



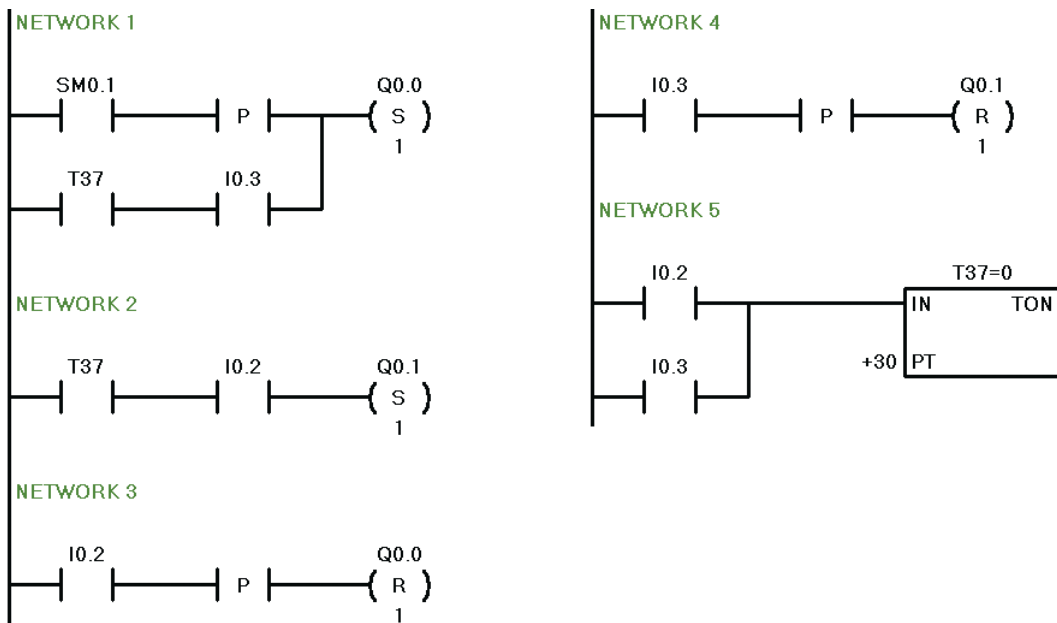
Уральский  
федеральный  
университет  
имени первого Президента  
России Б.Н.Ельцина

Уральский  
энергетический  
институт

**К. Е. НЕСТЕРОВ**  
**А. М. ЗЮЗЕВ**

# ПРОГРАММИРОВАНИЕ ПРОМЫШЛЕННЫХ КОНТРОЛЛЕРОВ

Учебно-методическое пособие





Министерство науки и высшего образования  
Российской Федерации  
Уральский федеральный университет  
имени первого Президента России Б. Н. Ельцина

К. Е. Нестеров  
А. М. Зюзев

# **ПРОГРАММИРОВАНИЕ ПРОМЫШЛЕННЫХ КОНТРОЛЛЕРОВ**

Учебно-методическое пособие

Рекомендовано методическим советом  
Уральского федерального университета  
для студентов вуза, обучающихся  
по направлению подготовки  
13.03.02 «Электроэнергетика и электротехника»

Екатеринбург  
Издательство Уральского университета  
2019

УДК 61.337.2:004.4(075.8)

ББК 31.264.34я73

Н56

Рецензенты:

кафедра электрификации горных предприятий ФГБОУ ВО «Уральский государственный горный университет» (завкафедрой д-р техн. наук, ст. науч. сотр. *А.Л. Карякин*);  
директор ООО «Тяжпромэлектропривод» канд. техн. наук, доц. *В.И. Зеленцов*

Научный редактор — канд. техн. наук, доц. *В.П. Метельков*

**Нестеров, К. Е.**

Н56 Программирование промышленных контроллеров : учеб.-метод. пособие / К. Е. Нестеров, А. М. Зюзев. — Екатеринбург : Изд-во Урал. ун-та, 2019. — 96 с. : ил.

ISBN 978-5-7996-2693-8

Пособие содержит основные сведения по системе программирования промышленных контроллеров серии Simatic S7-200, поясняет формат и назначение его основных команд. Здесь же рассмотрены примеры задач и предложены задания для самостоятельной работы, которые могут выполняться на специальных стендах, представляющих собой упрощенные макеты электроприводов и электрооборудования станков с ЧПУ, или на программных эмуляторах контроллера и объектов промышленной автоматики.

Издание ориентировано на современные технологии автоматизации с применением программируемых логических контроллеров и содержит оригинальный материал.

Библиогр.: 5 назв. Табл. 2. Рис. 15.

УДК 61.337.2:004.4(075.8)

ББК 31.264.34я73

ISBN 978-5-7996-2693-8

© Уральский федеральный  
университет, 2019

.....

# Оглавление

---

<b>1. Языки программирования контроллеров. Среда Step7-Micro/WIN.....</b>	<b>5</b>
1.1. Битовые логические команды .....	7
1.2. Операции сравнения .....	10
1.3. Счетчики и таймеры.....	11
1.4. Математические команды и преобразование типов переменных .....	15
1.5. Подпрограммы и прерывания .....	18
1.6. Указатели и косвенная адресация .....	25
1.7. Аналоговые входы и выходы, встроенные потенциометры ...	28
1.8. Цикл FOR-NEXT .....	30
1.9. Управление последовательностью операций.....	31
 <b>2. Программирование устройств автоматики на базе промышленных контроллеров.....</b>	 <b>34</b>
2.1. Структура комплекса для изучения систем управления промышленной автоматики.....	34
2.2. Задачи по теме «Автоматизация общепромышленных установок».....	38
2.2.1. Управление автоматизированными воротами.....	38
2.2.2. Электронное табло.....	38
2.2.3. Кодовый замок.....	43
2.2.4. Управление светофорами перекрестка .....	44
2.2.5. Система управления гирляндой .....	45
2.2.6. Автоматизация освещения .....	46
2.2.7. Автоматическое поддержание заданной температуры воды .....	48
2.2.8. Система автоматического поддержания температуры в помещении .....	49
2.2.9. Автоматизация дистиллятора .....	49
2.2.10. Система управления автоматизированной коробкой передач .....	50
2.2.11. Система управления башенными часами .....	50
2.2.12. Автоматизация насосной станции .....	51

.....

2.2.13. Система управления лифтом .....	52
2.2.14. Автоматизация установки для получения заданного количества смеси растворов требуемой температуры .....	53
2.2.15. Автоматизация установки для смешивания химических реактивов .....	54
2.2.16. Система управления установкой для получения смеси заданной температуры.....	55
2.2.17. Автоматизация крана-штабеллера .....	56
2.2.18. Автоматизация сверлильного станка .....	57
2.3. Задачи по теме «Электроавтоматика станков с ЧПУ».....	58
2.3.1. Автоматизация токарного станка.....	58
2.3.2. Автоматизация механизма смены инструмента .....	63
2.3.3. Управление механизмами участка механообработки...	69
2.4. Автоматизация робототехники .....	77
2.4.1. Цикловая система управления промышленным роботом МП-9С .....	80
2.4.2. Позиционная система управления роботом ТУР-10....	86
<b>Библиографический список .....</b>	<b>95</b>

# 1 Языки программирования контроллеров.

## 1. Среда Step7-Micro/WIN

Программное обеспечение Step7-Micro/WIN [4] предназначено для программирования и обслуживания логических контроллеров серии Simatic S7—200 фирмы Siemens. Среда разработки управляется операционной системой Windows и способна отображать программу в одном из трех видов: LAD (язык релейных схем), FBD (язык функциональных блоков) или STL (текстовый язык ассемблерного типа). Главное окно среды Step7-Micro/WIN по умолчанию содержит (рис. 1): 1 — кнопки выбора содержимого главного окна и вызова специализированных мастеров; 2 — таблицу переменных программы или параметров подпрограммы; 3 — дерево инструкций контроллера; 4 — поле для набора программы (подпрограммы); 5 — окно вывода сообщений компилятора.

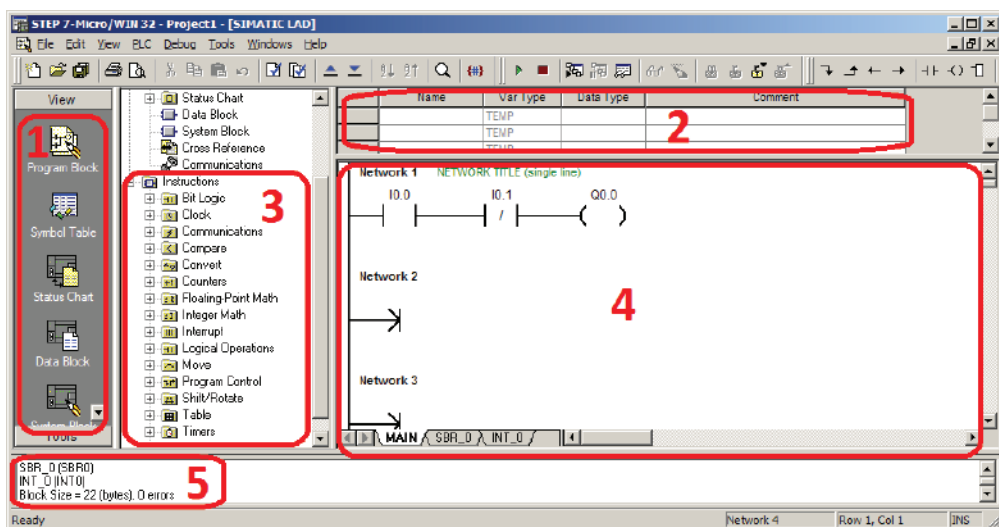


Рис. 1. Среда Step7-Micro/WIN

Программа для контроллера, показанная на рис. 1, представлена на языке Ladder. Среда позволяет изменить язык программирования (пункт меню View), даже если программа уже написана (некоторые программы, написанные на языке STL, нельзя перевести на другие языки). На рис. 2 приведена та же программа, представленная на других языках.

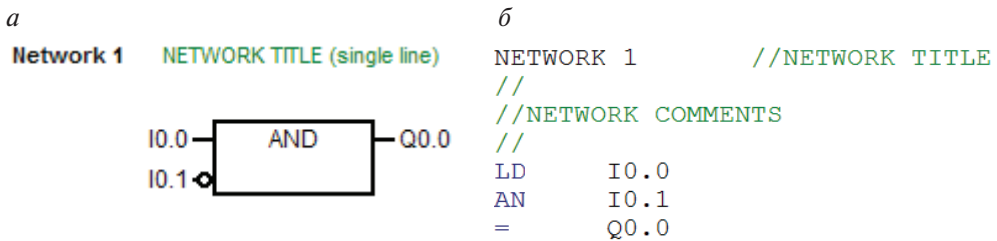


Рис. 2. Программа для контроллера на языках FBD (a) и STL (б)

Наиболее простым для понимания и наглядным является язык Ladder. Язык STL, хотя и позволяет создавать более эффективный в некоторых случаях код, является самым сложным и требует большего внимания при написании программ.

Среда разработки поддерживает абсолютную и символьную адресацию, позволяет создавать пользовательские подпрограммы и подпрограммы обработки прерываний, имеет встроенные средства для определения типа подключенного контроллера (от типа контроллера зависит набор доступных для программирования инструкций). Step7-Micro/WIN позволяет загружать программу из среды разработки в контроллер, выгружать ее из контроллера в среду разработки, может считывать и отображать значения переменных контроллера во время его работы.

Для подключения контроллера к компьютеру или программатору могут использоваться PC/PPI-кабель, USB/PPI-кабель, коммуникационные процессоры CP 5511 (PCMCIA-карта)/CP 5611 (PCI-карта) с MPI-кабелем, а также встроенные интерфейсы программаторов SIMATIC.

Среда Step7-Micro/WIN содержит набор мастеров для выполнения функций конфигурирования:

- текстового дисплея TD200;
- ПИД-регуляторов;
- коммуникационных соединений для обмена данными между центральными процессорами;



- скоростных счетчиков;
- модуля позиционирования EM253;
- модема EM 241;
- коммуникационного процессора CP 243–1.

Имеется встроенная система помощи по всем инструкциям контроллера, типам данных, ошибкам и возможностям среды разработки, поддерживается импорт и экспорт программ в формат AWL (используется при работе с эмулятором контроллера). Все данные проекта, включая программу, подпрограммы, параметры и настройки, сохраняются в одном файле, что упрощает перенос и копирование программ с одного компьютера на другой.

## 1.1. Битовые логические команды

---

Битовые команды предназначены для выполнения операций над переменными логического типа (принимающих одно из двух значений: 0 или 1), результатом их исполнения также является переменная логического типа.

- Команда **Normally Open** (нормально открытый контакт);
- Команда **Normally Close** (нормально закрытый контакт).

Эти команды получают значение из памяти или из регистра образа процесса, если типом данных является I или Q. В блоках AND (И) и OR (ИЛИ) можно использовать не более семи входов. Контакт Normally Open замкнут (включен), когда бит равен 1. Контакт Normally Close замкнут (включен), когда бит равен 0. В FBD команды, соответствующие нормально открытым контактам, представлены блоками AND/OR. Эти команды могут быть использованы для манипулирования булевыми сигналами таким же образом, как контакты LAD. Команды, соответствующие нормально замкнутым контактам, тоже представлены блоками AND/OR. Команда, соответствующая нормально замкнутому контакту, строится путем помещения символа отрицания на отметке входного сигнала. Количество входов блоков AND и OR может быть увеличено максимум до семи. В STL нормально открытый контакт представляется командами **LD** (load — загрузить), **A** (and — И) и **O** (or — ИЛИ). Эти команды загружают значение адресного бита в вершину стека или выполняют логическое сопряже-

ние значения адресного бита со значением в вершине стека в соответствии с таблицей истинности логического И или ИЛИ. В STL нормально замкнутый контакт представляется командами **LDN** (загрузить инверсное значение), **AN** (И-НЕ) и **ON** (ИЛИ-НЕ). Эти команды загружают логическое отрицание значения адресного бита в вершину стека или выполняют логическое сопряжение логического отрицания значения адресного бита со значением в вершине стека в соответствии с таблицей истинности логического И или ИЛИ.

- Команда **Output** (выход).

Когда выполняется команда **Output**, в регистре образа процесса устанавливается выходной бит. В LAD и FBD при выполнении команды **Output** указанный бит устанавливается равным потоку сигнала на входе команды **Output**. В STL команда **=** (выход) копирует вершину стека в указанный бит.

- Команда **Positive Transition** (положительный фронт);
- Команда **Negative Transition** (отрицательный фронт).

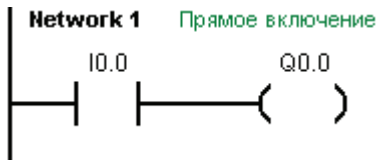
Команда **Positive Transition** пропускает поток сигнала в течение одного цикла при каждом появлении положительного фронта. Команда **Negative Transition** пропускает поток сигнала в течение одного цикла при каждом появлении отрицательного фронта. В LAD команды **Positive Transition** и **Negative Transition** представляются контактами. В FBD эти команды представляются блоками P и N. В STL команде **Positive Transition** соответствует команда **EU** (edge up). При обнаружении перехода значения в вершине стека с 0 на 1, значение в вершине стека устанавливается в 1; в противном случае оно устанавливается в 0. В STL команде **Negative Transition** соответствует команда **ED** (edge down). При обнаружении перехода значения в вершине стека с 1 на 0, значение в вершине стека устанавливается в 1, в противном случае оно устанавливается в 0.

- Команды **S** (set — установить) и **R** (reset — сбросить) для нескольких (N) разрядов.

Когда исполняется команда **S** или **R**, устанавливается (включается) или сбрасывается (выключается) указанное количество разрядов (N), начиная со значения, определенного параметром (бит) команды. Количество разрядов, которые могут быть установлены или сброшены, составляет 1...255. При использовании команды **R**, если указанный бит является битом таймера или счетчика, сбрасывается как таймер или счетчик, так и текущее значение таймера или счетчика.

**Задача 1.** Разработать программу, реализующую прямое управление выходом контроллера Q0.0 с помощью входа I0.0, т. е. выход Q0.0 должен повторять состояние входа I0.0.

