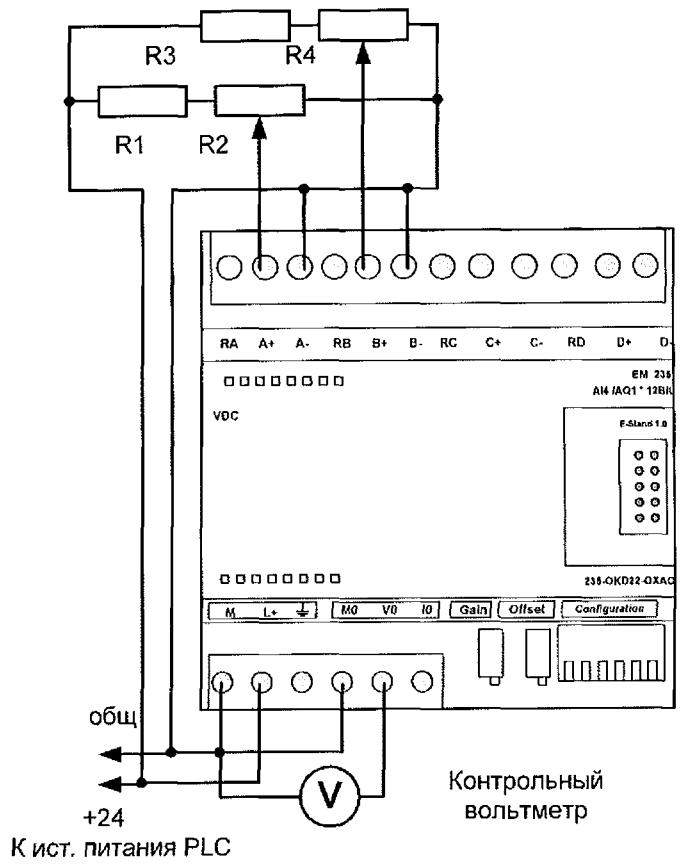


Рис.8.12. Схема лабораторного стенда

Включим питание контроллера и произведем визуальную диагностику состояния модуля. Убедимся, что на модуль подано питание (включен индикатор на модуле).

Произведем диагностику модуля в среде Step7 MicroWin 32, выполнив команду меню **PLC\Information**. Убедимся, что модуль исправен, а также определим адреса аналоговых входов и выходов.

*Примечание. В базовой конфигурации (модуль включен в слот 0) стартовые адреса входов и выходов должны быть AIW0 и AQW0. Для варианта с несколькими подключенными модулями адреса входов и выходов EM235 будут зависеть от положения модуля относительно PLC.*

Убедимся, что адрес PLC в панели оператора совпадает с адресом используемого PLC. Загрузим в PLC проект *VOLTAGE.MWP* и выполним старт программы (рис. 8.13).

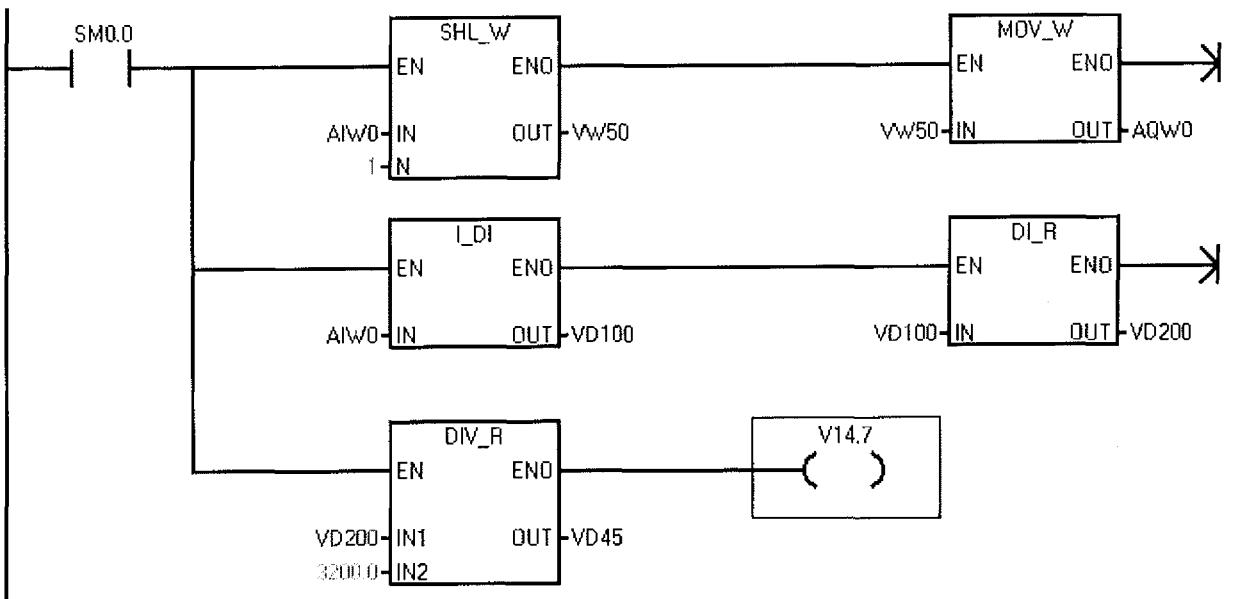


Рис. 8. 13. Программа, реализующая функцию вольтметра

Программа реализует алгоритм перевода величины напряжения, выраженного двоичным кодом в диапазоне 0-32000 (для 12 разрядов со смещением вправо) в величину напряжения в вольтах с последующим отображением этого значения на панели оператора. Для контроля входное напряжение дублируется на аналоговом выходе.

*Примечание. Панель оператора TD200 отображает значение выбранной ячейки памяти  $V$  как значение с плавающей точкой. В данном проекте панель оператора может отсутствовать, поэтому необходимость в установлении бита  $V14.7$  (разрешение отображения) отпадает. Значение напряжения можно просматривать в режиме «Programm Status»*

Так как полный диапазон входного напряжения составляет 0–10 вольт, что соответствует диапазону 0–32000 в двоичном эквиваленте, то для отображения значения в вольтах можно использовать формулу:  $U(B) = \text{код} \cdot (10/32000)$ .

Программа содержит блок данных для панели оператора, сконфигурированной для вывода сообщения, содержащего внедренные данные (Real) с отображением двух знаков справа от запятой. Сообщение не

требует квитирования и выводится после установки бита V14.7. В первом блоке производится считывание аналогового входа и сдвиг значения влево на один разряд. Следующий блок выдает модифицированное значение на аналоговый выход. Далее, производится последовательное преобразование целого представления числа в формат слова в представление «real». Таким образом, исходное значение кода преобразовано в формат с плавающей точкой для последующей операции деления. В следующем блоке производится деление преобразованного кода на величину 3200.0 (упрощенное выражение «10/32000») с записью результата в ячейку памяти VD45 и установкой бита разрешения выдачи сообщения V14.7. адреса VD45 и V14.7 были определены ранее при конфигурировании сообщений для панели оператора TD200.

Приведенная программа может служить основной для более сложных алгоритмов обработки. По этой причине целесообразно оформить программу в виде функции.

## 8.8. Варианты заданий

Для всех вариантов (табл. 8.1) справедливы следующие обозначения:

- X<sub>1</sub>, X<sub>2</sub> – напряжения на аналоговых входах, задаются с помощью потенциометров R2 и R4;
- Y – напряжение на аналоговом выходе;
- для всех вариантов полный диапазон регулировки потенциометра составляет 0–270 градусов, диапазон напряжений 0–10 В.

Так как модуль визуализации TD 200 (текстовый дисплей) является опциональный узлом, все варианты могут быть выполнены без конфигурирования TD 200. Текстовый дисплей подключается к ПЛК через свободный порт RS485 и является Master-устройством. Если контроллер имеет один порт, необходимо отключать TD200 на момент

программирования и отладки проекта от Ste7 MicroWin 22 во избежание сетевых конфликтов.

Таблица 8.1

Варианты заданий

№	Задание	Единица измерения	Отображение на TD200	Y
1	Написать программу, контролирующую разность значений двух аналоговых входов. Если величина рассогласования составляет 5% от U1, включить цифровой выход Q0.0	Вольты	X1, X2, X1-X2	X1-X2
2	Установить на выходе величину напряжения, равного значению X1. С помощью X2 производить точную подстройку выхода в положительную и отрицательную сторону от значения, соответствующего центральному положению движка резистора. Масштаб подстройки от -20 до +20 % от максимального значения X1	Вольты	X1, значение X1, скорректированное с помощью X2	Скорректированное значение X1
3	Написать программу, контролирующую разность между значениями двух аналоговых входов. Если величина рассогласования составляет 10% от X1, включить цифровой выход Q0.0. При рассогласовании 1% включить дополнительно Q0.1	Градусы	X1, X2, сообщения о порогах 10% и 1%	X1-X2

Продолжение таблицы 8.1

№	Задание	Единица измерения	Отображение на TD200	Y
4	Написать программу, контролирующую разность между значениями двух аналоговых входов. Если величина рассогласования составляет 8% от 10 В (конец шкалы) , включить цифровой выход Q0.0. При достижении X1 80% от конца шкалы, включить цифровой выход Q0.1 и изменить порог рассогласования на 3%.	Вольты	X1, X2, X1-X2, сообщение о смене порога	X1-X2
5	Установить на выходе величину напряжения, равного значению X2. С помощью X1 производить точную подстройку выхода в положительную и отрицательную сторону от значения, соответствующего центральному положению движка резистора. масштаб подстройки от -10 до +10 % от максимального значения X2	Градусы	X2, значение X2, скорректированное с помощью X1	Скорректиро ванное значение X1
6	Производить ступенчатый вывод значения X1. Шаг дискретизации по уровню регулируется с помощью X2 и изменяется в диапазоне от бесконечности до 5	Вольты	X1, ступенчатое X1, шаг дискретизации	Ступенчатое X1

№	Задание	Единица измерения	Отображение на TD200	Y
	Производить ступенчатый вывод значения X1. Шаг дискретизации по уровню регулируется с помощью X2 и изменяется в диапазоне от -10 до +10, начальное значение равно 5	Вольты	X1, ступенчатое X1, шаг дискретизации	Ступенчатое X1
7	Вычислять относительную погрешность величины X2 по отношению к X1 с учетом знака. Если погрешность превышает порог +10%, установить выход Q0.0. Если величина погрешности превышает -20%, установить Q0.1. В остальных случаях установить Q0.2	Вольты	X1, X2, погрешность в процентах	Погрешность в процентах
8	Произвести масштабирование X1. Для этого, установить X2 в произвольное положение, и принять значение X2 за 100%. При изменении X1 производить отсчет показаний в процентах по отношению к максимальному значению X2	Вольты	X1, X2, X1 в процентах	X1 в процентах
9	Произвести вывод на панель оператора значений X1 и X2. с шагом в 10 градусов. При величине рассогласования менее 10% производить вывод значений с повышенной точностью с шагом в 1 градус. Дополнительную точность показывать в отдельном поле	Градусы	X1, X2 (в вольтах и градусах), рассогласова ние с повышенной точностью в градусах.	Рассогласо вание в градусах.

## **9. ИЗМЕРЕНИЕ ТЕМПЕРАТУРЫ**

### **9.1. Измерительные преобразователи «температура-ток»**

Измерительный преобразователь — техническое средство с нормируемыми метрологическими характеристиками, служащее для преобразования измеряемой величины в другую величину или измерительный сигнал, удобный для обработки, хранения, дальнейших преобразований, индикации и передачи, но непосредственно не воспринимаемый оператором. ИП или входит в состав какого-либо измерительного прибора (измерительной установки, измерительной системы и др.), или применяется вместе с каким-либо средством измерений.

В контексте АСУ ТП измерительный преобразователь конвертирует измеряемую величину в электрический сигнал напряжения или тока. В автоматике используются следующие стандартные диапазоны выходного сигнала:

- 0...10 В;
- -10...+10 В;
- 0...20 мА;
- 4...20 мА;
- -20...+20 мА.

Наиболее часто используются преобразователи «величина - ток», так как благодаря слабой зависимости измеренной величины от сопротивления линии ИП может быть удален от ПЛК на значительное расстояние.

На измерительный преобразователь в общем случае возлагаются следующие задачи:

- формирование токового сигнала, пропорционального измеряемой величине;
- масштабирование измеряемой величины;

- предварительная обработка получаемых значений (например, линеаризация, фильтрация, клиппирование).

Очень часто измерительные преобразователи конструктивно объединены с самим датчиком, однако имеется возможность замены самого датчика. Это связано тем, что для измерений температуры существуют три вида сенсоров – термометры сопротивления, термопары и полупроводниковые терморезисторы (термисторы). Для более сложных измерительных систем используются универсальные измерительные преобразователи, работающие с определенной группой датчиков. Такие ИП позволяют производить предварительную обработку аналоговых величин с возможностью вывода измеряемого параметра в виде тока, напряжения, частоты следования импульсов, допускают подключение нескольких каналов измерения. При этом упрощается обработка и масштабирование получаемых данных в PLC.

На рис. 9.1 приведена обобщенная схема ИП.

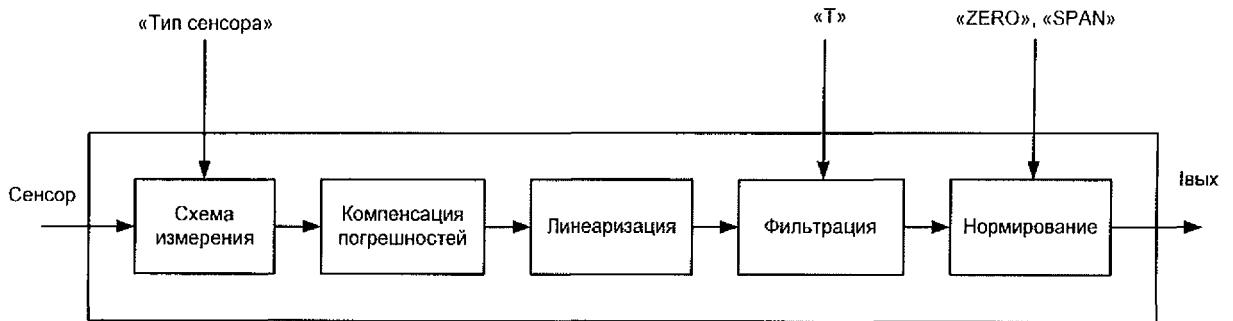


Рис. 9.1. Функциональная схема ИП

Настройка ИП в общем случае заключается в выборе типа сенсора и его параметров (например, сопротивления и коэффициента преобразования, соответствующего материалу резистора), величины начала шкалы измерения («ZERO»), величины предельного значения шкалы измерения («SPAN»), и в некоторых случаях – величину постоянной времени фильтра для устранения резких колебаний значения и снижения уровня помех в выходном сигнале.

На практике конфигурирование преобразователей производится тремя способами:

- регулировочными потенциометрами (для простых ИП);
- с помощью цифрового интерфейса или DIP-переключателями (для ИП, в состав которых входит микроконтроллер);
- с помощью протокола HART (обычно, для универсальных ИП с поддержкой различных типов сенсоров).

Подключение ИП к измерительной системе возможно при наличии на стороне ПЛК токового входа. Для упрощения часто используется двухпроводная схема, что подразумевает одновременную передачу энергии питания преобразователя и выходного сигнала тока по одной линии связи. Преобразователь в данном случае является устройством, ток потребления которого от источника питания модулируется измеряемым параметром и лежит в определенных пределах. Наиболее часто используемый диапазон изменения тока - 4...20 мА. Наличие тока в линии (4 мА) при минимальном значении параметра позволяет контролировать целостность линии связи, а также исправность самого преобразователя. Например, при величине тока 3.8 мА говорит о неисправности преобразователя или выходе значения за измерительный диапазон, а значение 0 мА – о нарушении целостности линии связи между ИП и ПЛК.

Рассмотрим пример подключения и использования ИП «температура-ток» **ТСПУ-205ex**, в состав которого входит температурный датчик на основе термометра сопротивления (Pt100) и измерительный преобразователь **ПТ 055/205** с унифицированным выходом по току 4...20 мА. В состав измерителя входит также компенсатор нелинейности измеряемой величины. ИП смонтирован в головке датчика и может быть заменен. Вариант исполнения «Ex» предназначен для использования в помещениях, где возможно образование взрывоопасных смесей категорий IIА, IIВ и IIС и групп взрывобезопасности Т1...T6. Области применения ИП - энергетика,

нефтяная и нефтехимическая промышленность, горнодобывающая промышленность, газовая промышленность.

Основные технические характеристики ТСПУ-205ех приведены в табл.