Решение:

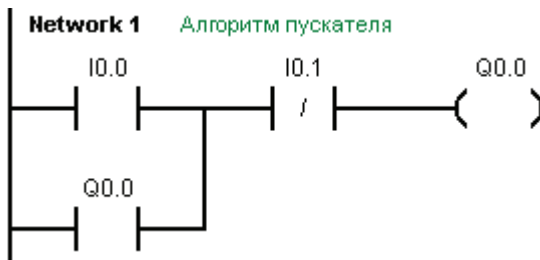


### Задание для самостоятельной работы

Модифицировать программу таким образом, чтобы вход I0.0 управлял не только выходом Q0.0, но и выходом Q0.1.

**Задача 2.** Разработать программу, реализующую алгоритм работы пускателя с кнопками Пуск и Стоп: импульс, поступивший на вход контроллера I0.0 (кнопка Пуск), включает выход Q0.0, а импульс на входе I0.1 (кнопка Стоп) отключает его.

Решение:



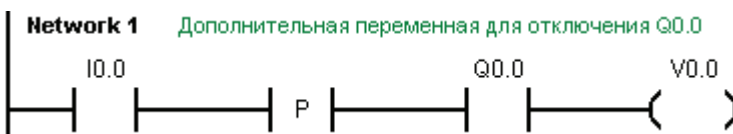
### Задание для самостоятельной работы

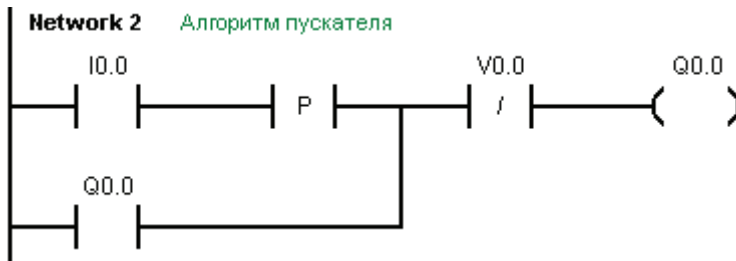
1. Добавить возможность управления выходом Q0.0 со второй кнопочной станции (вход I0.2 — Пуск, I0.3 — Стоп).

2. Решить задачу, используя триггерные команды S и R.

**Задача 3.** Разработать программу, управляющую выходом Q0.0 с помощью импульсов, поступающих с входа I0.0 (первый импульс включает выход, второй — выключает и т. д.).

Решение:





### Задание для самостоятельной работы

1. Добавить возможность управления выходом Q0.0 с входа I0.1.
2. Решить задачу, используя RS-триггер.
3. Создать программу, последовательно включающую выходы Q0.0 ...Q0.7 при появлении импульсов на входе I0.1.

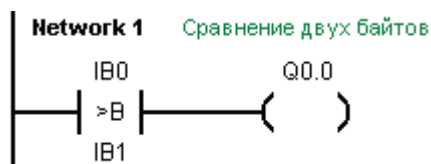
## 1.2. Операции сравнения

Операции сравнения могут осуществляться над переменными байтового (Byte), словного (двухбайтового Int), двухсловного (четырёхбайтового DInt) и вещественного (Real) типов. Для каждого типа сравниваемых переменных существует свой набор операций сравнения (больше, меньше, равно, не равно, больше или равно, меньше или равно), например, не допускается сравнение переменных словного типа при использовании операции сравнения байтов. Сравнивать между собой можно только переменные одного типа.

Команды сравнения используются для сравнения двух величин: IN1 и IN2. Возможны следующие сравнения:  $IN1 = IN2$ ,  $IN1 \geq IN2$ ,  $IN1 \leq IN2$ ,  $IN1 > IN2$ ,  $IN1 < IN2$  и  $IN1 \neq IN2$ . Байты сравниваются без знака. В LAD контакт включен, когда сравнение истинно. В FBD выход включен, когда сравнение истинно. В STL, если сравнение истинно, то эти команды загружают «1» в вершину стека или выполняют логическое сопряжение значения 1 со значением в вершине стека в соответствии с таблицей истинности для И или ИЛИ.

**Задача 4.** Разработать программу, выполняющую сравнение значений байтов IB0 и IB1. Если  $IB0 > IB1$ , то включить выход контроллера Q0.0. Примечание: младший бит имеет индекс 0 (для байта IB0 младший бит — I0.0).

Решение:



### Задание для самостоятельной работы

1. Дополнить программу так, чтобы включался выход Q0.1 при  $IB0 < IB1$ , а Q0.2 — при  $IB0 = IB1$ .
2. Разработать программу, выполняющую сравнение значений слов (двухбайтовых целочисленных переменных) VW0 и VW2. Результат сравнения отобразить аналогично предыдущей задаче (включить соответствующий выход контроллера).

## 1.3. Счетчики и таймеры

Счетчики предназначены для подсчета импульсов, поступающих на их вход. Существует 3 вида счетчиков: прямой, реверсивный и обратный.

Команда **CTU** (count up — прямой счет) увеличивает значение счетчика вплоть до максимального значения при появлении нарастающих фронтов сигнала на входе CU. Когда текущее значение (Sxxx) больше или равно предустановленному значению (PV), бит счетчика (Sxxx) устанавливается. Счетчик сбрасывается, когда включается вход сброса. Он прекращает счет при достижении PV.

Команда **CTUD** (count up/down — реверсивный счет) увеличивает значение счетчика при появлении нарастающих фронтов сигнала на входе CU. Она уменьшает значение счетчика при появлении нарастающих фронтов сигнала на входе CD. Когда текущее значение (Sxxx) больше или равно предустановленному значению (PV), бит счетчика (Sxxx) устанавливается. Счетчик сбрасывается, когда включается вход сброса.

Команда **CTD** (count down — обратный счет) уменьшает значение счетчика от предустановленного значения при появлении нарастающих фронтов сигнала на входе CD. Когда текущее значение равно

нулю, бит счетчика (Cxxx) включается. Счетчик сбрасывает свой бит (Cxxx) и загружает текущее значение предустановленным значением (PV), когда включается вход загрузки (LD). Обратный счет прекращается при достижении нуля. Область счетчиков Cxxx = C0...C255. В STL входу сброса STU соответствует значение, находящееся в вершине стека, а входу «Прямой счет» — значение, загруженное во второй уровень стека. В STL входу сброса STUD соответствует значение, находящееся в вершине стека, входу «Обратный счет» — значение, загруженное во второй уровень стека, а входу «Прямой счет» — значение, загруженное в третий уровень стека. В STL входу загрузки CTD соответствует вершина стека, а входу «Обратный счет» — значение, загруженное во второй уровень стека.

Таймеры используются для реализации временных задержек между некоторыми событиями, например, между поступлением сигнала на контроллер и включением его выхода. Таймеры различаются временным разрешением — минимально возможной задержкой, реализуемой с их помощью. По величине разрешения они разделены на три группы: 1, 10 и 100 мс. К первой группе относятся таймеры с адресами T32, T96; ко второй — T33...T36 и T97...T100; к третьей — T37...T63 и T101...T255. Далее рассмотрены следующие таймерные команды:

- **TON** (On-Delay Timer — таймер с задержкой включения);
- **TOF** (Off-Delay Timer — таймер с задержкой выключения).

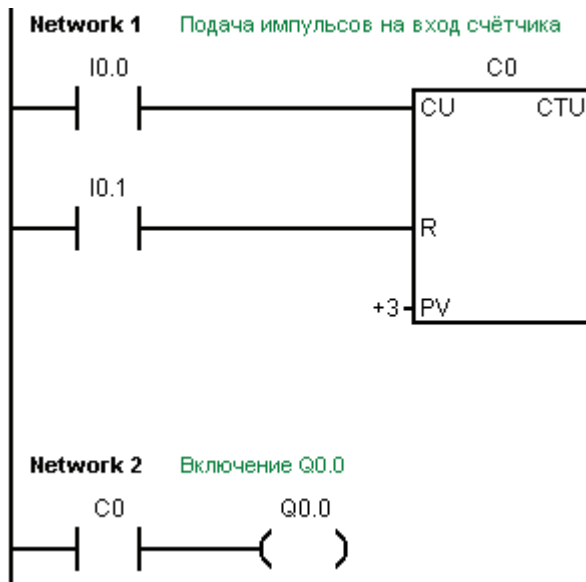
Команды **TON** (таймер с задержкой включения) и **TONR** (таймер с задержкой включения с запоминанием) отсчитывают время, когда включен разрешающий вход (IN). Когда текущее значение (Txxx) становится больше или равно предустановленному времени (PT), бит таймера устанавливается.

Текущее значение таймера с задержкой включения сбрасывается, когда выключается разрешающий вход, в то время как текущее значение таймера с задержкой включения с запоминанием сохраняется, когда этот вход выключается. Можно использовать таймер с задержкой включения с запоминанием для накопления времени за несколько периодов, когда включен разрешающий вход. Для стирания текущего значения таймера с задержкой включения с запоминанием, используется команда **R** (сбросить). Таймер с задержкой включения и таймер с задержкой включения с запоминанием продолжают счет после достижения предустановленного значения, они останавливают счет при достижении максимального значения, равного 32767.

**TOF** (таймер с задержкой выключения) используется для задержки выключения выхода на фиксированный интервал времени после выключения входа. Когда включается разрешающий вход, немедленно включается бит таймера, а текущее значение устанавливается в 0. Когда разрешающий вход выключается, таймер ведет отсчет времени, пока истекшее время не достигнет предустановленного времени. Когда предустановленное время достигнуто, бит таймера сбрасывается, а отсчет текущего значения прекращается. Команда **TOF** должна обнаружить переход от включенного состояния к выключенному, чтобы начать отсчет времени.

**Задача 5.** Создать программу, включающую выход Q0.0, если на вход I0.0 поступило 3 импульса.

Решение:

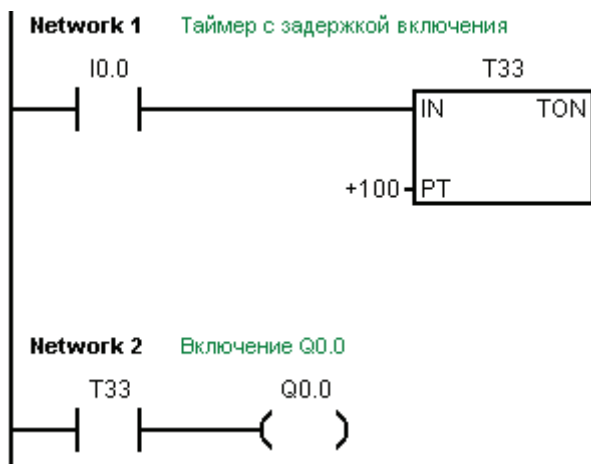


### Задание для самостоятельной работы

1. Модифицировать программу для подсчета 5 импульсов.
2. Изменить программу таким образом, чтобы при поступлении следующей группы импульсов включался очередной выход контроллера (Q0.1, Q0.2, ...), а предыдущий оставался во включенном состоянии. После включения выхода Q0.7 и поступления группы импульсов, все выходы контроллера должны выключиться, а система — вернуться в исходное состояние.

**Задача 6.** Разработать программу, включающую выход Q0.0 через 1 с после появления сигнала логической единицы на входе I0.0.

Решение:

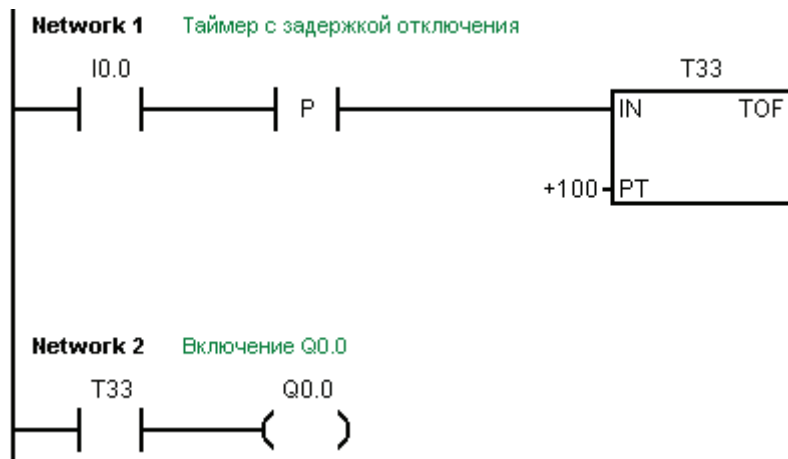


### Задание для самостоятельной работы

Увеличить время реакции контроллера до 3 с.

**Задача 7.** Создать программу, включающую выход Q0.0 на 1 с при появлении сигнала логической единицы на входе I0.0.

Решение:



### Задание для самостоятельной работы

1. Увеличить продолжительность включенного состояния выхода Q0.0 до 3 с.



2. Создать программу, обеспечивающую мигание выхода Q0.0 с частотой 1 Гц при нулевом сигнале на входе I0.0 и с частотой 2 Гц при единичном.

3. Используя таймеры и команды циклического сдвига **ROR** или **ROL** из группы инструкций Shift/Rotate, создать гирлянду «Бегущие огни».

## 1.4. Математические команды и преобразование типов переменных

Все математические команды разделены на два блока: команды для работы с целочисленными переменными — блок **Integer Math** и операции вещественной математики — блок **Floating-Point Math**.

Команды **INCB** (increment byte — увеличить байт на 1) и **DECB** (decrement byte — уменьшить байт на 1) прибавляют к входному байту (IN) или вычитают из него 1 и помещают результат в переменную, определенную параметром OUT. Операции увеличения и уменьшения байта на 1 являются беззнаковыми.

Команды **+I** (сложить целые числа) и **-I** (вычесть целые числа) складывают или вычитают два 16-битовых целых числа и дают 16-битовый результат (**OUT**).

В LAD и FBD:  $IN1 + IN2 = OUT$ .

$IN1 - IN2 = OUT$ .

В STL:  $IN1 + OUT = OUT$ .  $OUT - IN1 = OUT$

Команда **\*I** (умножить целые числа) перемножает два 16-битовых целых числа и дает 16-битовое произведение. Команда **/I** (разделить целые числа) делит два 16-битовых целых числа и дает 16-битовое частное.

В LAD и FBD:  $IN1 * IN2 = OUT$

$IN1 / IN2 = OUT$

В STL:  $IN1 * OUT = OUT$

$OUT / IN1 = OUT$

Команда **MOVb** пересылает входной байт (IN) в выходной байт (OUT). Команда **MOVw** пересылает входное слово (IN) в выходное слово (OUT). Команда **MOVD** пересылает входное двойное слово (IN) в выходное двойное слово (OUT). Команда **MOVr** пересылает входное вещественное число (двойное слово, 32 бита) (IN) в выходное (OUT). Входные переменные (IN) командами пересылки не изменяются.

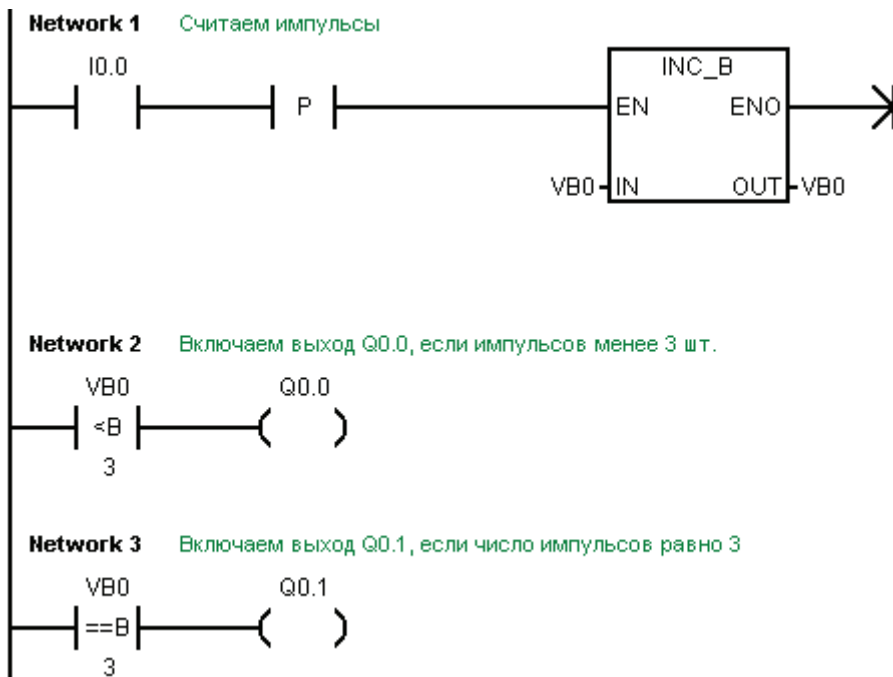
Команда **SQRT** извлекает квадратный корень из 32-битового вещественного числа (IN) и дает результат в виде 32-битового вещественного числа (OUT).