### 9.1.

Таблица 9.1

#### Технические данные ТСПУ-205ех

Диапазон температур	0...300°C
Выходной ток	4...20mA
Показатель тепловой инерции	20 с
Сопротивление нагрузки	до 1000 Ом
Предельное избыточное давление	1 МПа
Схема подключения	двухпроводная
Напряжение источника питания	18-36 В
Потребляемая мощность	не более 0.8 Вт
Степень защиты	IP54
Относительная погрешность	0.5%

Подключение ТСПУ-205 к модулю ввода-вывода аналоговых сигналов показано на рис. 9.2.

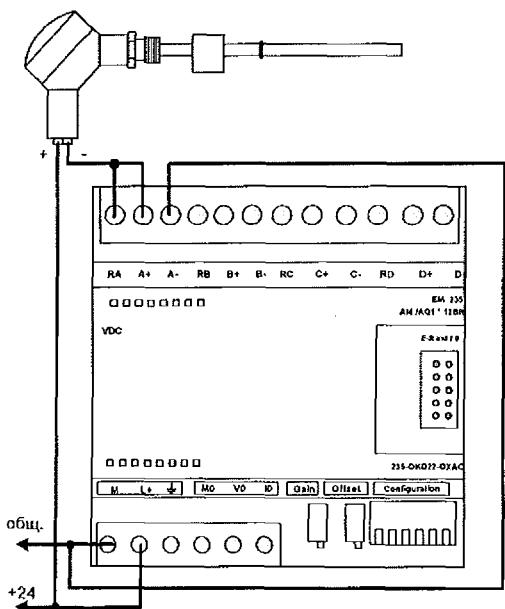


Рис. 9.2. Схема подключения ТСПУ

Преобразователь включен в разрыв провода, соединяющего последовательно источник питания PLC (или модуля), и токового входа EM235, сконфигурированного соответствующим образом (как вход тока 4...20 мА). Статическая характеристика преобразователя приведена на рис. 9.3.

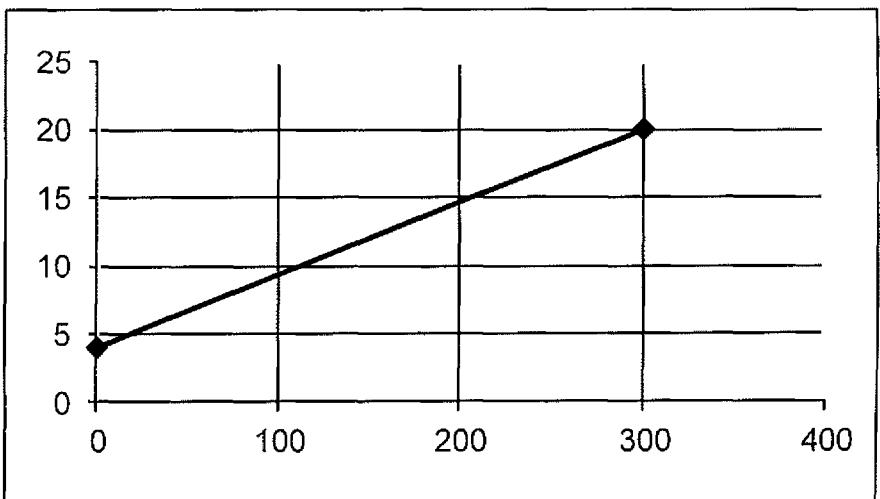


Рис. 9.3. Статическая характеристика ТСПУ

По виду полученной характеристики можно получить ее аналитическое выражение  $I=f(T)$ , при условии отсутствия нелинейностей:

$$I = \frac{I_{\max} - I_{\min}}{300} \cdot T + 4, \quad (9.1)$$

где

$I_{\max}, I_{\min}$  – границы диапазона изменения тока (20 мА, 4 мА);

$T$  – температура в градусах Цельсия.

На рис. 9.4 приведена характеристика преобразования величины тока в цифровой код, при условии нормирования максимального значения тока к величине 4000.

*Примечание. Любая схема преобразования допускает небольшой выход значений тока за указанные пределы с целью диагностики неисправностей ИП.*

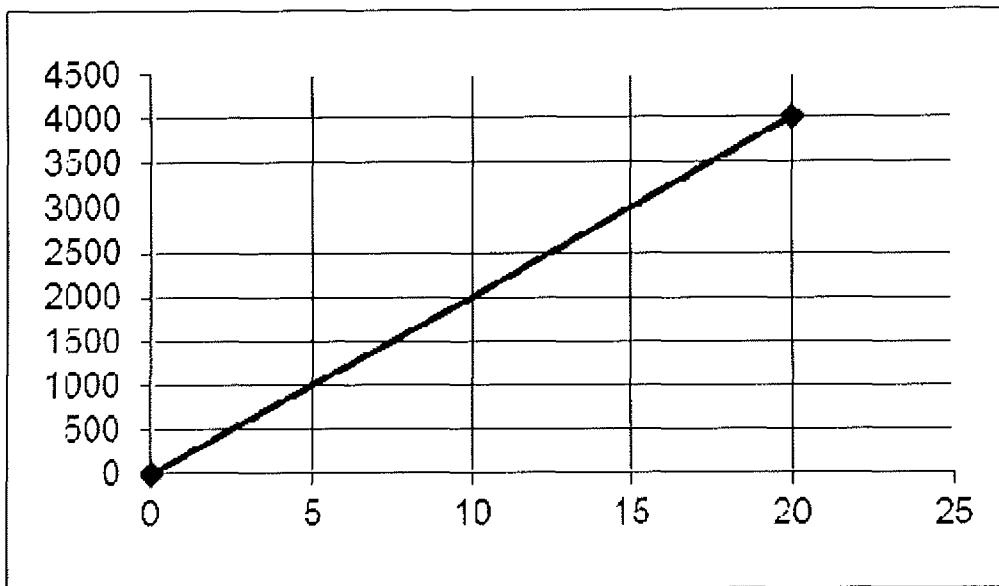


Рис. 9.4. Характеристика преобразования

Уравнение этой характеристики выражается формулой

$$I = \frac{20 \cdot N}{4000}, \quad (9.2)$$

где

$N$  – величина кода, представленная 12-битным кодом;

$I$  – величина тока в милиамперах.

Приравнивая левые части характеристик (9.1) и (9.2), можно получить зависимость величины кода от температуры. После упрощения уравнение выглядит следующим образом:

$$T = \frac{N \cdot 15}{160} - 75. \quad (9.3)$$

Приведенная зависимость (9.3) позволяет рассчитывать значение температуры в градусах и использовать это значение, не учитывая промежуточное преобразование в величину тока. В общем случае, значение температуры может храниться в формате с плавающей точкой, однако при

величине погрешности преобразователя, составляющей единицы процентов, не имеет смысла выводить значения температуры с точностью до сотых долей градуса. Для упрощенных вычислений можно использовать в формуле (9.3) целочисленные операции.

На рис. 9.5 приведен текст программы проекта *TEMP.MWP* на языке LAD, предназначенный для измерения температуры.

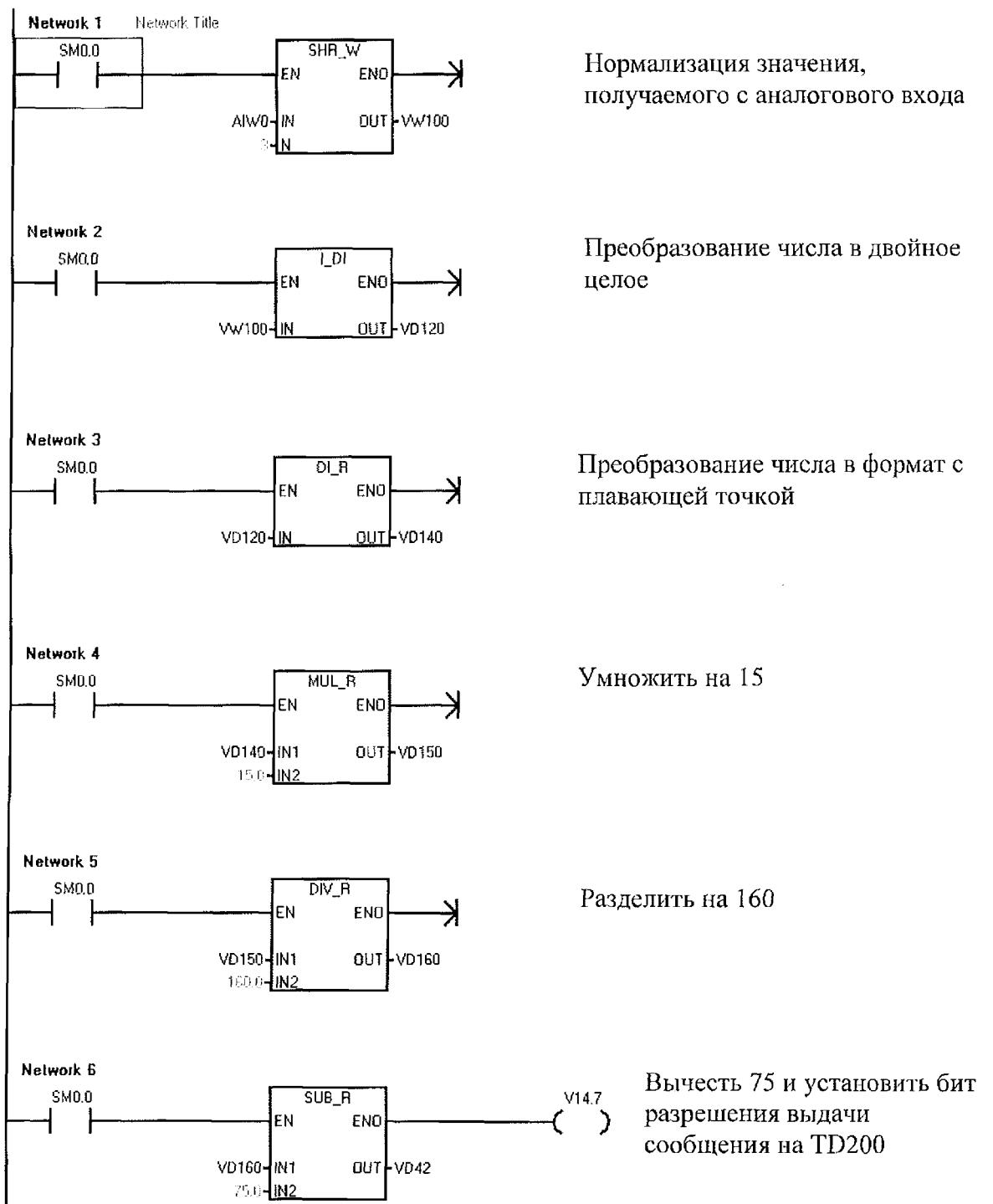


Рис. 9.5. Проект «TEMP.MWP»

## 9.2. Термометры сопротивления

Термометр сопротивления - это электронный прибор, предназначенный для измерения температуры. Принцип действия основан на зависимости электрического сопротивления металлов, сплавов и полупроводниковых материалов от температуры. При применении полупроводниковых материалов его обычно называют термосопротивлением, терморезистором или термистором.

Металлический термометр сопротивления представляет собой резистор, выполненный из металлической проволоки или пленки и имеющий известную зависимость электрического сопротивления от температуры. Наиболее распространённый тип термометров сопротивления - платиновые термометры. Это объясняется тем, что платина имеет стабильную и хорошо изученную зависимость сопротивления от температуры и высокую стойкость к окислению, что обеспечивает их высокую воспроизводимость при массовом производстве. Эталонные термометры изготавливаются из платины высокой чистоты с температурным коэффициентом 0,003925. В качестве рабочих средств измерений применяются также медные и никелевые термометры. Промышленные платиновые термометры сопротивления в большинстве случаев считаются имеющими стандартную зависимость сопротивление-температура (НСХ), что обуславливает погрешность не более 0,1 °C (класс АА при 0 °C).

Термометры сопротивления (терморезисторы) подключаются к измерительной системе тремя способами (рис. 9.6 - 9.8). Измерение температуры производится методом измерения сопротивления RTD. В общем случае, для измерения сопротивления через RTD пропускается постоянный ток заданной величины (не вызывающий разогрев RTD). Падение напряжения на RTD преобразовывается в цифровой код.

На рис. 9.6 приведена двухпроводная схема подключения RTD, являющаяся самой простой по исполнению.

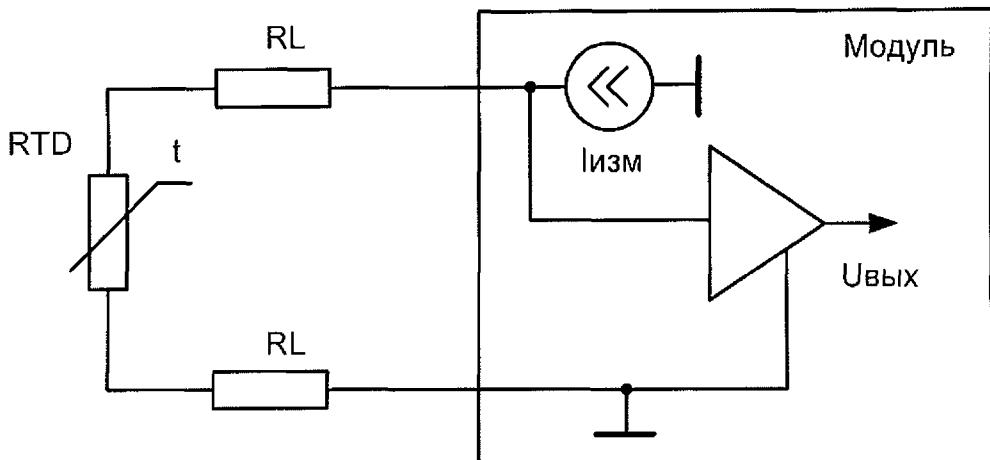


Рис. 9.6. Двухпроводная схема подключения RTD

Напряжение на выходе измерительного усилителя с коэффициентом усиления  $K=1$  можно рассчитать по формуле

$$U_{\text{вых}} = I_{\text{изм}} \cdot (RTD + 2 \cdot RL), \quad (9.4)$$

где

$RL$  – сопротивление одного провода соединительной линии;

$I_{\text{изм}}$  – измерительный ток.

Очевидно, что при  $RL \neq 0$  измерение сопротивления будет производиться с погрешностью, зависящей от длины линии. Минимизировать данную погрешность можно следующими способами:

- уменьшить длину линии;
- увеличить сечение провода, уменьшив, таким образом, величину  $RL$ ;
- увеличить соотношение  $RL/RTD$  таким образом, что  $RTD \gg RL$ ;
- скомпенсировать аддитивную погрешность, зная величину  $RL$ .

На практике величина RTD составляет 50, 100, 200 Ом, что при относительно коротких линиях связи обеспечивает минимальную погрешность, при условии, что линия находится в неизменном

температурном фоне. Если известна температура линии, можно скомпенсировать погрешность, связанную с конечной величиной  $RL$ . Часто такая схема измерения используется при небольшом удалении RTD от модуля измерения.

В тех случаях, когда сопротивление линии изменяется от температуры, скомпенсировать погрешность путем простого вычитания значения  $I_{изм}RL$  нельзя. В этом случае применяется трехпроводная схема подключения, позволяющая скомпенсировать изменяющееся сопротивление  $RL$  путем его измерения. Трехпроводная схема измерения приведена на рис. 9.7.

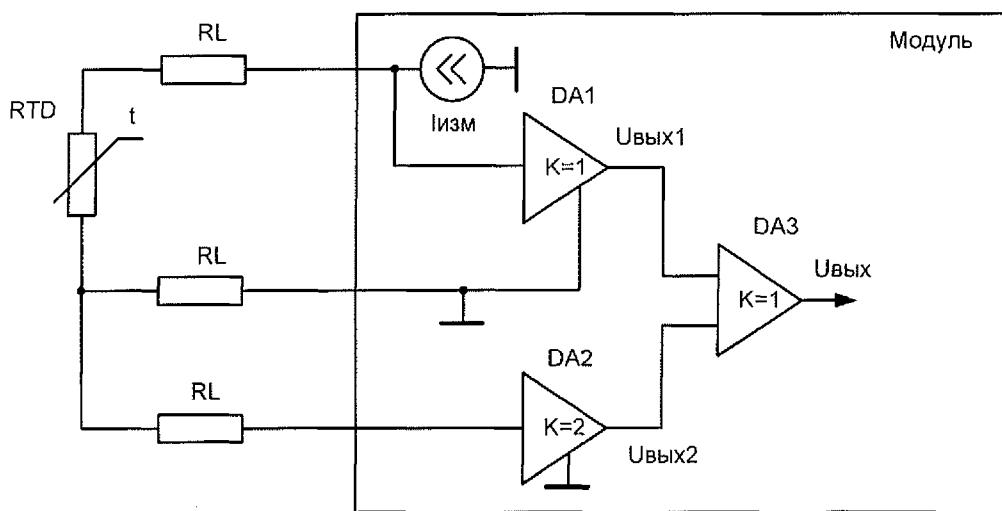


Рис. 9.7. Трехпроводная схема подключения RTD

В данной схеме  $U_{вых1}$  рассчитывается по формуле (1). Напряжение на выходе DA2 можно рассчитать по формуле

$$U_{вых2} = I_{изм} \cdot RL \cdot 2. \quad (9.5)$$

Данное выражение основано на том, что входное сопротивление DA2 стремится к бесконечности. По этой причине ток в цепи входа DA2 отсутствует, следовательно, напряжение на выходе ОУ DA2 равно напряжению на нижнем (по схеме) выводе RTD, которое определяется как  $I_{изм}RL$ . Умножая это значение на 2, получим результат, показанный в