Команда **BTI** (byte to integer) преобразует байт (IN) в целое число и помещает результат в переменную, указанную в OUT. Байт не имеет знака, поэтому распространения знака не происходит. Команда **ITB** (integer to byte) преобразует слово (IN) в байт и помещает результат в переменную, указанную в OUT. Преобразуются значения от 0...255. Все остальные значения приводят к переполнению разрядной сетки и не влияют на выход OUT. Команда **DTR** (double to real) преобразует 32-битовое целое со знаком (IN) в 32-битовое вещественное число и помещает результат в переменную, указанную в OUT.

Команда **ROUND** (округлить) преобразует вещественное число (IN) в двойное целое число и помещает результат в переменную, указанную в параметре OUT. Если дробная часть равна 0,5 или больше, то число округляется в большую сторону.

**Задача 8.** Разработать программу, включающую выход Q0.0, если на вход I0.0 поступило менее 3 импульсов, и выход Q0.1, если количество поступивших импульсов равно 3.

Решение:

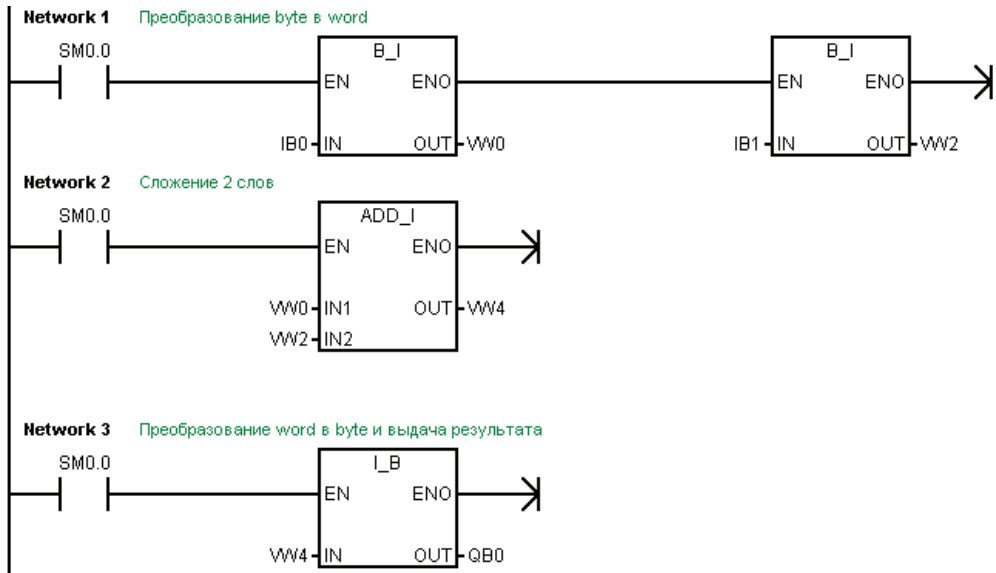


### Задание для самостоятельной работы

Дополнить программу так, чтобы при поступлении на вход I0.0 более 3 импульсов включался выход Q0.2.

**Задача 9.** Разработать программу, осуществляющую сложение байтов IB0 и IB1 с выдачей результата на выход QB0.

Решение:



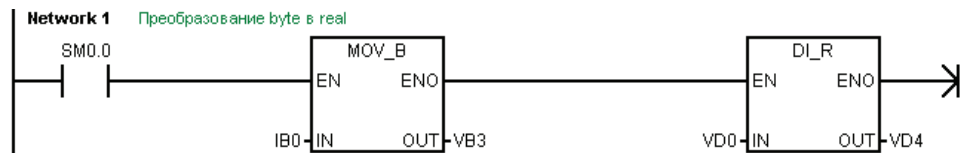
### Задание для самостоятельной работы

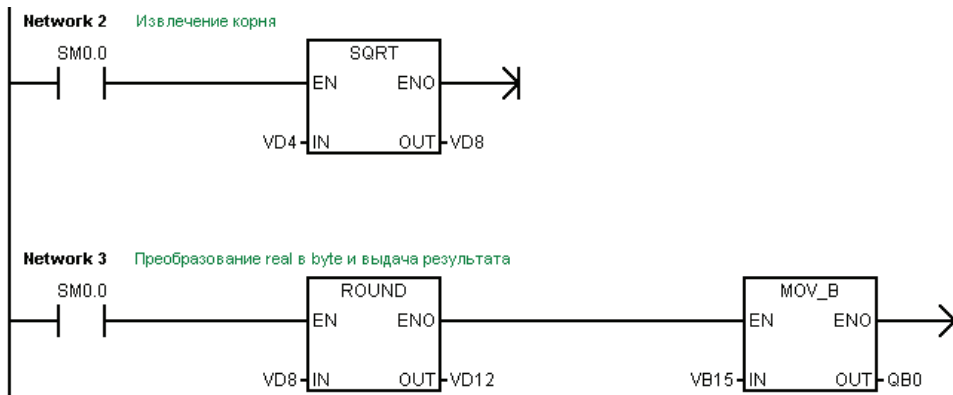
1. Модифицировать программу таким образом, чтобы она выполняла операцию умножения байтов IB0 и IB1.

2. Создать программу, выполняющую операцию деления двойных слов VD0 и VD4 с выдачей результата по адресу VW10 в виде 16-битной переменной.

**Задача 10.** Создать программу, реализующую операцию извлечения корня из байта IB0 с выдачей результата на выход QB0.

Решение:





### Задание для самостоятельной работы

Модифицировать программу таким образом, чтобы она выполняла операцию извлечения корня из слова IW0.

## 1.5. Подпрограммы и прерывания

В главной программе допускается вкладывать подпрограммы друг в друга (помещать вызов подпрограммы внутри другой подпрограммы) на глубину до 8 уровней. В программе обработки прерывания, вложение подпрограмм друг в друга невозможно. Подпрограмма не может быть помещена ни в какую другую подпрограмму, вызываемую из программы обработки прерывания. Рекурсия (вызов подпрограммы, вызывающей саму себя) разрешена.

Команда **CALL** (вызвать подпрограмму) передает управление подпрограмме. Команду **CALL** можно использовать с параметрами или без них. Команда **CRET** (условный возврат из подпрограммы) используется для завершения подпрограммы в зависимости от предшествующей логической операции. Как только исполнение подпрограммы завершается, управление возвращается команде, следующей за вызовом подпрограммы.

Подпрограмма может иметь или не иметь параметров. Существует три типа параметров: входные (in), проходные (in\_out) и выходные (out). Входные и проходные параметры должны быть инициализированы до передачи в подпрограмму. В качестве входных параметров мо-

гут использоваться константы или переменные (параметр-значение), проходные и выходные параметры должны быть переменными (параметр-переменная), т. к. они используются для передачи значений из подпрограммы. При изменении значения входного параметра, его новое значение не передается в вызвавший подпрограмму код.

Для добавления подпрограммы можно выбрать пункт меню Edit → Insert → Subroutine или в дереве проекта щелкнуть правой кнопкой мыши по разделу Program Block и выбрать пункт Insert Subroutine. Параметр подпрограммы добавляется путем вписывания его имени в таблицу переменных соответствующей подпрограммы и выбора типа параметра — бит, байт и т. д.

Существует несколько видов событий, для которых могут быть назначены прерывания. К ним относятся: нарастающий (спадающий) фронт сигнала на дискретных входах контроллера, циклические прерывания, прерывания от скоростных счетчиков и др.

Прежде чем программа обработки прерывания сможет быть вызвана, должно быть установлено соответствие между прерывающим событием и сегментом программы, который необходимо выполнить, когда это событие происходит. Для организации связи между прерывающим событием (задаваемым номером прерывающего события) и сегментом программы (задаваемым именем программы обработки прерывания) используется команда **ATCH** (attach — назначить). Команда **ATCH** связывает прерывающее событие EVNT с подпрограммой обработки прерывания INT и разблокирует прерывающее событие. Одной программе обработки прерываний можно поставить в соответствие несколько прерывающих событий, но одно событие не может быть одновременно поставлено в соответствие нескольким программам обработки прерываний. Когда происходит событие при разблокированных прерываниях, тогда исполняется только последняя программа обработки прерывания, поставленная в соответствие этому событию.

Когда программе обработки прерывания назначается прерывающее событие, это прерывание автоматически разблокируется. Если заблокировать все прерывания с помощью команды глобального блокирования прерываний, то каждое возникновение прерывающего события ставится в очередь, пока прерывания не будут снова разблокированы с помощью глобального разблокирования прерываний.

Отдельные прерывающие события можно заблокировать разрывом связи между этим прерывающим событием и программой обработ-

ки прерывания с помощью команды **DTCH** (detach — разорвать). Команда **DTCH** разрывает связь прерывающего события EVNT с подпрограммой обработки прерывания и блокирует прерывающее событие.

Команда **ENI** (enable interrupts — разблокировать прерывания) разблокирует обработку всех назначенных прерывающих событий. Команда **DISI** (disable interrupts — заблокировать прерывания) блокирует обработку всех прерывающих событий.

Когда контроллер переходит в режим RUN, прерывания первоначально заблокированы. Находясь в режиме RUN, можно разблокировать все прерывания, выполнив команду **ENI**. Команда **DISI** дает возможность ставить прерывания в очередь, но не позволяет вызывать подпрограммы обработки прерываний.

К прерываниям, управляемым по времени, относятся циклические прерывания и прерывания, вызываемые таймерами T32 и T96. С помощью циклических прерываний можно задать действия, которые должны выполняться циклически. Время цикла устанавливается в пределах 1...255 мс шагами по 1 мс. Всего может быть назначено до 2 прерываний, управляемых по времени. Для этого необходимо записать время цикла в миллисекундах в SMB34 для циклического прерывания № 1 или в SMB35 для циклического прерывания № 2. Событие, вызывающее циклические прерывания, передает управление соответствующей программе обработки прерываний каждый раз, как истекает время работы таймера. Обычно циклические прерывания используются для управления опросом аналоговых входов через регулярные интервалы времени или для организации работы ПИД-регулятора. Циклическое прерывание разблокируется, и начинается отсчет времени, когда назначена программа обработки прерывания событию, вызывающему циклическое прерывание. При этом система воспринимает значение времени цикла, и последующие изменения на это время цикла влияния не оказывают. Чтобы изменить время цикла, вы должны задать для него новое значение, а затем снова назначить программу обработки прерывания событию, вызывающему циклическое прерывание. Когда происходит повторное назначение, функция циклического прерывания сбрасывает все накопленное время от предыдущего назначения и начинает отсчет времени с новым значением.

Будучи разблокированным, циклическое прерывание работает постоянно, выполняя назначенную программу обработки прерывания при каждом истечении заданного временного интервала. Если вый-

ти из режима RUN или отсоединить циклическое прерывание, то оно заблокируется.

События, вызывающие прерывания, их коды и приоритеты перечислены в табл. 1.

Таблица 1

Список прерывающих событий, их кодов и приоритетов

Код события	Описание прерывания	Приоритет в группе
Группа приоритета — коммуникации (наивысшая)		
8	Порт 0: символ принят	0
9	Порт 0: передача завершена	0
23	Порт 0: прием сообщения завершен	0
24	Порт 1: прием сообщения завершен	1
25	Порт 1: символ принят	1
26	Порт 1: передача завершена	1
19	Прерывание при завершении РТО 0	0
20	Прерывание при завершении РТО 1	1
Группа приоритета — дискретные операции (средняя)		
19	Прерывание при завершении РТО 0	0
20	Прерывание при завершении РТО 1	1
0	Нарастающий фронт, I0.0	2
2	Нарастающий фронт, I0.1	3
4	Нарастающий фронт, I0.2	4
6	Нарастающий фронт, I0.3	5
1	Падающий фронт, I0.0	6
3	Падающий фронт, I0.1	7
5	Падающий фронт, I0.2	8
7	Падающий фронт, I0.3	9
12	HSC0: CV=PV (текущее значение = предустановленному)	10
27	HSC0: направление изменено	11
28	HSC0: внешний сброс	12
13	HSC1: CV=PV (текущее значение = предустановленному)	13
14	HSC1: Направление изменено	14
15	HSC1: внешний сброс	15
16	HSC2: CV=PV (текущее значение = предустановленному)	16
17	HSC2: направление изменено	17
18	HSC2: внешний сброс	18
32	HSC3: CV=PV (текущее значение = предустановленному)	19
29	HSC4: CV=PV (текущее значение = предустановленному)	20
30	HSC4: направление изменено	21

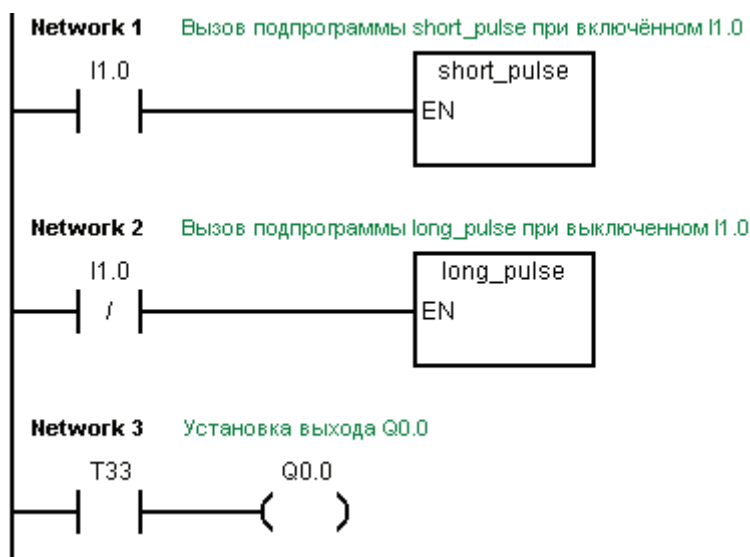
Окончание табл. 1

Код события	Описание прерывания	Приоритет в группе
31	HSC4: внешний сброс	22
33	HSC5: CV=PV (текущее значение = предустановленному)	23
Группа приоритета — управление временем (низшая)		
10	Циклическое прерывание 0	0
11	Циклическое прерывание 1	1
21	Прерывание от таймера T32 CT=PT	2
22	Прерывание от таймера T96 CT=PT	3

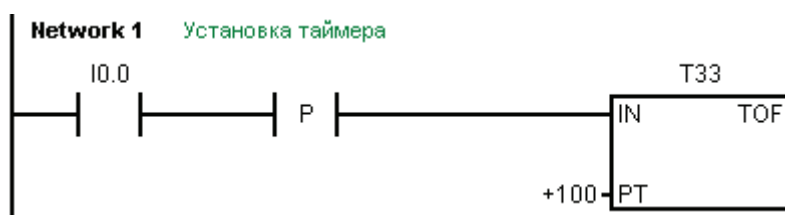
**Задача 11.** Разработать программу, включающую выход Q0.0 на 1 или 3 с (в зависимости от состояния входа I1.0) при появлении сигнала логической единицы на входе I0.0.