формуле (9.5). Так как DA3 – дифференциальный усилитель с  $K=1$ , напряжение на его выходе:

$$U_{\text{вых}} = U_{\text{вых1}} - U_{\text{вых2}} = (I_{\text{изм}} \cdot RTD + I_{\text{изм}} \cdot 2 \cdot RL) - I_{\text{изм}} \cdot 2 \cdot RL,$$

откуда

$$U_{\text{вых}} = I_{\text{изм}} \cdot RTD. \quad (9.6)$$

Таким образом, трехпроводная схема обеспечивает автоматическую компенсацию падения напряжения в соединительных линиях. Очевидно, что для точных измерений необходимо соблюдать следующих условий:

- все линии связи должны иметь одинаковое сечение и материал;
- все линии должны находиться в непосредственной близости друг от друга и в одном температурном поле.

Если обеспечить перечисленные требования, трехпроводная схема устраняет недостатки двухпроводной, однако требует трех соединительных линий связи.

Если возникают принципиальные сложности в обеспечении перечисленных выше требований, применяется четырехпроводная схема подключения, показанная на рис. 9.8.

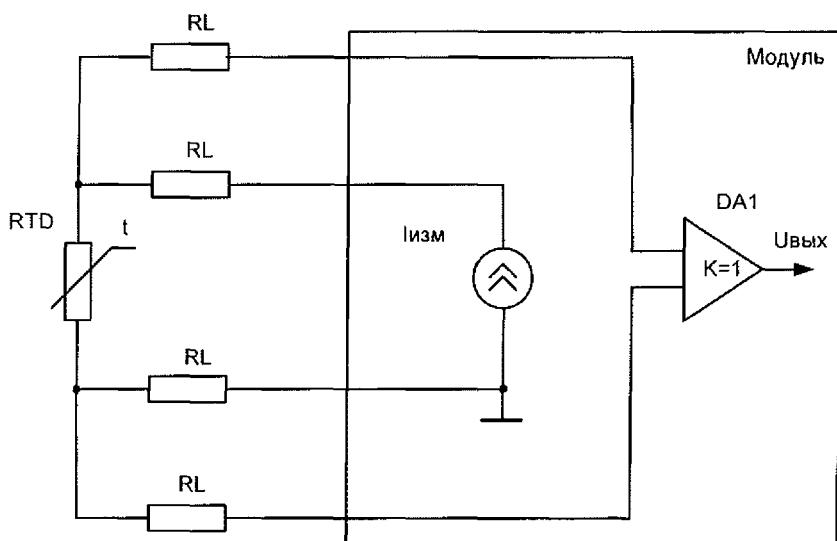


Рис. 9.8. Четырехпроводная схема подключения RTD

В данной схеме линии питания и линии измерения принципиально разделены. Так как входное сопротивление DA1 велико, падение напряжения в соединительных линиях, подключенных к входу DA1 отсутствует.

Таким образом, на выходе DA1 (дифференциальный усилитель) возникает напряжение, равное разности напряжений на двух выводах RTD.

Четырехпроводная схема обеспечивает точное измерение сопротивления RTD и нетребовательна к параметрам соединительных линий.

Модули, предназначенные для подключения датчиков RTD, обеспечивают все три приведенные схемы подключения.

### **9.3. Формат данных и схема подключения модуля EM 231 RTD**

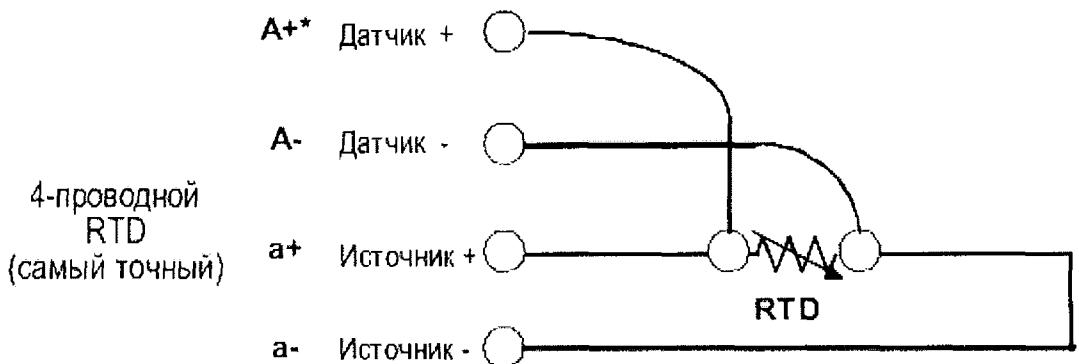
Модуль EM235RTD разработан специально для измерения температуры, поэтому его способ представления аналоговых величин отличается от других модулей и имеет особенность – двоичное число, считываемое с регистра модуля, пересчитано непосредственно в градусы Цельсия или Фаренгейта, включая отрицательные значения (зависит от положения переключателя конфигурации). Это освобождает пользователя от необходимости программного масштабирования получаемых значений.

Так как размер слова составляет 16 бит, то полный диапазон знакового числа составляет (-32768...+32768). Аппаратная часть модуля позволяет производить измерения температуры с точностью 0.1°C, в диапазоне (-243°C...+1000) °C, поэтому последний разряд представлен как десятые доли градуса. Таким образом, в десятичном эквиваленте значение температуры лежит в диапазоне (-243.0°C ...+1000.0) °C, но так как формат аналогового входа – целое число со знаком, то десятичная точка при отображении должна быть введена пользователем самостоятельно при выводе данных, или при операциях со значением температуры в программе. Например, значение

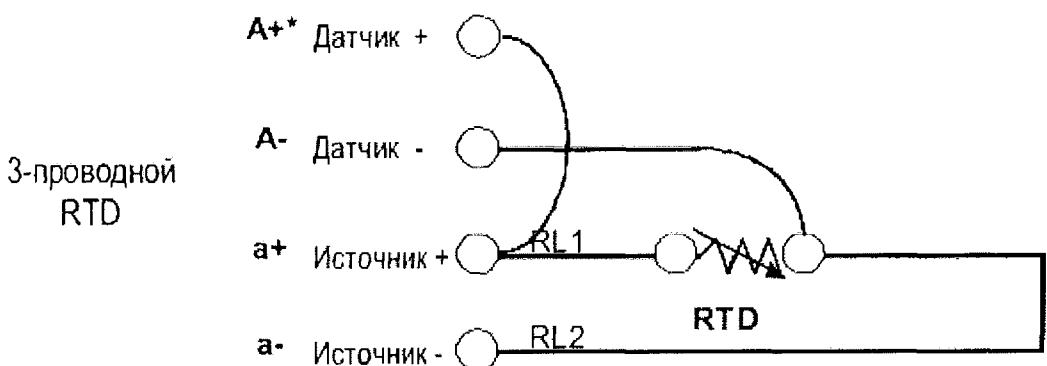
температуры 20.6 °С будет представлено в регистре AIWx как 206 (0CEH). Для вывода числа на панель оператора необходимо отделить дробную часть от целой, разделив полученное значение на 10.

Технические данные EM 231 RTD приведены в прил. 3.

Схемы подключения датчиков RTD к модулю показаны на рис. 9.9.

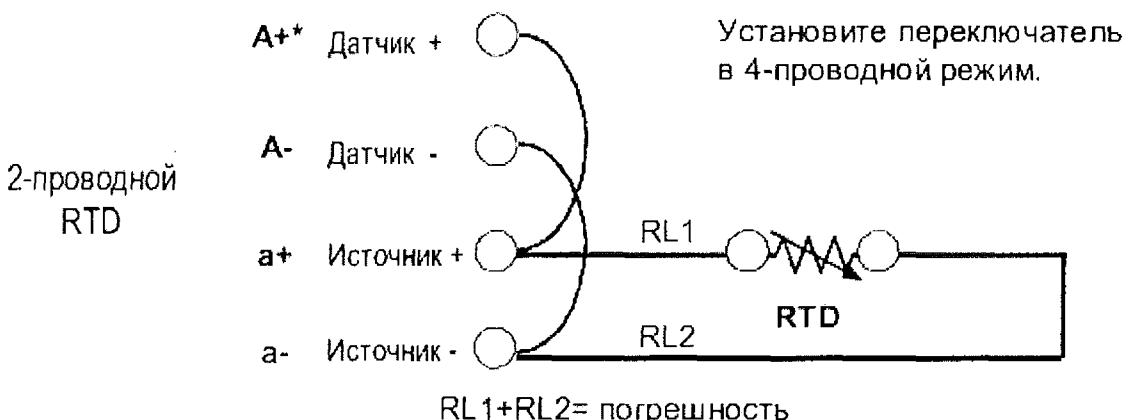


\* А относится к датчику; а относится к источнику питания.



Если  $RL_1 = RL_2$ , то погрешность минимальна.

\* А относится к датчику; а относится к источнику питания.



\* А относится к датчику; а относится к источнику питания.

Рис. 9.9. Схемы подключения RTD к EM321 RTD

Как видно из приведенных схем, отличие двухпроводной схемы от трехпроводной не требует изменения конфигурации модулей, тогда как перевод на четырехпроводную схему требует установления DIP-переключателя и перезагрузки питания модуля. Способы конфигурирования EM231 приведены в прил. 5, таблица диагностирования неисправностей приведена в прил. 6.

Приведем пример получения данных от RTD. В данном примере используется стенд на базе ПЛК S7-200 CPU 226XM с подключенным модулем EM 231 RTD. Термометр сопротивления PT100 подключен по двухпроводной схеме так, как показано на рис. 9.10.

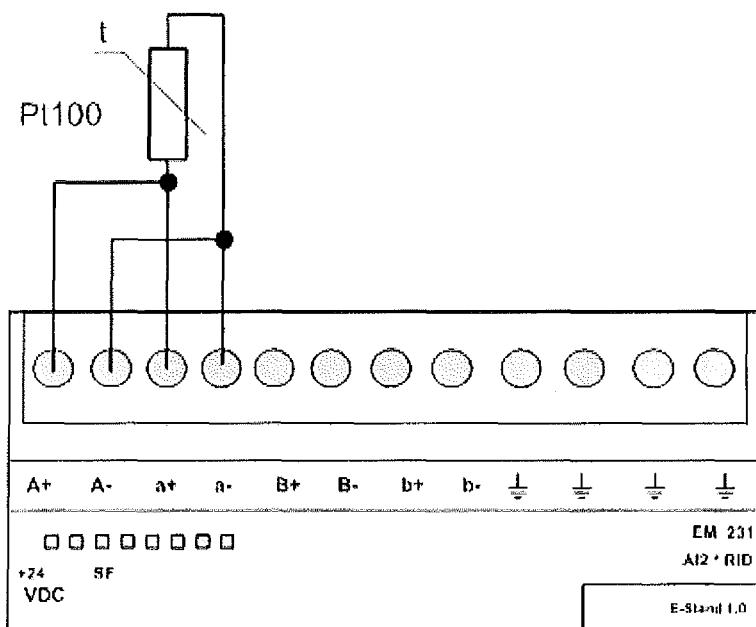


Рис. 9.10. Схема подключения терморезистора

Соединительные перемычки между генератором тока и измерительным входом установлены непосредственно на модуле. Для получения правильных результатов измерения температуры необходимо сделать следующее:

- Выбрать тип терморезистора, путем установки конфигурационных переключателей SW1-SW5 на самом модуле (прил. 5);

- Выбрать температурную шкалу измерений – отсчет температуры в градусах Цельсия или Фаренгейта путем установки SW7 в соответствующее положение.
- Выбрать тип схемы подключения – трехпроводная или двухчетырехпроводная, используя переключатель SW8.
- Выбрать направление установки предельной величины датчика, путем установки SW6. Так как любой терморезистор имеет определенный диапазон сопротивлений, отличный от нуля и бесконечности, то разработчики предусмотрели возможность аппаратной диагностики обрыва соединительной линии, либо замыкания цепи датчика. В этих случаях значение данных может быть равно или нижнему, или верхнему пределу диапазона (-3276.8...+3276.8), что является признаком неисправности цепи измерения , который можно опросить программно.
- Написать программу, опрашивающую значение соответствующего аналогового входа.

*Примечание: для того чтобы изменения вступили в силу, необходимо включить и выключить питание контроллера. Аналоговая фильтрация должна быть отключена.*

На рис. 9.11 показан текст программы проекта *RTD.MWP*. Программа производит вывод на панель оператора значение температуры в градусах Цельсия. Проект содержит текст программы и DATA-блок операторской панели TD200, сконфигурированной на выдачу единственного сообщения с внедренными данными значения температуры в формате с плавающей точкой (одна значащая цифра после запятой). Значения конфигурационных переключателей на модуле:

SW1...SW5 - 00100;

SW6 - 0;

SW7 - 0;

SW8 - 1.

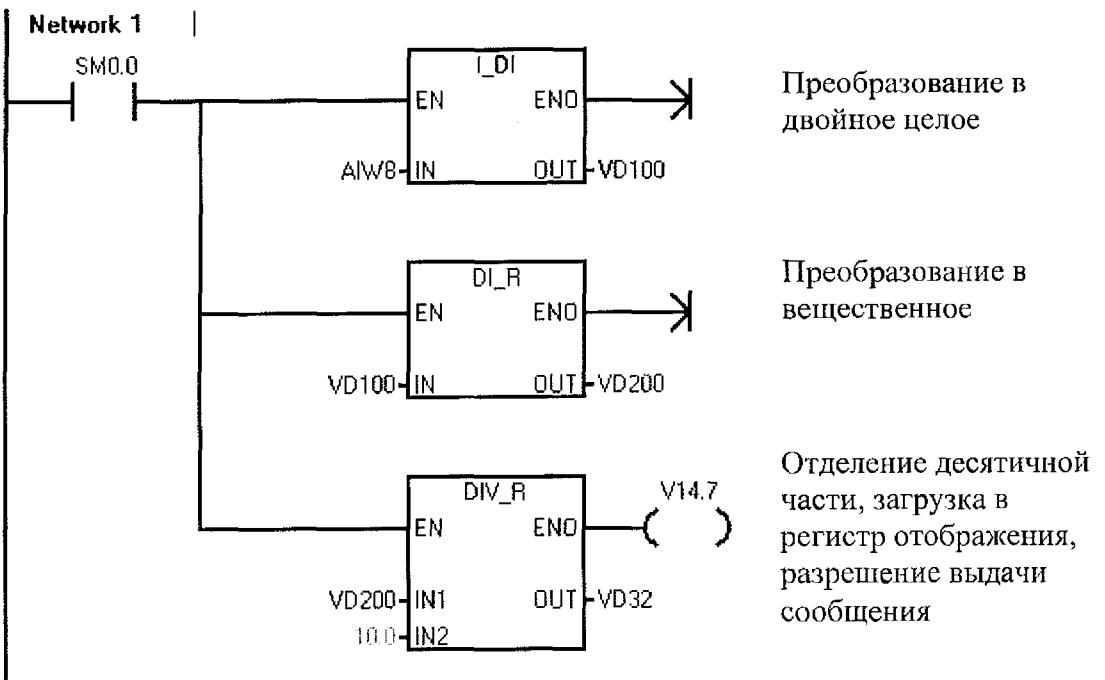


Рис.9.11. Текст программы вывода значения температуры в градусах

Примечание. Если в проекте не сконфигурирован вывод на панель оператора, значение температуры можно наблюдать в среде Step 7 Micro Win 32 в режиме «Program Status»

#### 9.4. Варианты заданий

Порядок выполнения работы:

- Подключить преобразователь ТСПУ-205 (рис. 9.10).
- Убедиться, что модуль EM231 подключен к контроллеру и встроенная диагностика индицирует его исправность.
- В среде Step7 MicroWin 32 открыть проект *TEMP.MCR*. Загрузить проект в контроллер и выполнить. Убедиться, что на панель оператора выводится значение температуры в градусах Цельсия.
- Согласно вариантам задания из табл. 9.2, выполнить следующие проекты на базе рассмотренного выше примера.