Решение:

код главной программы —

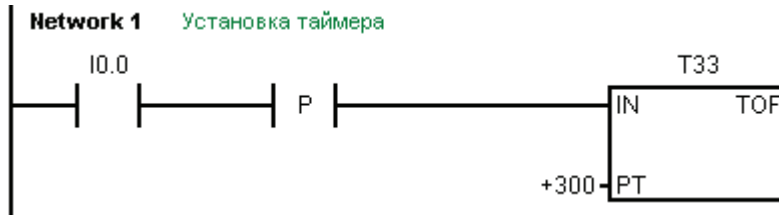


Код подпрограммы short\_pulse:





Код подпрограммы long\_pulse:



### Задание для самостоятельной работы

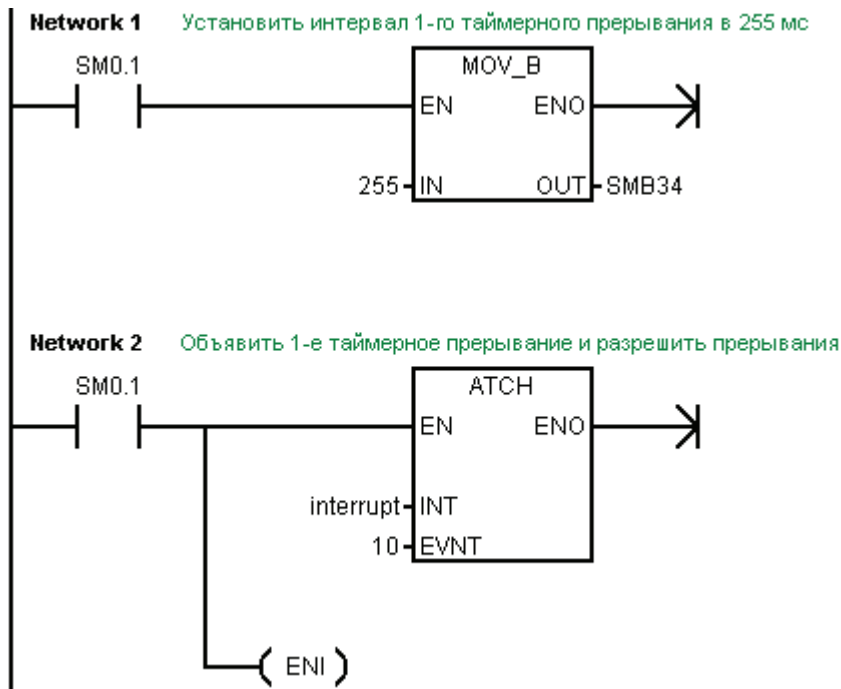
1. Изменить программу таким образом, чтобы вход контроллера I0.1 так же, как и вход I0.0, вызывал включение выхода Q0.0.

2. Избавиться от «лишней» подпрограммы, добавив в оставшуюся параметр, определяющий длительность импульса.

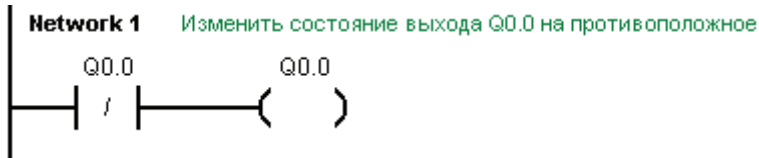
**Задача 12.** Разработать программу, осуществляющую мигание выхода Q0.0 с частотой 2 Гц (время включенного состояния выхода Q0.0 равно времени его отключенного состояния и составляет 250 мс).

Решение:

код главной программы —



Код подпрограммы обработки прерывания interrupt:



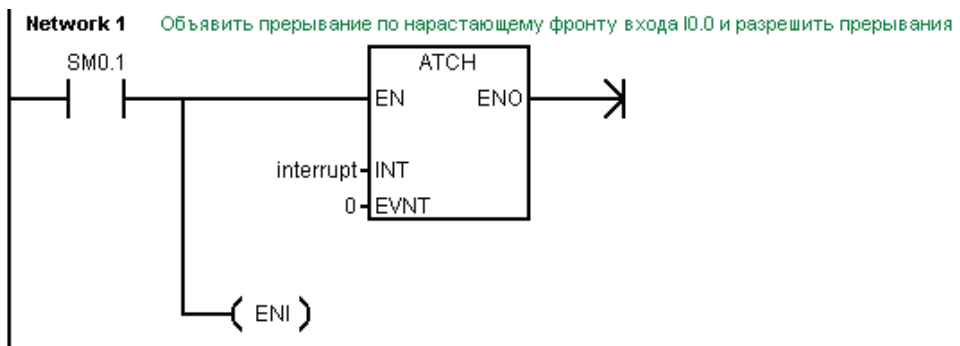
### Задание для самостоятельной работы

1. Найти опечатку, из-за которой частота мигания оказывается несколько меньше требуемой.
2. Добавить возможность переключения частоты мигания с помощью входа I0.0: если вход выключен, то частота мигания должна быть равна 2 Гц, если включен — 4 Гц.

**Задача 13.** Реализовать программный реверсивный счетчик импульсов, поступающих на вход I0.0, количество импульсов выдать на выход QV0. Направление счета (увеличение или уменьшение) должно определяться входом I0.1 (1 — увеличение, 0 — уменьшение).

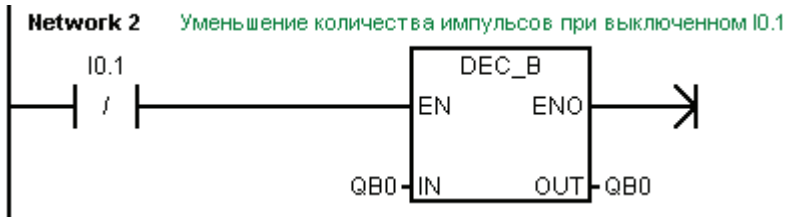
Решение:

код главной программы —



Код подпрограммы обработки прерывания interrupt:





### Задание для самостоятельной работы

1. Модифицировать программу так, чтобы она считала импульсы, поступающие с входа IO.2.
2. Решить эту же задачу, используя стандартный CTUD-счетчик.

## 1.6. Указатели и косвенная адресация

При косвенной адресации для доступа к данным в памяти используется указатель. CPU S7—200 дает возможность использования указателей для косвенной адресации следующих областей памяти: I, Q, V, M, S, T (только текущее значение) и C (только текущее значение). Косвенно адресовать можно переменные байтового, словного и двухсловного типов. Косвенную адресацию нельзя использовать для обращения к отдельному биту или аналоговым значениям.

Для косвенного обращения к адресу в памяти сначала необходимо создать указатель на этот адрес. Указатель — это ячейка памяти, имеющая размер двойного слова (4 байта), которая содержит адрес другой ячейки памяти. В качестве указателей можно использовать только ячейки V-памяти, L-памяти или аккумуляторные регистры (AC1, AC2, AC3). Для создания указателя нужно использовать команду **MOVD** (переместить двойное слово), чтобы переместить адрес косвенно адресованной ячейки памяти в ячейку указателя. При этом имени входной переменной должен предшествовать знак амперсанд (&), чтобы указать на необходимость перемещения в указатель адреса ячейки памяти, а не ее содержимого. Например, после выполнения операции  $AC1 = \&VB0$  указатель AC1 будет содержать адрес байта VB0, а не его значение.

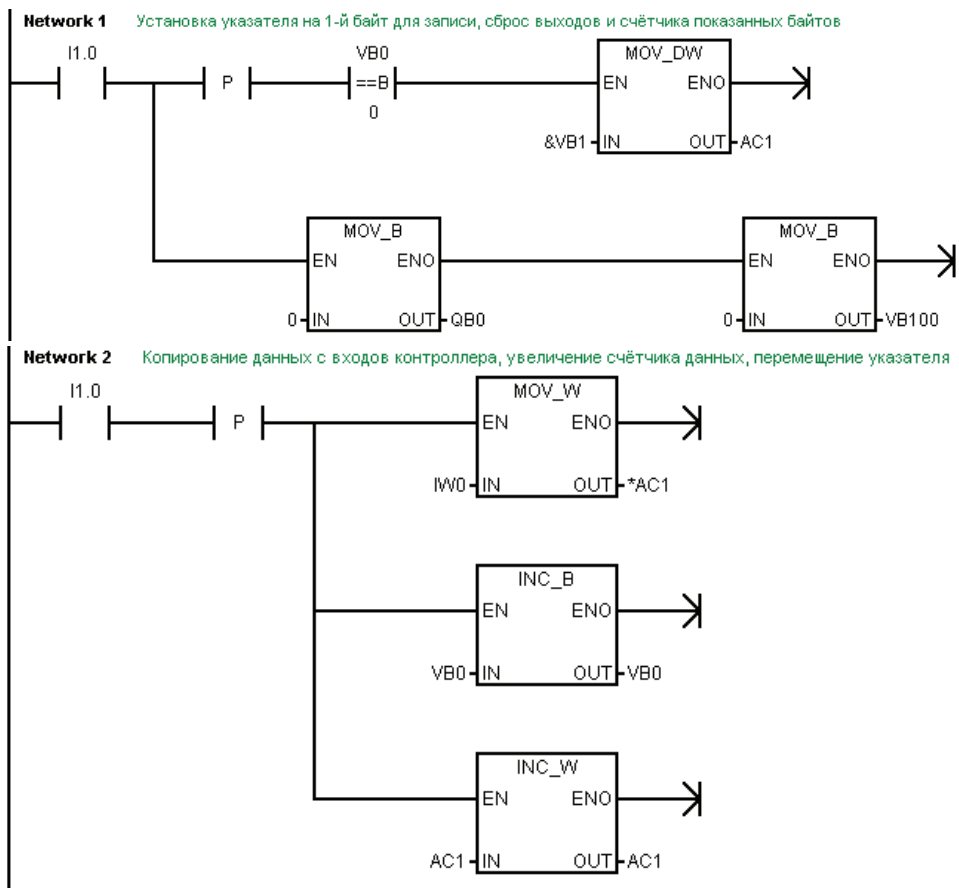
Ввод знака астериска (\*) перед именем переменной указывает, что эта переменная является указателем и хранит адрес ячейки памяти. На-

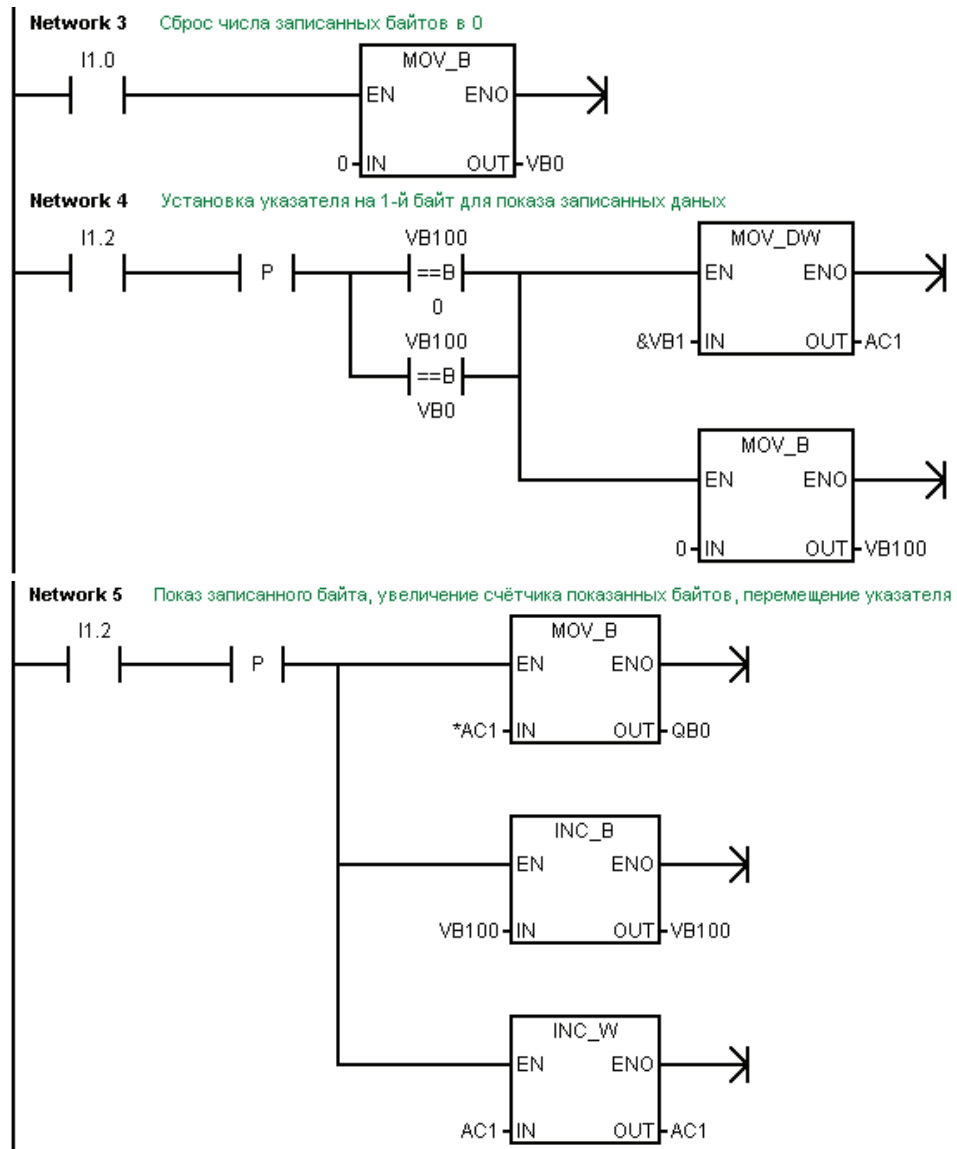
пример, после выполнения операции  $VB1 = *AC1$  переменной  $VB1$  будет присвоено значение переменной, на которую ссылается указатель  $AC1$ , а не ее адрес.

Значение указателя может быть изменено для того, чтобы перейти к другой ячейке памяти. Например, увеличение значения указателя на единицу приведет к тому, что он будет хранить адрес следующей ячейки памяти.

**Задача 14.** Разработать программу, запоминающую последовательность байтов, вводимых с входа  $IB0$  при появлении сигнала на  $IB1.0$ , и отображающую их на выходе  $QB0$  при появлении сигнала  $IB1.2$ . Для сброса запомненных данных использовать вход  $IB1.1$ .

Решение:





*Примечание:* представленное решение содержит опечатку, из-за которой оно неработоспособно.

### Задание для самостоятельной работы

1. Найти и исправить опечатку.
2. Модифицировать программу таким образом, чтобы в процессе ввода данных они отображались на выходе контроллера QB0.

3. Дополнить программу подсчетом среднего значения вводимых данных (байт IB0) и отображением его на выходе контроллера QB1.

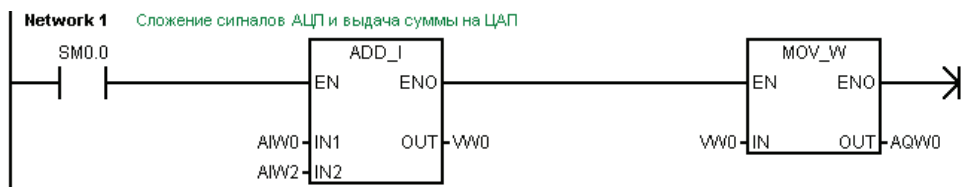
## 1.7. Аналоговые входы и выходы, встроенные потенциометры

Для считывания значения с аналоговых входов контроллера используются двухбайтовые переменные AIW0, AIW2, AIW4, и т. д., принимающие значения  $-32768 \dots +32767$ . Значения этих переменных соответствуют напряжению (току) на входе, масштабированному на весь диапазон напряжения (тока) входа. Например, если входы контроллера поддерживают диапазон напряжений  $0 \dots 10 \text{ В}$ , то значению  $10 \text{ В}$  на входе AIW0 будет соответствовать код  $32767$ . Аналоговые выходы масштабируются аналогично, а адресуются при помощи переменных AQW0, AQW2, AQW4 и т. д., также имеющих двухбайтовый формат.

В зависимости от модели, контроллеры серии Simatic S7–200 имеют до двух встроенных аналоговых потенциометров, положение которых может считываться и использоваться программой контроллера. Положение движков аналоговых потенциометров преобразуется в цифровое значение между 0 и 255 и заносится в байты специальной памяти SMB28 и SMB29, причем SMB28 содержит значения потенциометра № 1, а SMB29 — значение потенциометра № 2.

**Задача 15.** Создать программу, осуществляющую сложение двух аналоговых сигналов, поступающих на входы контроллера AIW0 и AIW2, с выдачей результата на выход AQW0.