Таблица 9.2

## Варианты заданий

№	Задание	Отображение
1	Дополнить исходный проект переводом температуры в Кельвины	Градусы Цельсия, °C Кельвин, K
2	Дополнить исходный проект переводом температуры в градусы Фаренгейта	Градусы Цельсия, °C Фаренгейта, °F
3	Включить выход Q0.0, если температура превышает 31 градус. Выключить выход , если температура ниже 29 градусов	градусы Цельсия, °C
4	Измерить скорость нарастания температуры в диапазоне от 28 до 32 градусов. Использовать таймер милисекундных интервалов	T/мс
5	Измерить время, за которое температура возрастает от постоянной до заданной (напр. 30 градусов). Началом запуска счета является превышение исходной температуры на 5 % от исходной	время, с, мс
6	Кратковременно нагреть термопреобразователь. импульсным воздействием. Определить точку, когда значение температуры достигает максимума. При достижении включить Q0.0	
7	Измерить постоянную времени термопреобразователя. Диапазон температур – от комнатной до 30 градусов.	время, с
8	Вывести на панель оператора значение тока в цепи преобразователя в амперах. Контролировать обрыв цепи тока	ток, А
9	Вывести на панель оператора значение тока в цепи преобразователя в миллиамперах. Контролировать превышение тока за величину 15 мА	ток, мА
10	Вывести на панель оператора значение тока в цепи преобразователя в миллиамперах. Включать последовательно цифровые выходы через каждые 3 мА	ток, мА

Окончание табл. 9.2

№	Задание	Отображение
11	Вывести на панель оператора значение температуры. При достижении температуры верхней уставки включить выход Q0. Дополнительно выводить диагностические данные: Обрыв цепи датчика, отсутствие питания модуля	Градусы Цельсия, °C
12	Вывести на панель оператора значение температуры. Если значение долей градуса четное – включить выход Q1, если нечетное – Q2, дополнительно выводить аварийное сообщение об обрыве цепи датчика	Градусы Фаренгейта, °F
13	Выводить на панель оператора значение температуры в градусах Цельсия и Фаренгейта., при условии, что модуль сконфигурирован на градусы Фаренгейта. Выводить дополнительную информацию о типе модуля и количестве аналоговых входов	
14	Вывести на панель оператора значение температуры. Показывать процент изменения температуры по отношению к комнатной ( $20^{\circ}\text{C}$ ). Дополнительно выводить сообщение об обрыве цепи датчика	Градусы Цельсия, °C
15	Вывести на панель оператора значение температуры. Начиная с 25 градусов , включать выходы Q0 – Q8 , если температура увеличивается на $0.2^{\circ}\text{C}$ . Дополнительно выводить сообщение о направлении изменения температуры – рост или падение	Градусы Цельсия, °C

Примечание: для перевода температуры из градусов Фаренгейта в градусы Цельсия необходимо воспользоваться формулой

$${}^{\circ}\text{C} = \frac{({}^{\circ}\text{F} - 32) \cdot 5}{9}. \quad (9.7)$$

Для перевода температуры из градусов Цельсия в градусы Кельвина необходимо воспользоваться формулой

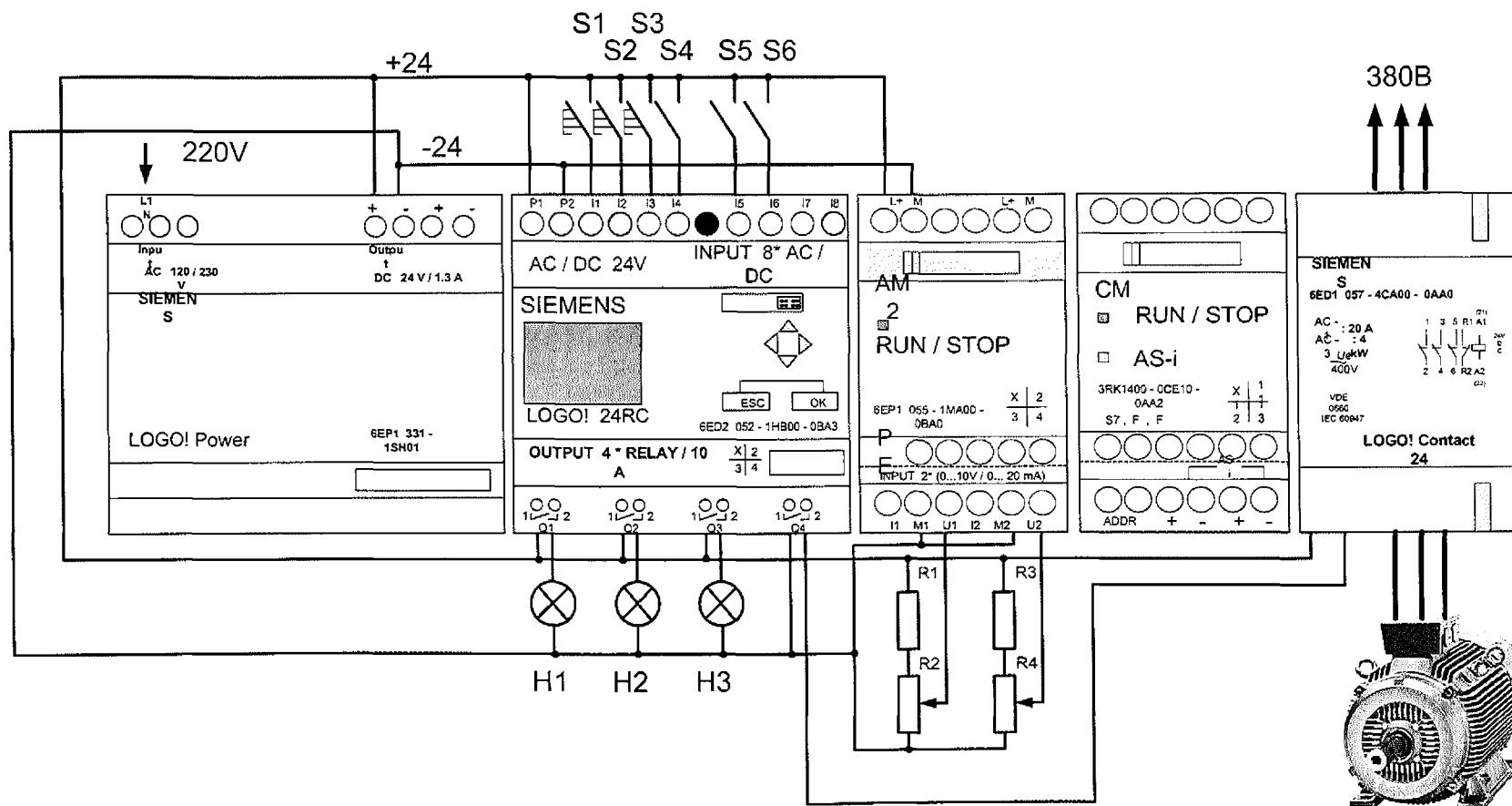
$$K = {}^{\circ}\text{C} + 273. \quad (9.8)$$

## БИБЛИОГРАФИЯ

1. Simatic LOGO!. Системное руководство. - 03/03. - A5E00119092.02. - Редакция 0.4. - М.: AO Siemens, 2008.- 308 с.
2. Программируемый контроллер S7-200. Системное руководство. C79000-G7076-C233. М.: AO Siemens.- 2003.- 909 с.

ПРИЛОЖЕНИЕ 1

**Схема лабораторного стенда LOGO!**



## ПРИЛОЖЕНИЕ 2

### Технические данные модуля EM235

<b>Модули ввода-вывода аналоговых сигналов EM 235 для CPU 22x</b>	
Съемные терминальные блоки	Есть
Количество входов	4, дифференциальных
Задержка от неправильной полярности входных сигналов	Нет
Пределы измерения	0...50мВ/ 0...100мВ/ 0...500мВ/ 0...1В/ 0...5В/ 0...10В/ 0...20mA/ ± 25мВ/ ± 50мВ/ ± 100мВ/ ± 250мВ/ ± 500мВ/ ± 1В/ ± 2.5В/ ± 5В/ ± 10В
Максимально допустимое напряжение на входе измерения напряжения	30В
Максимально допустимый ток входа измерения силы тока	32mA
Гальваническое разделение цепей	нет
Разрядность преобразования	12бит
Время аналого-цифрового преобразования	Не более 250мкс
Разрешение	Смотри руководство по программируемым контроллерам S7-200
Подавление шумов	40Дб, при =60В
· для частот	Нет
Максимальное синфазное напряжение	12В
Диапазон представления результатов преобразования:	
· для униполярных сигналов	0...32.000
· для биполярных сигналов	-32000...+32000
Линеаризация характеристик	Нет
Температурная компенсация	Нет
Диагностика	Светодиод EXTF (отсутствие напряжения питания)