Решение:

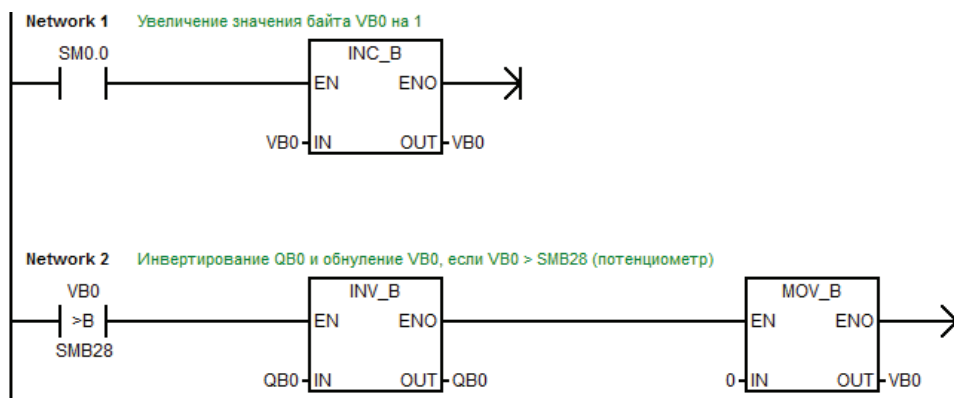


### Задание для самостоятельной работы

1. Модифицировать программу таким образом, чтобы выходной сигнал контроллера AQW0 соответствовал произведению сигналов, поступающих на входы AIW0 и AIW2.

3. Создать программу, масштабирующую сигнал, поступающий с входа AIW0, и выдающую результат на выход AQW0. Коэффициент масштабирования должен задаваться при помощи потенциометра в диапазоне 0,0...2,0.

Решение:



1. Добавить возможность включения — отключения мигания с помощью входа I0.0 (1 — разрешить мигание, 0 — запретить).

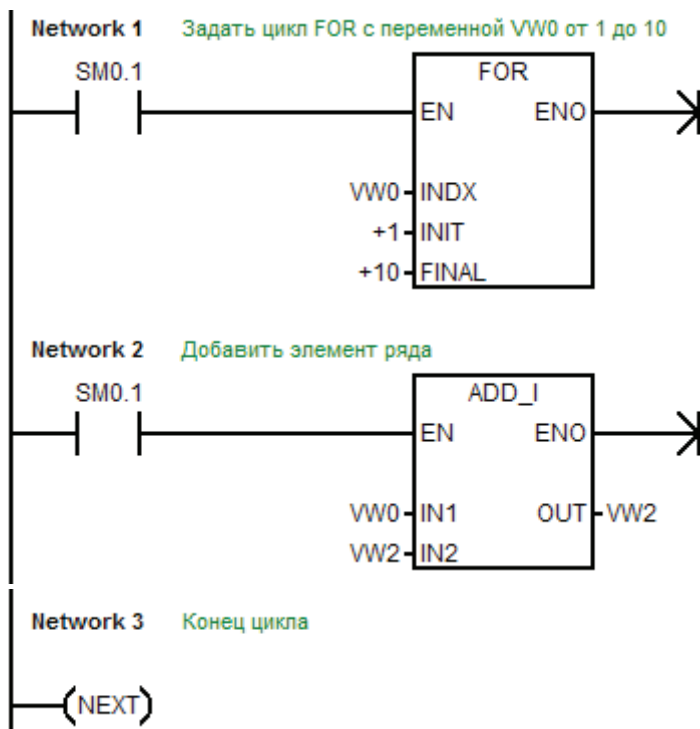
3. Используя выходы контроллера Q0.0 ...Q0.7, создать индикатор положения потенциометра № 1, функционирующий следующим образом: при коде переменной потенциометра менее 30 все выходы выключены, 30 ...60 — включен выход Q0.0, 60 ...90 — Q0.0 и Q0.1 и т.д. При коде более 240 все выходы включены.

## 1.8. Цикл FOR-NEXT

Команда **FOR** выполняет команды, расположенные между операторами FOR и NEXT. Необходимо задать значение индекса или счетчик цикла **INDX**, начальное значение **INIT** и конечное значение **FINAL**. Команда **NEXT** отмечает конец цикла FOR и устанавливает вершину стека в 1. Например, если значение **INIT** равно 1, значение **FINAL** равно 10, то команды между FOR и NEXT исполняются 10 раз, причем значение **INDX** каждый раз увеличивается на единицу: 1, 2, 3, ..., 10. Если начальное значение больше конечного, то цикл не выполняется. После каждого исполнения команд между FOR и NEXT значение **INDX** увеличивается, а результат сравнивается с конечным значением. Если **INDX** больше конечного значения, то цикл завершается.

**Задача 17.** Разработать программу, рассчитывающую сумму ряда  $1 + 2 + 3 + \dots + 10$ .

Решение:





**Задание для самостоятельной работы**

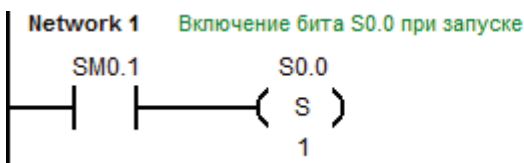
1. Изменить программу так, чтобы она рассчитывала сумму ряда  $1 + 2 + 3 + \dots + 300$ .
2. Изменить программу так, чтобы она рассчитывала сумму ряда  $1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{300}$ .
3. Изменить программу так, чтобы она рассчитывала сумму знакопеременного ряда  $1 - \frac{1}{2} + \frac{1}{3} - \dots - \frac{1}{300}$ .
4. Создать программу для расчета суммы бесконечного знакопеременного ряда  $1 - \frac{1}{2} + \frac{1}{3} - \dots \pm \frac{1}{\infty}$  без использования операторов цикла FOR-NEXT.

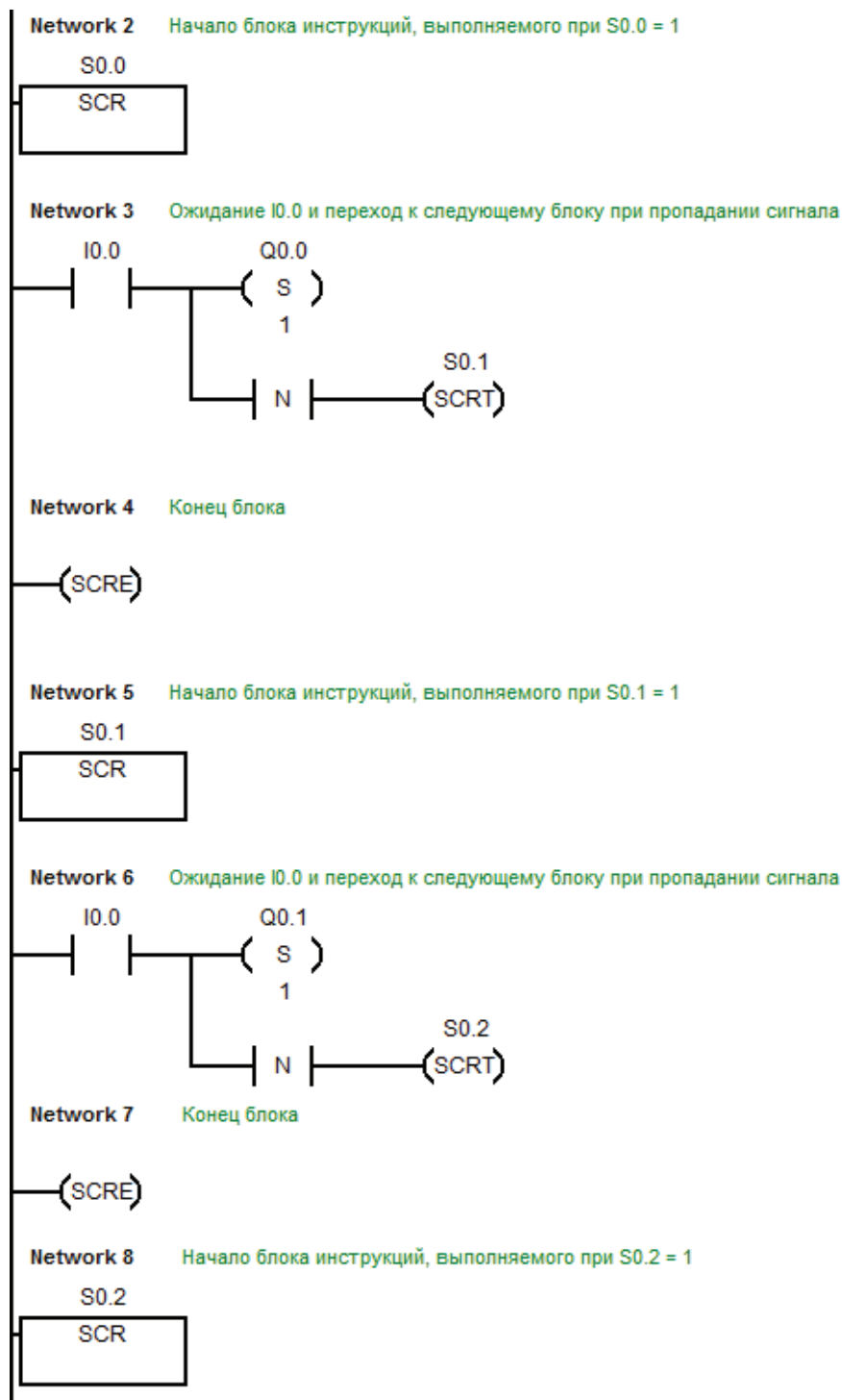
**1.9. Управление последовательностью операций**

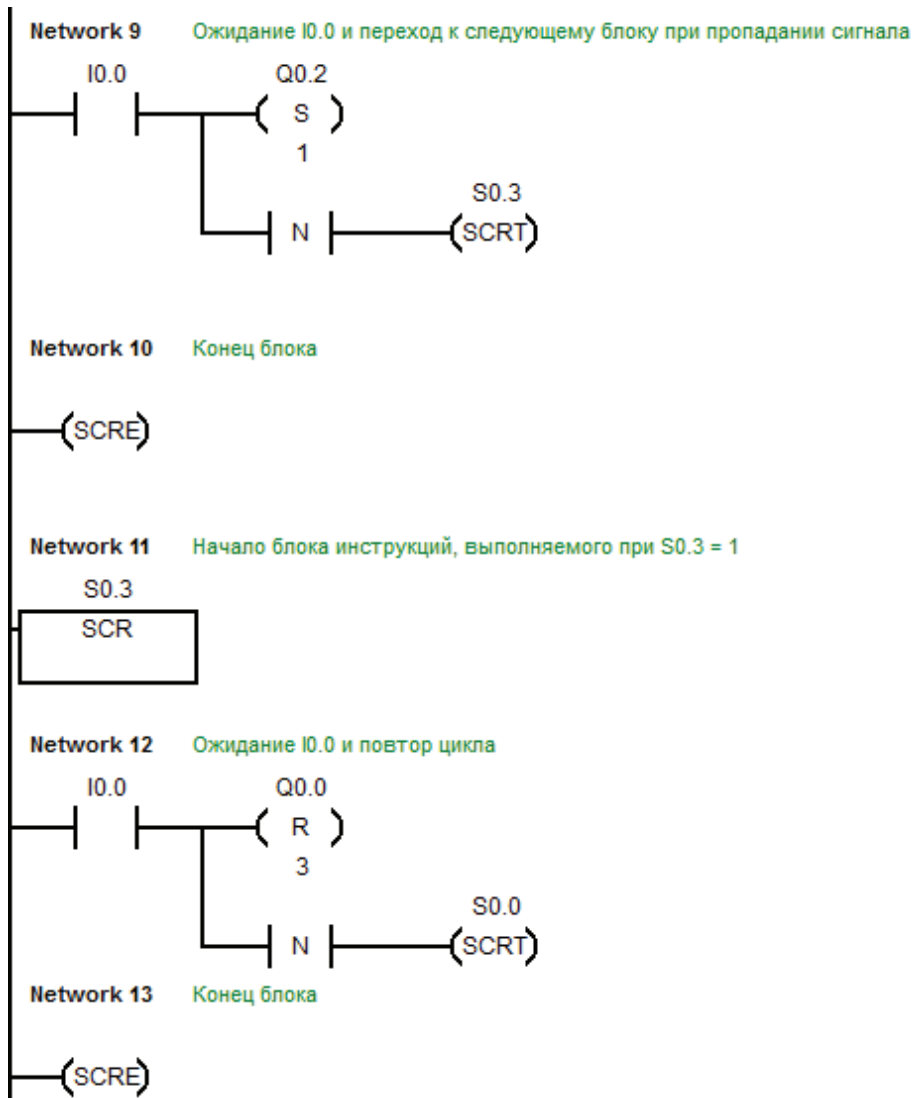
Команда **LSCR** (load sequence control relay — загрузить реле управления последовательностью) отмечает начало сегмента SCR. Когда  $n$  равно 1, поток сигнала пропускается к сегменту SCR. Сегмент SCR должен завершаться командой **SCRE**. Команда **SCRT** (sequence control relay to — перейти к следующему реле управления последовательностью) определяет бит SCR, который должен быть разблокирован (следующий S-бит, который должен быть установлен). Когда поток сигнала достигает катушки или блока FBD, S-бит, к которому производится обращение, устанавливается, а S-бит команды **LSCR** (который разблокировал этот сегмент SCR) сбрасывается. Команда **SCRE** (sequence control relay end — конец реле управления последовательностью) отмечает конец сегмента SCR.

**Задача 18.** Создать программу, последовательно включающую выходы Q0.0...Q0.2 по переднему фронту сигнала на входе I0.0.

Решение:







### Задание для самостоятельной работы

1. Исследуя работу программы, постройте реализованную в ней циклограмму.
2. По найденной циклограмме разработайте алгоритм и реализуйте его без использования инструкций для управления последовательностью операций, а только лишь на основе инструкций группы Bit Logic.
3. Решите исходную задачу при помощи счетчика.
4. Решите исходную задачу на основе прерывания.

---

## 2. Программирование устройств автоматки на базе промышленных контроллеров

---

### 2.1. Структура комплекса для изучения систем управления промышленной автоматки

---

**П**редлагаемый инструментально-программный методический комплекс включает в себя следующие компоненты:

- программу для ЭВМ «Эмулятор контроллера» [1];
- программы-имитаторы промышленного оборудования [2].

Основой комплекса является программа «Эмулятор контроллера» (далее — эмулятор). Эмулятор может использоваться как отдельно, так и совместно с остальными перечисленными программами. Программы-имитаторы оборудования предназначены для совместного использования с указанным эмулятором, они дополняют его. В результате этого значительно упрощается и улучшается понимание принципов работы технологического оборудования.

В работе указанных программ используются такие технологии, как Dynamic Data Exchange (DDE), Transmission Control Protocol/Internet Protocol (TCP/IP), OpenGL. Технология DDE применена для связи эмулятора контроллера с имитатором оборудования, TCP/IP — для связи эмулятора со SCADA-системой, OpenGL задействована для быстрой и качественной прорисовки трехмерных моделей технологического оборудования.

Эмулятор контроллера предназначен для моделирования работы программируемых логических контроллеров и может использоваться только в учебных целях. Эмулятор поддерживает программы, записанные в формате AWL. Преобразование написанных на языке Step7 программ в формат AWL осуществляется с помощью процедуры экс-

порта программы из среды Step7-MicroWIN (File → Export), при этом предварительно необходимо перейти к основной программе (закладка Main в главном окне Step7-MicroWIN), иначе процедура экспорта будет применена только к отображаемой подпрограмме.

Эмулятор поддерживает следующие функции Step7 (в скобках указано обозначение команд на языке STL):

- нормально открытый — закрытый контакт (LD, LDN, LDI, LDNI);
- выход (=, =I);
- логическое И (A, AI, AN, ANI);
- логическое ИЛИ (O, OI, ON, ONI);
- нарастающий/спадающий фронт (EU, ED);
- инверсия (NOT);
- триггеры (S, SI, R, RI);
- счетчики (CTU, CTD, CTUD);
- таймеры (TON, TONR, TOF);
- сравнение байтов (LDB=, LDB<>, LDB>, LDB<, LDB>=, LDB<=, AB=, AB<>, AB>, AB<, AB>=, AB<=, OB=, OB<>, OB>, OB<, OB>=, OB<=);
- сравнение слов (LDW=, LDW<>, LDW>, LDW<, LDW>=, LDW<=, AW=, AW<>, AW>, AW<, AW>=, AW<=, OW=, OW<>, OW>, OW<, OW>=, OW<=);
- сравнение двойных слов (LDD=, LDD<>, LDD>, LDD<, LDD>=, LDD<=, AD=, AD<>, AD>, AD<, AD>=, AD<=, OD=, OD<>, OD>, OD<, OD>=, OD<=);
- сравнение вещественных чисел (LDR=, LDR<>, LDR>, LDR<, LDR>=, LDR<=, AR=, AR<>, AR>, AR<, AR>=, AR<=, OR=, OR<>, OR>, OR<, OR>=, OR<=);
- логические команды над байтами, словами и двойными словами (INVB, INVW, INVD, ANDB, ORB, XORB, ANDW, ORW, XORW, ANDD, ORD, XORD);
- арифметические операции с целыми числами (+I, -I, \*I, /I, +D, -D, \*D, /D, MUL, DIV);
- арифметические операции с вещественными числами (+R, -R, \*R, /R, SQRT, SIN, COS, TAN, LN, EXP);
- команды преобразования типов (BTI, ITB, ITD, DTI, DTR, ROUND, TRUNC);
- команды пересылки (MOVB, MOVW, MOVD, MOVR);

- команды групповой пересылки (BMB, BMW, BMD);
- команды, выполняемые над логическим стеком (ALD, OLD, LPS, LRD, LPP, LDS);
- команды сдвига и циклического сдвига (SRB, SLB, SRW, SLW, SRD, SLD, RRB, RLB, RRW, RLW, RRD, RLD, SHRB);
- команды для управления программой (JMP, LBL, END, STOP, WDR, LSCR, SCRT, FOR, NEXT).

Допускается использование:

- до 16 байт дискретных входов (I) / выходов (Q) (3 байта дискретных входов и выходов отображаются в главном окне эмулятора и в окне модуля расширения, значения остальных можно увидеть и задать в таблице переменных);
- до 5200 байт памяти переменных (V);
- до 32 байт битовой памяти (маркеры) (M);
- до 32 байт памяти реле управления последовательностью операций (S);
- до 180 байт специальной памяти (SM) (эмулятор автоматически управляет значениями следующих битов: SM0.0, SM0.1, SM0.4, SM0.5, SM0.6, SM0.7; остальные биты могут быть установлены вручную в таблице переменных);
- до 64 байт локальной памяти (L);
- до 256 счетчиков (C) и таймеров (T) (таймеры имеют следующее разрешение по времени: T0, T32, T64, T96 — 1 мс; T1...T4, T33...T36, T65...T68, T97...T100 — 10 мс; T5...T31, T37...T63, T69...T95, T101...T255 — 100 мс);
- до 32 аналоговых входов (AIW)/выходов (AQW) (адреса с 0 по 62) (значения 5-ти аналоговых входов и выходов отображаются в окне модуля расширения, значения остальных можно увидеть и задать в таблице переменных);
- до 4 аккумуляторов (AC).

Правильная эмуляция циклических прерываний, таймеров и рядов SM0.4 и SM0.5 возможна только в непрерывном режиме работы и при достаточной мощности ЭВМ (для подавляющего большинства задач достаточным является ПК на базе процессора i9–9980XE).

Имеется два режима эмуляции таймеров (доступ через всплывающее меню, возникающее при щелчке правой кнопки мыши в главном окне):

- с высокой точностью — точно моделируется работа всех таймеров и циклических прерываний;

- со средней точностью — имитация циклических прерываний и таймеров разрешением 1 мс не будет соответствовать реальной работе при задании в них интервалов времени, некратных 10 мс.

Режим эмуляции таймеров со средней точностью предназначен для случаев, когда вычислительной мощности ЭВМ недостаточно для работы эмулятора в режиме с высокой точностью. Требования к мощности ЭВМ возрастают с увеличением сложности загруженной программы и числа открытых окон эмулятора (таблица переменных, модуль расширения, окно загруженной программы).

Для редактирования списка переменных, отображаемых в таблице, следует щелкнуть правой кнопкой мыши по таблице и воспользоваться соответствующими пунктами всплывающего меню для того, чтобы добавить в таблицу новые переменные или удалить из нее существующие.

Эмулятором поддерживаются прерывания от входов I0.0...I0.3, циклические и таймерные прерывания, приоритеты и очередь прерываний, подпрограммы с параметрами и без (глубина вложения до 8 уровней), рекурсия, косвенная адресация памяти (указатели), константы, записанные в десятичном, шестнадцатеричном или двоичном формате.

Эмулятор контроллера способен отрабатывать программу в одном из 3-х режимов (выбирается в главном окне): непрерывном, поцикловом, пошаговом. В первом случае программа работает непрерывно с момента нажатия кнопки «Старт». Во втором — после нажатия кнопки «Старт» отрабатывается один цикл, и эмулятор приостанавливает работу до следующего нажатия кнопки «Старт». В пошаговом режиме работы одно нажатие кнопки «Старт» приводит к отработке только одной команды программы, при этом выполняемая команда отображается в специальном поле.


Имитаторы оборудования предназначены для графического отображения оборудования или отдельных его узлов. Имитаторы имеют строку состояния, в которой выводятся сообщения о режимах работы оборудования и об авариях. Трехмерные имитаторы позволяют вращать и масштабировать изображение механизма при помощи мыши или клавиш курсора.

Порядок работы с комплексом следующий.

1. Разработать программу для контроллера в среде STEP 7/MicroWIN.
  2. Экспортировать программу, выбрав пункт меню File → Export.
- Экспорт следует выполнять при открытой закладке «Main», иначе бу-

дет экспортирована только отображаемая на экране подпрограмма, а не вся программа.

3. Запустить программу для ЭВМ «Эмулятор контроллера».

4. Загрузить в эмулятор экспортированную программу, нажав кнопку  и выбрав нужный файл или перетащив файл на главное окно эмулятора.

5. Запустить выполнение программы, нажав кнопку .

6. Запустить необходимый имитатор оборудования.

7. Убедиться в правильности работы программы для контроллера.

## 2.2. Задачи по теме

### «Автоматизация общепромышленных установок»

---

#### 2.2.1. Управление автоматизированными воротами

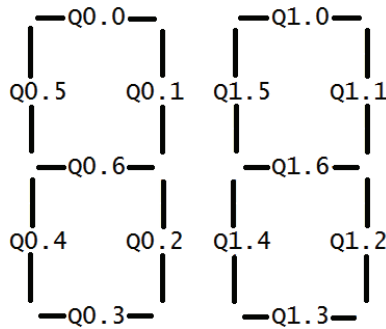
Автоматизированные ворота открываются и закрываются при помощи электропривода, пульт управления оборудован кнопками «Открыть», «Закрыть» и «Стоп». Для контроля положения ворот используются два датчика типа «конечный выключатель» — «Открыто» и «Закрыто».

**Задача.** Создать программу для управления автоматическими воротами: при нажатии кнопки Открыть (I0.0) должен включиться контактор Открыть (Q0.0), при нажатии кнопки Закрыть (I0.1) должен включиться контактор Закрыть (Q0.1). Контактор Открыть отключается при срабатывании конечного выключателя Открыто (I0.2), контактор Закрыть отключается при срабатывании конечного выключателя Закрыто (I0.3). Предусмотреть возможность остановки привода ворот при нажатии кнопки Стоп (I0.4) или повторного нажатия кнопок Открыть/Закрыть. Через 20 с после открывания, ворота должны автоматически закрыться, если перед этим не была нажата кнопка Стоп.

#### 2.2.2. Электронное табло

Схема подключения выходов контроллера к сегментам индикаторов табло показана ниже.





**Задача 1.** Разработать программу управления одной цифрой электронного табло. Двоичный код отображаемого числа (в диапазоне 0...9) должен приниматься контроллером на входы I0.0...I0.3. Остальные входы контроллера не должны влиять на работу табло. Если на вход подан код больший девяти, табло должно остаться пустым.

**Задача 2.** Разработать программу управления одной цифрой электронного табло. Двоичный код отображаемого числа (в диапазоне 0...9) должен приниматься контроллером на входы I0.0...I0.3. Остальные входы контроллера не должны влиять на работу табло. Если на вход подан код больший девяти, табло должно отобразить букву Е (error — ошибка).

**Задача 3.** Разработать программу управления одной цифрой электронного табло (шестнадцатеричный код). Двоичный код отображаемого числа (в диапазоне 0...F) должен приниматься контроллером на входы I0.0...I0.3. Остальные входы контроллера не должны влиять на работу табло.

**Задача 4.** Разработать программу управления двумя цифрами электронного табло. Двоичный код отображаемого числа (в диапазоне 0...99) должен приниматься контроллером на входы I0.0...I0.6. Остальные входы контроллера не должны влиять на работу табло. Если на вход подан код больший 99, табло должно остаться пустым.

**Задача 5.** Разработать программу управления двумя цифрами электронного табло. Двоичный код отображаемого числа (в диапазоне 0...99) должен приниматься контроллером на входы I0.0...I0.6. Остальные входы контроллера не должны влиять на работу табло. Если на вход подан код больший 99, табло должно отобразить букву Е (error — ошибка).

**Задача 6.** Разработать программу управления двумя цифрами электронного табло (шестнадцатеричный код). Двоичный код отображаемого числа (в диапазоне 0...FF) должен приниматься контроллером

на входы I0.0...I0.7. Остальные входы контроллера не должны влиять на работу табло.

**Задача 7.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...9. Если число импульсов превышает 9, на табло должна загореться буква Е. Остальные входы контроллера не должны влиять на работу табло.

**Задача 8.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...9. Если число импульсов превысило 9, счетчик должен сброситься в 0 и начать отсчет заново. Остальные входы контроллера не должны влиять на работу табло.

**Задача 9.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...F. Если число импульсов превысило 15, счетчик должен сброситься в 0 и начать отсчет заново. Остальные входы контроллера не должны влиять на работу табло.

**Задача 10.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...99. Если число импульсов превышает 99, на табло должна загореться буква Е. Остальные входы контроллера не должны влиять на работу табло.

**Задача 11.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...99. Если число импульсов превысило 99, счетчик должен сброситься в 0 и начать отсчет заново. Остальные входы контроллера не должны влиять на работу табло.

**Задача 12.** Разработать программу-счетчик импульсов, поступивших на вход I0.0. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...FF. Если число импульсов превысило 255, счетчик должен сброситься в 0 и начать отсчет заново. Остальные входы контроллера не должны влиять на работу табло.

**Задача 13.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...9. Если число импульсов превысило 9 или стало меньше 0, на табло должна загореться буква Е. Остальные входы контроллера не должны влиять на работу табло.

**Задача 14.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...9. Если число импульсов превысило 9, счетчик должен сброситься в 0 и начать отсчет заново. Если число импульсов стало равно -1, счетчик должен сброситься на 9. Остальные входы контроллера не должны влиять на работу табло.

**Задача 15.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться одной цифрой табло в диапазоне 0...F. Если число импульсов превысило 15, то счетчик должен сброситься в 0 и начать отсчет заново. Если число импульсов стало равно -1, счетчик должен сброситься на 15 (F). Остальные входы контроллера не должны влиять на работу табло.

**Задача 16.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...99. Если число импульсов превышает 99 или стало меньше 0, на табло должна загореться буква E. Остальные входы контроллера не должны влиять на работу табло.

**Задача 17.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...99. Если число импульсов превысило 99, счетчик должен сброситься в 0 и начать отсчет заново. Если число импульсов стало равно -1, счетчик должен сброситься на 99. Остальные входы контроллера не должны влиять на работу табло.

**Задача 18.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться двумя цифрами табло в диапазоне 0...FF. Если число импульсов превысило 255, счетчик должен сброситься в 0 и начать отсчет заново. Если число импульсов стало равно -1, счетчик должен сброситься на 255 (FF). Остальные входы контроллера не должны влиять на работу табло.

**Задача 19.** Разработать программу-счетчик импульсов, поступивших на входы I0.0 и I0.1, причем импульс на входе I0.0 увеличивает значение, а I0.1 — уменьшает. Число импульсов должно отображаться двумя

мя цифрами табло в диапазоне 0...99. Если число импульсов превышает 99 или стало меньше 0, на табло должна загореться буква Е. Импульс на входе I0.2 должен сбрасывать счетчик в 0. Остальные входы контроллера не должны влиять на работу табло.

**Задача 20.** Создать программу, считывающую сигнал датчика напряжения (AIW0) и отображающую значение в вольтах двумя цифрами на табло. Если значение отобразить нельзя, табло должно показывать букву Е.

**Задача 21.** Создать программу, считывающую сигнал датчика тока (AIW2) и отображающую значение в амперах двумя цифрами на табло. Величину тока нужно округлить до целого числа, например, если ток равен 1,4 А, то на табло должна быть цифра 1, если 1,6 — то 2.

**Задача 22.** Создать программу, считывающую сигнал датчика температуры (AIW4) и отображающую значение в градусах двумя цифрами на табло. Поскольку датчик температуры имеет нелинейную характеристику, предварительно требуется найти функцию, аппроксимирующую эту характеристику. Сделать это можно, пользуясь методами высшей математики или при помощи специализированного пакета (Matlab, MathCAD) или табличного редактора (Excel, функция аппроксимации данных). После нахождения аппроксимирующей функции ее следует реализовать на контроллере.

**Задача 23.** Создать программу, считывающую сигналы датчиков напряжения (AIW0) и тока (AIW2) и вычисляющую значение сопротивления цепи (Ом), которое нужно отобразить двумя цифрами на табло. Если значение отобразить нельзя, табло должно показывать букву Е.

**Задача 24.** Создать программу, считывающую сигналы датчиков постоянного напряжения (AIW0) и постоянного тока (AIW2) и вычисляющую значение мощности (Вт), которое нужно отобразить двумя цифрами на табло. Если значение отобразить нельзя, табло должно показывать букву Е.

**Задача 25.** Создать программу, считывающую сигнал датчика тока (AIW2) и отображающую значение в амперах двумя цифрами на табло. Величину тока нужно округлить до целого числа, например, если ток равен 1,4 А, то на табло должна быть цифра 1, если 1,6 — то 2. Если ток превышает 5 А, цифры на табло должны мигать с частотой 1 Гц.

**Задача 26.** Создать программу, считывающую сигнал датчика температуры (AIW4) и отображающую значение в градусах двумя цифрами на табло. Поскольку датчик температуры имеет нелинейную ха-

рактеристику, предварительно требуется найти аппроксимирующую эту характеристику функцию. Сделать это можно, пользуясь методами высшей математики или при помощи специализированного пакета (Matlab, MathCAD) или табличного редактора (Excel, функция аппроксимации данных). После нахождения аппроксимирующей функции ее следует реализовать на контроллере. Если значение температуры превышает  $25^{\circ}\text{C}$ , цифры на табло должны мигать с частотой 1 Гц.

**Задача 27.** Создать программу, считывающую сигнал импульсного датчика скорости (I0.0) и отображающую значение скорости в метрах в секунду одной цифрой на табло. Период измерения должен составлять 2 с, а величину скорости нужно округлить до целого числа, например, если скорость равна 1,4 м/с, то на табло должна быть цифра 1, если 1,6 — то 2.

**Задача 28.** Создать программу, считывающую сигнал импульсного датчика скорости (I0.0) и отображающую значение пройденного пути в метрах двумя цифрами на табло. При достижении максимального значения (99), табло должно сброситься в 0.

### 2.2.3. Кодовый замок

Входы контроллера соединены с кнопками кодового замка по следующей схеме: I0.0 — кнопка 1, I0.1 — 2, ..., I0.7 — 8, I1.0 — 9, I1.1 — \*, I1.2 — 0, I1.3 — #. Выход контроллера Q0.0 управляет электромагнитом замка таким образом, что при  $Q0.0 = 1$  замок открыт, а при  $Q0.0 = 0$  — закрыт.

**Задача 1.** Разработать программу управления кодовым замком, чтобы замок открывался при вводе правильного кода (например, 159). Ввод цифр кода должен осуществляться с паузой между нажатиями кнопок не более 2 с, иначе контроллер должен «забыть» введенные ранее цифры. Ввод частично правильного кода (например, 1159) не должен приводить к открыванию замка.

**Задача 2.** Дополнить программу управления замком счетчиком неудачных попыток открывания замка и блокировкой замка на 30 с после трех неудачных попыток открывания.

## 2.2.4. Управление светофорами перекрестка

Выходы контроллера управляют лампами двух светофоров, установленных на перекрестке и регулирующих движение в перпендикулярных направлениях. Если на одном из светофоров горит зеленый сигнал, то на другом в этот момент должен гореть красный. Кроме ламп контроллер управляет электронными табло, отображающими две цифры. Табло показывает оставшееся до переключения светофора время, измеренное в секундах.