Диапазоны изменения выходных сигналов:	
· напряжения	-10...+10В
· силы тока	0...20mA
Сопротивление нагрузки:	
· для выходного канала напряжения, не менее	5кОм
· для выходного канала силы тока, не более	0.5кОм
Гальваническое разделение цепей	Нет
Разрешение:	
· для канала напряжения	12бит
· для канала силы тока	11бит
Время цифро-аналогового преобразования:	
· для канала напряжения	100мкс
· для канала силы тока	2мс
Диапазон представления чисел:	
· для униполярных сигналов	0...32.000
· для биполярных сигналов	-32000...+32000
Рабочая погрешность преобразования (0...60°C по отношению к предельному значению выходного сигнала)	± 2%
Базовая погрешность преобразования (рабочая погрешность при 25°C по отношению к предельному значению выходного сигнала)	± 0,5%
Общие технические характеристики	
Потребляемый ток:	
· от внутренней шины контроллера (=5В)	30mA
· от источника питания датчиков или от внешнего блока питания =24В	60mA
Потребляемая мощность, типовое значение	2.0Вт
Габариты в мм	71.2 x 80 x 62
Масса	186г

## ПРИЛОЖЕНИЕ 3

### Технические данные модуля EM231 RTD

Подключение к CPU 222/224/226	
Съемные терминальные блоки	Есть
Количество входов	2 аналоговых
Типы датчиков	Pt 100, Pt 200, Pt 500, Pt 1000 ( $a = 3850$ ppm, 3920 ppm, 3850,55 ppm, 3916 ppm, 3902 ppm); Pt 10000 ( $a = 3850$ ppm); Cu 9.035 ( $a = 4270$ ppm); Ni 10, Ni 120, Ni 1000 ( $a = 6720$ ppm, 6178 ppm); R 150, 300, 600 Ом
Входное сопротивление	Не менее 10МОм
Максимально допустимое напряжение на входах	=30В (датчик) =5В (источник)
Гальваническое разделение цепей	Есть
· полевой уровень - цепи логики	~500В
· полевой уровень - цепи =24В	~500В
· цепи =24В - цепи логики	~500В
Время обновления информации	405мс (оба канала), 700мс для Pt 10000
Метод преобразования	Sigma-Delta
Разрешение:	15бит + знаковый разряд
· по температуре	0.1°C/0.1°F
· по сопротивлению	15бит + знаковый разряд
Подавление шумов:	85Дб
· для частот	50/60/400Гц
Синфазное напряжение	0
Минимальное отклонение синфазного сигнала	120Дб при ~120В
Диапазон изменения измеренных величин:	
· для биполярных сигналов	-27648...+27648
Базовая погрешность преобразования	0.1% FS (напряжение)

Повторяемость	0.05% FS
Диагностика	Светодиоды: EXTF (контроль напряжения питания), SF (системная ошибка)
Максимальная длина соединительного кабеля	100м на датчик
Сопротивление кабеля	20 Ом (до 2.7 Ом для меди)
Потребляемый ток:	
· от внутренней шины контроллера (=5В)	87mA
· от источника L+	60mA
Потребляемая мощность:	1.8Вт
· датчиком, не более	1mВт
Габариты в мм	71.2 x 80 x 62
Масса	210г

## Конфигурирование модуля ЕМ 235

Униполярный						Максимальный сигнал на входе	Разрешающая способность
SW1	SW2	SW3	SW4	SW5	SW6		
ВКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВКЛ	0 -50 мВ	12,5 мкВ
ВЫКЛ	ВКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВКЛ	0 -100 мВ	25 мкВ
ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВКЛ	0 -500 мВ	125 мкВ
ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВКЛ	0 -1 В	250 мкВ
ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	0 -5 В	1,25 мВ
ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	0 -20 мА	5 мА
ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	0 -10 В	2,5 мВ
Биполярный						Максимальный сигнал на входе	Разрешающая способность
SW1	SW2	SW3	SW4	SW5	SW6		
ВКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	+25 мВ	12,5 мкВ
ВЫКЛ	ВКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	+50 мВ	25 мкВ
ВЫКЛ	ВЫКЛ	ВКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	+100 мВ	50 мкВ
ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	+250 мВ	125 мкВ
ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	+500 мВ	250 мкВ
ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	+1 В	500 мкВ
ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	+2,5 В	1,25 мВ
ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	+5 В	2,5 мВ
ВЫКЛ	ВЫКЛ	ВКЛ	ВЫКЛ	ВЫКЛ	ВЫКЛ	+10 В	5 мВ

ПРИЛОЖЕНИЕ 5

**Конфигурирование модуля EM231 RTD**

**Выбор типа сенсора**

Тип RTD и Alpha	SW1	SW2	SW3	SW4	SW5
100 Ом Pt 0,003850 (по умолчанию)	0	0	0	0	0
200 Ом Pt 0,003850	0	0	0	0	1
500 Ом Pt 0,003850	0	0	0	1	0
1000 Ом Pt 0,003850	0	0	0	1	1
100 Ом Pt 0,003920	0	0	1	0	0
200 Ом Pt 0,003920	0	0	1	0	1
500 Ом Pt 0,003920	0	0	1	1	0
1000 Ом Pt 0,003920	0	0	1	1	1
100 Ом Pt 0,00385055	0	1	0	0	0
200 Ом Pt 0,00385055	0	1	0	0	1
500 Ом Pt 0,00385055	0	1	0	1	0
1000 Ом Pt 0,00385055	0	1	0	1	1
100 Ом Pt 0,003916	0	1	1	0	0
200 Ом Pt 0,003916	0	1	1	0	1
500 Ом Pt 0,003916	0	1	1	1	0
1000 Ом Pt 0,003916	0	1	1	1	1
100 Ом Pt 0,00302	1	0	0	0	0
200 Ом Pt 0,003902	1	0	0	0	1
500 Ом Pt 0,003902	1	0	0	1	0
1000 Ом Pt 0,003902	1	0	0	1	1
100 Ом Ni 0,00672	1	0	1	0	1
120 Ом Ni 0,00672	1	0	1	1	0
1000 Ом Ni 0,00672	1	0	1	1	1
100 Ом Ni 0,006178	1	1	0	0	0
120 Ом Ni 0,006178	1	1	0	0	1
1000 Ом Ni 0,006178	1	1	0	1	0
10000 Ом Pt 0,003850	1	1	0	1	1
10 Ом Cu 0,004270	1	1	1	0	0

150 Ом FS резистор	1	1	1	0	1
300 Ом FS резистор	1	1	1	1	0
600 Ом FS резистор	1	1	1	1	1

### Выбор предельного значения

Направление перегорания	SW6
К верхнему пределу шкалы(+3276,7 градусов)	0
К нижнему пределу шкалы -3276,8 градусов	1

### Выбор единиц измерения

Шкала	SW7
Цельсия ( $^{\circ}$ C)	0
Фаренгейта ( $^{\circ}$ F)	1

### Выбор схемы подключения

Схема подключения	SW8
3-проводная	0
2-проводная или 4-проводная	1

## Внешняя диагностика модуля EM231 RTD

Состояние ошибки	Данные канала	Светодиод SF	Светодиод 24V	Бит состояния диапазона	Бит неисправного состояния
Нет ошибок	Преобразованые данные	выкл	вкл	0	0
Отсутствие 24V	32766	выкл	выкл	0	1
Обрыв провода	-32768/32767	мигание	вкл	1	0
Входной сигнал за пределами диапазона	-32768/32767	мигание	вкл	1	0
Ошибка диагностики	0000	вкл	выкл	0	*

*Учебное издание*

**Иванов Виктор Эдуардович  
Конопелько Геннадий Константинович**

**ОСНОВЫ ПРОЕКТИРОВАНИЯ МИКРОСИСТЕМ НА БАЗЕ  
SIEMENS LOGO! И S7-200**

*Учебное пособие*

Дизайнер обложки И. Л. Тюкавкина  
Отпечатано с авторского оригинала – макета

Подписано в печать 22.12.2016. Формат 60x84 1/16.  
Бумага писчая. Гарнитура «Таймс» Печать цифровая.  
Усл. печ. л. 9,42. Тираж 500 экз. Заказ 342.

Издательство Тихоокеанского государственного университета.  
680035, Хабаровск, ул. Тихоокеанская, 136.  
Отдел оперативной полиграфии  
издательства Тихоокеанского государственного университета.  
680035, Хабаровск, ул. Тихоокеанская, 136.