**Задача 1.** Поочередно включая выходы контроллера, определить схему его подключения к светофорам и сегментам табло.

**Задача 2.** Разработать программу, управляющую двумя светофорами перекрестка. Последовательность включения сигналов светофоров:

Номер светофора	Время включения сигнала, с					
	0...20	20...23	23...25	25...45	45...48	48...50
1	Красный	Красный	Красный	Зеленый	Мигающий зеленый	Желтый
2	Зеленый	Мигающий зеленый	Желтый	Красный	Красный	Красный

**Задача 3.** Разработать программу, управляющую двумя светофорами перекрестка. Последовательность включения сигналов светофоров:

Номер светофора	Время включения сигнала, с					
	0...20	20...23	23...25	25...45	45...48	48...50
1	Красный	Красный	Желтый	Зеленый	Мигающий зеленый	Желтый
2	Зеленый	Мигающий зеленый	Желтый	Красный	Красный	Желтый

**Задача 4.** Разработать программу, управляющую двумя светофорами перекрестка. Последовательность включения сигналов светофоров:

Номер светофора	Время включения сигнала, с					
	0...20	20...23	23...25	25...45	45...48	48...50
1	Красный	Красный	Красный+желтый	Зеленый	Мигающий зеленый	Желтый
2	Зеленый	Мигающий зеленый	Желтый	Красный	Красный	Красный+желтый

**Задача 5.** Разработать программу, управляющую двумя светофорами перекрестка. Последовательность включения сигналов светофоров:

Номер светофора	Время включения сигнала, с							
	0...20	20...23	23...25	25...28	28...45	45...48	48...50	50...52
1	Красный	Красный	Красный	Красный + желтый	Зеленый	Мигающий зеленый	Желтый	Красный
2	Зеленый	Мигающий зеленый	Желтый	Красный	Красный	Красный	Красный	Красный + желтый

**Задача 6.** Разработать программу, управляющую двумя светофорами перекрестка. Последовательность включения сигналов светофоров:

Номер светофора	Время включения сигнала, с									
	0...20	20...23	23...25	25...28	28...30	30...45	45...48	48...50	50...52	52...55
1	Красный	Красный	Красный	Красный	Красный + желтый	Зеленый	Миг. зеленый	Желтый	Красный	Красный
2	Зеленый	Миг. зеленый	Желтый	Красный	Красный	Красный	Красный	Красный	Красный	Красный + желтый

### 2.2.5. Система управления гирляндой

Схема подключения гирлянды к контроллеру следующая: выходы контроллера с Q0.0 по Q3.7 подключены к лампам, а вход I0.0 — к кнопке переключения режимов работы.

**Задача 1.** Разработать программу, обеспечивающую равномерное мигание гирлянды с частотой 1 Гц.

**Задача 2.** Разработать программу, обеспечивающую мигание гирлянды с частотой 1 Гц в течение 10 с, потом — с частотой 2 Гц в течение 10 с, далее цикл повторяется (1 Гц).



**Задача 3.** Разработать программу, обеспечивающую мигание гирлянды. Частота мигания ламп должна переключаться нажатием кнопки (быстро — 2 Гц и медленно — 1 Гц).

**Задача 4.** Разработать программу, обеспечивающую мигание гирлянды в режиме «Бегущие огни».

**Задача 5.** Разработать программу, обеспечивающую мигание гирлянды в режиме «Бегущие огни». Скорость переключения ламп должна меняться при нажатии кнопки (2 скорости: быстро и медленно).

**Задача 6.** Разработать программу, обеспечивающую мигание гирлянды так, что ее цвета меняются по очереди: сначала горят все лампы красного цвета, потом — зеленого, затем — синего и т. д.

**Задача 7.** Разработать программу, обеспечивающую мигание гирлянды так, что ее цвета меняются по очереди: сначала горят все лампы красного цвета, потом — зеленого, затем — синего и т. д. Скорость переключения ламп должна меняться нажатием кнопки (2 скорости: быстро и медленно).

**Задача 8.** Создать гирлянду, поддерживающую 3 режима работы: мигание всех ламп с частотой 1 Гц, «Бегущие огни» и мигание разными цветами по очереди. Переключение режимов должно происходить при нажатии кнопки.

## 2.2.6. Автоматизация освещения

Система автоматического управления освещением содержит в своем составе контроллер с подключенными к нему двумя кнопками (кнопка 1 — I0.0, кнопка 2 — I0.1), датчиками (датчик освещенности — I0.2, датчик движения — I0.3) и контактором (Q0.0), подключающим группу мощных ламп к сети.

**Задача 1.** Разработать программу, управляющую освещением при помощи кнопок 1 и 2 следующим образом: кнопка 1 включает свет, кнопка 2 отключает.

**Задача 2.** Разработать программу, управляющую освещением при помощи кнопки 1 следующим образом: первое нажатие кнопки 1 включает свет, второе — отключает.

**Задача 3.** Разработать программу, управляющую освещением при помощи кнопок 1 и 2 следующим образом: нажатие любой кнопки переключает свет из включенного состояния в отключенное или наоборот.



**Задача 4.** Разработать программу, управляющую освещением на основе сигналов датчиков движения и освещенности следующим образом: в ночное время, при срабатывании датчика движения, включается свет, при выключении датчика движения свет горит еще 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. В дневное время система не реагирует на датчик движения.

**Задача 5.** Разработать программу, управляющую освещением при помощи кнопки 1 следующим образом: первое нажатие кнопки 1 включает свет, второе — отключает. Свет должен автоматически отключиться через 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. Если кнопка 1 удерживается в нажатом состоянии более 2 с, то свет не должен автоматически выключиться, в этом случае для отключения необходимо кратковременно нажать на кнопку 1.

**Задача 6.** Разработать программу, управляющую освещением при помощи кнопок 1 и 2 следующим образом: нажатие любой кнопки переключает свет из включенного состояния в отключенное или наоборот. Свет должен автоматически отключиться через 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. Если любая из кнопок удерживается в нажатом состоянии более 2 с, то свет не должен автоматически выключиться, в этом случае для отключения необходимо кратковременно нажать на любую кнопку.

**Задача 7.** Разработать программу, управляющую освещением при помощи кнопки 1 следующим образом: первое нажатие кнопки 1 включает свет, второе — отключает. Свет должен автоматически отключиться через 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. Если при включении света кнопка 1 нажимается дважды в течение 1 с, то свет не должен автоматически выключиться, в этом случае для отключения необходимо кратковременно нажать на кнопку 1.

**Задача 8.** Разработать программу, управляющую освещением при помощи кнопок 1 и 2 следующим образом: нажатие любой кнопки переключает свет из включенного состояния в отключенное или наоборот. Свет должен автоматически отключиться через 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. Если при включении света любая из кнопок на-

жимается дважды в течение 1 с, то свет не должен автоматически выключиться, в этом случае для отключения необходимо кратковременно нажать на любую кнопку.

**Задача 9.** Разработать программу, управляющую освещением при помощи кнопки 1 и на основе сигналов от датчиков движения и освещенности следующим образом: в ночное время, при срабатывании датчика движения, включается свет, при выключении датчика движения свет горит еще 10...35 с. Выдержка времени определяется встроенным потенциометром контроллера в диапазоне 10...35 с. В дневное время система не реагирует на датчик движения. Нажатие кнопки 1 переключает свет из включенного состояния в отключенное или наоборот независимо от датчика освещенности. При управлении кнопкой, свет не должен автоматически отключаться после выдержки времени.

**Задача 10.** Разработать программу, управляющую освещением при помощи кнопок 1 и 2 и на основе сигналов датчиков движения и освещенности следующим образом: в ночное время, при срабатывании датчика движения, включается свет, при выключении датчика движения свет горит еще 10...35 с. Выдержка времени в диапазоне 10...35 с определяется встроенным потенциометром контроллера. В дневное время система не реагирует на датчик движения. Нажатие любой кнопки переключает свет из включенного состояния в отключенное или наоборот независимо от датчика освещенности. При управлении кнопками свет не должен автоматически отключаться после выдержки времени.

### 2.2.7. Автоматическое поддержание заданной температуры воды

Система оснащена аналоговым датчиком температуры, подключенным к входу контроллера AIW0, кнопкой «Вскипятить» (I0.0), 4-х позиционным переключателем задания температуры (I0.1, I0.2) и нагревателем, который управляется выходом Q0.0 по принципу «Включен — выключен».

**Задача.** Разработать программу управления для системы поддержания температуры воды: температура должна поддерживаться на заданном переключателем уровне (входы I0.1, I0.2) с погрешностью в 2 °С. Предусмотреть режим «Кипячение», при котором вода нагревается

до 100 °С вне зависимости от положения переключателя задания и кипит 3 с. Отмена режима кипячения должна осуществляться повторным нажатием кнопки Кипячение.

### 2.2.8. Система автоматического поддержания температуры в помещении

Для обеспечения комфортных условий работы, температура в помещении должна поддерживаться на определенном уровне. Система автоматического поддержания температуры оснащена датчиком температуры, подключенным к аналоговому входу контроллера AIW0, и средствами нагрева и охлаждения: регулируемым по скорости вентилятором (подключен к аналоговому выходу AQW0) и нагревателем с возможностью регулирования мощности (подключен к выходу AQW2). Задание температуры в градусах Цельсия поступает в контроллер от SCADA, в ячейку памяти VW0.

**Задача.** Разработать программу управления системой поддержания температуры: температура должна поддерживаться при помощи нагревателя или вентилятора на уровне, заданном словом «VW0» (в градусах Цельсия). Статическая ошибка системы должна отсутствовать.

### 2.2.9. Автоматизация дистиллятора

Установка для дистилляции жидкостей состоит из нагревателя, который нагревает резервуар с жидкостью, конденсатора, охлаждаемого при помощи вентилятора, и бака готового продукта, в который из конденсатора поступает дистиллят. Нагреватель управляется выходом Q0.0 контроллера, вентилятор — Q0.1. Температура жидкости контролируется при помощи датчика, подключенного ко входу AIW2 контроллера. Датчик температуры имеет нелинейную характеристику. Пульт управления содержит 2 кнопки («Пуск» — I0.0 и «Стоп» — I0.1) и регулятор температуры жидкости AIW0.

**Задача.** Создать программу для контроллера, управляющего дистилляционной установкой. Установка должна работать следующим образом. Нажатие кнопки «Пуск» запускает процесс дистилляции: при помощи нагревателя температура жидкости поддерживается на уровне заданной с точностью ( $\pm 1$ ) °С. После нагрева жидкости до 60 °С вклю-

чается вентилятор конденсатора. Кнопка «Стоп» выключает нагреватель и вентилятор.

### 2.2.10. Система управления автоматизированной коробкой передач

Задачей системы управления коробкой передач является правильный выбор передачи в зависимости от положения рукоятки управления автоматической коробкой скоростей, положения педали акселератора и сигнала от датчика частоты вращения коленчатого вала двигателя или спидометра. В рассматриваемом примере аналоговые датчики подключены следующим образом: AIW0 — педаль акселератора; AIW2 — тахометр; AIW4 — спидометр. Положение рукоятки управления автоматической коробкой скоростей можно определить по двоичному коду, формируемому на входах контроллера I0.0, I0.1 и I0.2 (I0.0 — младший бит): 0 — положение P (парковка); 1 — положение R (движение назад); 2 — положение N (нейтральная передача); 3 — положение D (движение вперед); 4 — положение 2 (не включать передачи выше второй); 5 — положение 1 (движение на первой передаче). Управление коробкой скоростей контроллер осуществляет путем подачи сигналов на выходы с Q0.0 по Q0.7: Q0.0 — включить заднюю передачу; Q0.1 — включить первую передачу; Q0.2 — включить вторую передачу; Q0.3 — включить третью передачу; Q0.4 — включить четвертую передачу; Q0.5 — включить пятую передачу; Q0.6 — включить шестую передачу; Q0.7 — включить блокировку коробки. Одновременное включение нескольких выходов не допускается.

**Задача.** Разработать программу управления коробкой скоростей, учитывающую положение рукоятки управления автоматической коробкой скоростей, положение педали акселератора и сигнал от датчика частоты вращения коленчатого вала двигателя или спидометра.

### 2.2.11. Система управления башенными часами

Башенные часы имеют 3 стрелки: часовую, минутную и секундную. Перемещение стрелок осуществляется путем подачи импульса длительностью не менее 0,2 с на вход электропривода. За один импульс стрелка, в том числе и часовая, совершает перемещение, соответствующее

ющее  $6^\circ$ , т. е. одной минуте или секунде на циферблате. Стрелки могут вращаться как в направлении вперед, так и в направлении назад (для подстройки часов). Часовая стрелка управляется выходами Q0.0 (вперед) и Q0.1 (назад), минутная — Q0.2 и Q0.3, секундная — Q0.4 и Q0.5. Пульт подстройки часов содержит кнопки «+час» (I0.0), «–час» (I0.1), «+мин» (I0.2), «–мин» (I0.3), «+сек» (I0.4) и «–сек» (I0.5).

**Задача 1.** Разработать программу, управляющую стрелками часов без учета кнопок пульта управления.

**Задача 2.** Разработать программу, управляющую стрелками часов. Кнопки пульта управления должны использоваться для настройки часов.

### 2.2.12. Автоматизация насосной станции

На насосной станции установлены два одинаковых насоса, отличающихся только типом привода: первый насос подключается к электрической сети через устройство плавного пуска (УПП), а второй — через преобразователь частоты. Насосы включены параллельно в одну магистраль. Станция может работать в ручном или автоматическом режиме. В ручном режиме оператор может включать и отключать первый насос и регулировать скорость второго насоса. В автоматическом режиме контроллер должен поддерживать заданное значение давления на выходе станции, изменяя скорость второго насоса и включая или отключая первый насос.

**Задача.** Разработать программу для контроллера, управляющего станцией. Контроллер должен обеспечить работу станции в одном из двух режимов: ручном или автоматическом. Переключатель режимов работы подключен ко входу контроллера I0.0 (ручной — 0, автоматический — 1). В ручном режиме оператор должен иметь возможность управлять насосами: I0.1 — кнопка «Пуск УПП»; I0.2 — кнопка «Стоп УПП»; AIW0 — задание скорости второго насоса. В автоматическом режиме работы вход AIW0 используется как задание давления на выходе станции. Станция оборудована датчиком давления, подключенным ко входу AIW2. Выходы контроллера подключены следующим образом: Q0.0 — насос № 1 (вкл./выкл.); AQW0 — скорость насоса № 2 (0 ...32000).

### 2.2.13. Система управления лифтом

Электрооборудование лифта включает в себя два электропривода (подъема и дверей), кнопки вызовов и приказов, датчики, установленные в шахте лифта для контроля положения кабины, датчики дверей кабины и датчик наличия пассажиров в кабине. Кнопки вызовов, установленные на этажах, подключены ко входам контроллера I0.0...I0.3 (I0.0 — 1-й этаж). Кнопки приказов, установленные в кабине, подключены ко входам I0.4...I0.7 (I0.4 — 1-й этаж). Датчики, установленные в шахте лифта, подключены ко входам I1.0...I1.3 (I1.0 — 1-й этаж), I1.4 — датчик зоны точного останова. Датчики, установленные в кабине: I1.5 — двери кабины закрыты; I1.6 — двери кабины открыты; I1.7 — датчик наличия пассажира в кабине. Электропривод подъема лифта имеет две скорости — полную и пониженную. Выходы контроллера Q0.0 и Q0.1 задают направление движения кабины (вниз и вверх соответственно), при этом кабина движется на пониженной скорости. Если дополнительно к одному из перечисленных выходов включен выход Q0.2, то кабина будет двигаться на полной скорости. Датчики положения кабины, подключенные ко входам I1.0...I1.3, срабатывают при нахождении кабины в зоне пониженной скорости. Датчик зоны точного останова (I1.4) включается при нахождении кабины в зоне точного останова на любом этаже. Выходы Q0.3 и Q0.4 управляют приводом дверей кабины (закрыть и открыть).

Программа-имитатор лифта позволяет следить за положением кабины даже при закрытых дверях шахты лифта, для этого необходимо щелкнуть правой кнопкой мыши в окне программы и выбрать пункт Режим рентгена из всплывающего меню.

**Задача 1.** Разработать программу управления лифтом, обеспечивающую следующий алгоритм работы лифта.

При вызове лифта на этаж, он должен приехать, только если в кабине нет пассажиров, иначе команды кнопок вызовов должны игнорироваться. Кабина лифта должна перемещаться между этажами на полной скорости, при подходе к нужному этажу (срабатывает один из датчиков, подключенных ко входам I1.0...I1.3) лифт должен перейти на пониженную скорость, а при последующем срабатывании датчика точной остановки — остановиться и открыть двери. Привод дверей должен отключаться по сигналам конечных выключателей I1.5 и I1.6.

**Задача 2.** Дополнить решение задачи 1 обработкой всех попутных вызовов вниз: при движении вниз с 4-го этажа на 1-й и наличии сигналов вызовов на 2-м и 3-м этажах, кабина должна сделать попутные остановки на указанных этажах, после чего автоматически продолжить движение на 1-й этаж.

#### **2.2.14. Автоматизация установки для получения заданного количества смеси растворов требуемой температуры**

Установка для получения смеси имеет два мерных резервуара, оборудованных электроклапанами быстрого и медленного наполнения, аналоговыми датчиками уровня и электроклапанами слива в промежуточный резервуар для смешивания. Емкость каждого мерного резервуара составляет 10 л, емкость промежуточного резервуара — 25 л. Промежуточный резервуар опорожняется при помощи отдельного электроклапана, через который смесь поступает в накопительный бак. Для контроля наличия смеси в резервуаре для смешивания предусмотрен дискретный поплавковый датчик. Задание температуры готовой смеси и требуемого ее количества осуществляется при помощи потенциометров, подключенных ко входам контроллера AIW0 и AIW2. Датчики наполнения мерных резервуаров подключены ко входам AIW4 и AIW6, а датчик наличия смеси в резервуаре для смешивания — ко входу IO.2. Кнопки управления установкой подключены таким образом: «Пуск» — IO.0, «Стоп» — IO.1. Выходы контроллера управляют следующими аппаратами: Q0.0 — клапан медленного наполнения резервуара № 1; Q0.1 — клапан быстрого наполнения резервуара № 1; Q0.2 — клапан слива резервуара № 1; Q0.3 — клапан медленного наполнения резервуара № 2; Q0.4 — клапан быстрого наполнения резервуара № 2; Q0.5 — клапан слива резервуара № 2; Q0.6 — клапан слива промежуточного резервуара. Температура раствора, попадающего в первый мерный резервуар, составляет +90 °С, во второй — +20 °С.

**Задача.** Разработать программу, обеспечивающую работу установки следующим образом. При нажатии кнопки «Пуск», контроллер считывает заданную температуру смеси и требуемое ее количество и открывает электроклапаны быстрого наполнения мерных резервуаров. Для точного дозирования компонентов, в конце процесса наполнения ре-



резервуаров вместо электроклапанов быстрого наполнения открываются электроклапаны медленного наполнения. Далее срабатывают электроклапаны слива мерных резервуаров, после опустошения которых смесь сливается в бак готовой продукции. Если за один цикл смешивания не удастся получить заданного количества продукта, то должно быть выполнено несколько циклов, причем температура сливаемой в бак готовой смеси на каждом цикле должна быть равна заданной. Кнопка «Стоп» используется в качестве аварийной и отключает все электроклапаны.

### 2.2.15. Автоматизация установки для смешивания химических реактивов

Установка для смешивания химических реактивов имеет два мерных резервуара, оборудованных электроклапанами быстрого и медленного наполнения, аналоговыми датчиками уровня и электроклапанами слива в резервуар для смешивания, в котором установлен перемешивающий аппарат, приводимый в действие электроприводом. Емкость каждого мерного резервуара составляет 10 л, емкость резервуара для смешивания — 25 л. После смешивания реактивов срабатывает электроклапан слива резервуара для смешивания, и смесь поступает в накопительный бак. Для контроля над наличием смеси, в резервуаре для смешивания предусмотрен дискретный поплавковый датчик. Задание соотношения компонентов смеси и требуемого количества готового продукта осуществляется при помощи потенциометров, подключенных ко входам AIW0 и AIW2 контроллера. Датчики наполнения мерных резервуаров подключены ко входам AIW4 и AIW6, а датчик наличия смеси в резервуаре для смешивания — ко входу I0.2. Кнопки управления установкой подключены таким образом: «Пуск» — I0.0; «Стоп» — I0.1. Выходы контроллера управляют следующими аппаратами: Q0.0 — клапан медленного наполнения резервуара № 1; Q0.1 — клапан быстрого наполнения резервуара № 1; Q0.2 — клапан слива резервуара № 1; Q0.3 — клапан медленного наполнения резервуара № 2; Q0.4 — клапан быстрого наполнения резервуара № 2; Q0.5 — клапан слива резервуара № 2; Q0.6 — привод мешалки; Q0.7 — клапан слива резервуара для смешивания.

**Задача.** Разработать программу, обеспечивающую работу установки следующим образом.



При нажатии кнопки «Пуск», контроллер считывает заданное соотношение компонентов смеси и ее количество и открывает электроклапаны быстрого наполнения мерных резервуаров. Для точного дозирования компонентов, в конце процесса наполнения резервуаров вместо электроклапанов быстрого наполнения открываются электроклапаны медленного наполнения. Далее срабатывают электроклапаны слива мерных резервуаров и запускается электропривод механизма перемешивания. Через 10 с электропривод отключается, и полученная смесь сливается в бак готовой продукции. Если за один замес не удастся получить заданного количества смеси, то должно быть выполнено несколько замесов, причем соотношение компонентов смеси каждого замеса должно быть равно заданному. Кнопка «Стоп» используется в качестве аварийной, она останавливает электропривод установки и отключает электроклапаны.

#### **2.2.16. Система управления установкой для получения смеси заданной температуры**

Смесь заданной температуры получается из двух жидкостей: горячей и холодной. Оператор установки задает необходимую производительность установки в литрах в минуту и температуру смеси. Система управления контролирует температуры исходных жидкостей и, регулируя подачу каждой из них при помощи плавно управляемой задвижки, обеспечивает нужный расход и температуру смеси.

Задание температуры готовой смеси и требуемого расхода осуществляется при помощи потенциометров, подключенных ко входам AIW0 и AIW2 контроллера. Датчики температуры исходных жидкостей подключены ко входам AIW4 и AIW6. Управление задвижками осуществляется выходами AQW0 и AQW2 контроллера.

**Задача 1.** Автоматизировать работу установки, полагая, что температуры исходных жидкостей постоянны и составляют +90 °С (горячая) и +20 °С (холодная).

**Задача 2.** Автоматизировать работу установки с учетом того, что температуры исходных жидкостей могут меняться в процессе работы установки. Температура смеси должна поддерживаться равной заданной с точностью ( $\pm 1$ )°С.

### 2.2.17. Автоматизация крана-штабеллера

Кран-штабеллер используется на складах для перевозки грузов из (в) ячейки стеллажей. Кран оборудован тремя основными механизмами: механизмом горизонтального перемещения, вертикального перемещения и горизонтального перемещения захвата для транспортировки груза с (на) кран. Для повышения производительности крана и обеспечения требуемой точности его перемещений, механизмы горизонтального и вертикального перемещения могут работать на полной и пониженной скоростях. Контроль положения механизмов крана осуществляется при помощи путевых и конечных выключателей. В рассматриваемом примере кран и пульт управления им подключены к контроллеру по следующей схеме: I0.0 — переключатель режима работы «Ручной»/«Автоматический» (в ручном режиме I0.0 = 0). В ручном режиме работы используются кнопки «Вперед» — I0.1, «Назад» — I0.2, «Вверх» — I0.3, «Вниз» — I0.4, «Выдвинуть захват» — I0.5, «Задвинуть захват» — I0.6. В автоматическом режиме работы входы контроллера используются для других целей: I0.1 — «Привезти груз»; I0.2 — «Отвезти груз»; входы с I0.3 по I0.5 формируют код ряда (I0.3 — младший разряд); с I0.6 по I1.0 — код этажа (I0.6 — младший разряд). Путевые и конечные выключатели подключены к следующим входам контроллера: I1.1 — датчик перехода на пониженную скорость при движении по рельсам; I1.2 — датчик точного останова при движении по рельсам; I1.3 — датчик перехода на пониженную скорость при движении по вертикали; I1.4 — датчик точного останова при движении по вертикали; I1.5 — конечный выключатель «Захват задвинут»; I1.6 — конечный выключатель «Захват выдвинут». Для управления механизмами крана предназначены выходы контроллера: Q0.0 — вперед на пониженной скорости; Q0.1 — назад на пониженной скорости; Q0.2 — нормальная скорость привода горизонтального перемещения; Q0.3 — вверх на пониженной скорости; Q0.4 — вниз на пониженной скорости; Q0.5 — нормальная скорость привода вертикального перемещения; Q0.6 = 0 — задвинуть захват; Q0.6 = 1 — выдвинуть захват.

**Задача.** Разработать программу для управления краном-штабеллером, удовлетворяющую следующим требованиям.

В ручном режиме оператор управляет приводами крана так: электроприводы горизонтального и вертикального перемещения работают на пониженной скорости при нажатии и удержании соответствующей

кнопки, гидропривод механизма захвата управляется кратковременным нажатием кнопки пульта. В автоматическом режиме оператор при помощи переключателей «Ряд» и «Этаж» выбирает позицию на стеллаже и кратковременно нажимает на кнопку «Привезти» или «Отвезти», после чего кран автоматически выполняет команду.

Для того чтобы кран забрал груз со стола, он должен выполнить следующие действия:

- остановиться напротив стола (по датчику точной остановки по горизонтали) с опущенным и задвинутым захватом;
- опустить захват до «съезда» с датчика точной остановки по вертикали (ниже уровня стола);
- выдвинуть захват;
- поднять захват, «проехав» датчик точной остановки по вертикали (выше уровня стола);
- задвинуть захват.

Аналогично выполняется погрузка — выгрузка груза на стеллаж.

### 2.2.18. Автоматизация сверлильного станка

Сверлильный станок может работать в ручном или автоматическом режиме работы.

В ручном режиме оператор программирует станок методом обучения: управляя приводами осей  $X$  и  $Y$  при помощи кнопок  $X+$ ,  $X-$ ,  $Y+$ ,  $Y-$ , выставляет сверло в нужную позицию и, нажимая на кнопку  $Z+$ , сверлит отверстие до нужной глубины, после чего нажимает кнопку «Запись» для запоминания позиции и глубины отверстия. Таким образом станок может запомнить несколько точек для сверления. Кнопка «Сброс» используется для удаления всех записанных ранее точек.

В автоматическом режиме, после нажатия кнопки «Пуск» и при наличии заготовки, станок начинает сверление отверстий в «запомненных» точках.

Схема подключения контроллера к станку выглядит следующим образом. Аналоговые датчики положения:  $AIW0$  — положение по оси  $X$ ;  $AIW2$  — положение по оси  $Y$ ;  $AIW4$  — положение по оси  $Z$ ;  $I0.0$  — переключатель режима работы «Ручной»/«Автоматический»;  $I0.1$  — датчик наличия заготовки. Кнопки подключены таким образом: « $X+$ » —  $I0.2$ ; « $X-$ » —  $I0.3$ ; « $Y+$ » —  $I0.4$ ; « $Y-$ » —  $I0.5$ ; « $Z+$ » —  $I0.6$ ; « $Z-$ » —  $I0.7$ ; «За-

пись» — I1.0; «Сброс» — I1.1; «Пуск» — I1.2; «Стоп» — I1.3. Управление станком реализовано через следующие выходы: Q0.0 — шпиндель включен — выключен; AQW0 — задание скорости по оси X; AQW2 — задание скорости по оси Y; AQW4 — задание скорости по оси Z.


**Задача.** Создать программу для контроллера, управляющего станком, которая реализует описанный алгоритм работы.

## 2.3. Задачи по теме «Электроавтоматика станков с ЧПУ»

### 2.3.1. Автоматизация токарного станка

Одной из задач системы управления станка (системы ЧПУ) является задача управления устройствами электроавтоматики в ручном и автоматическом режиме работы станка. Эта задача, как правило, возлагается на программируемый логический контроллер (далее — контроллер), который в том или ином варианте реализации присутствует в системе управления станка. Предлагается разобраться в том, как работают несколько типичных устройств автоматики токарного станка и написать управляющие программы для них, которые будут выполняться контроллером. Команды на выполнение тех или иных действий могут поступать как с пульта управления станка, так и с устройства ЧПУ (УЧПУ) в автоматическом режиме работы станка.

Программирование устройств автоматики токарного станка может быть выполнено на специальном учебном стенде, изображенном на рис. 3, или в комплексе программ «Эмулятор контроллера» и «Имитатор электрооборудования токарного станка».

Главное окно программы «Имитатор электрооборудования токарного станка» служит для выбора имитируемого узла станка: коробки скоростей, патрона электромеханического, ограждения, револьверной головки. Для каждого из узлов предусмотрен автоматический контроль над правильностью его работы под управлением программы, созданной обучаемым. Для проверки программы, ее необходимо загрузить в эмулятор контроллера и запустить, а в программе-имитаторе токарного станка выбрать режим контроля (поэтапный для пошаговой проверки или автоматический для полной автоматической проверки) и нажать на кнопку . По результатам контроля в строке


состояния программы-имитатора будет выведено сообщение о правильности работы узла или об ошибке с кратким ее описанием. Нажав на кнопку , можно посмотреть этапы контроля.



Рис. 3. Внешний вид стенда токарного станка

Объект управления «Коробка скоростей» имеет два вала. Передача движения от входного к выходному валу осуществляется одной из двух зубчатых пар. Одна из шестерен каждой пары сопряжена с валом с помощью электромагнитной муфты. Цепи коммутации электромуфт управляются через выходы контроллера: Q1.0 — первая ступень; Q1.1 — вторая. Включение 1-й и 2-й ступени контролируется датчиками, подключенными соответственно к входам I1.4 и I1.5 контроллера. Например, чтобы включить 1-ю ступень, необходимо подать на выход контроллера Q1.0 логическую единицу. При этом, если включение действительно будет иметь место, на вход I1.4 контроллера также поступит логическая единица.

На макете включение муфт отображается зажиганием соответствующих светодиодов. Срабатывание датчиков необходимо имитировать переключением тумблеров.

**Задача 1.** Разработать схему управления коробкой скоростей, работающей следующим образом.

УЧПУ передает команду включения 1-й или 2-й ступени сигналами, постоянно поступающими на входы контроллера I0.0 или I0.1. Схема управления должна включить соответствующую электромуфту. После того как датчик подтвердит включение заданной ступени, схема управления должна ответить УЧПУ сигналом готовности, подаваемым логической единицей на выход Q0.0 контроллера. Сигнал готовности подается всегда, когда заданная УЧПУ ступень соответствует реально включенной. Сигнал готовности снимается, если поступившая от УЧПУ команда не соответствует включенной ступени и схема управления должна произвести переключение электромуфты. Если устройство УЧПУ ошибочно выдаст сразу две команды включения обеих ступеней, то ни одна ступень не должна быть включена, сигнал готовности должен быть снят.

Губки патрона механического приводятся в движение от реверсивного привода постоянного тока. Включение мотора привода для зажима и разжима патрона осуществляется путем подачи выходных сигналов контроллера Q0.3 и Q0.4. У патрона есть один датчик усилия зажима и окончания разжима. Датчик подключен ко входу I0.5 контроллера. В процессе зажима и разжима датчик выдает импульсы, которые подсчитывает схема управления для определения момента окончания зажима или разжима.

На макете вращение мотора в одном или другом направлении индицируется зажиганием соответствующего светодиода. Импульсы с датчика усилия зажима — разжима имитируются кратковременным нажатием соответствующей кнопки.

**Задача 2.** Разработать схему управления патроном, работающую следующим образом.

УЧПУ передает импульсные сигналы зажима и разжима патрона на входы контроллера I0.2 и I0.3 соответственно. В зависимости от команды, схема должна включить вращение мотора привода патрона в нужном направлении. Выключиться вращение мотора должно после считывания схемой управления трех импульсов с датчика зажима —



разжима. После выполнения команды от УЧПУ, проектируемая схема управления должна выдать сигнал готовности на УЧПУ через выход контроллера Q0.1. Сигнал готовности снимается, когда поступает команда, не соответствующая текущему состоянию патрона.

Кожух ограждения перемещается вручную по направляющим влево и вправо, открывая и закрывая доступ к зоне резания. Когда ограждение закрывает доступ к зоне обработки, срабатывает конечный выключатель, и с него на вход I1.3 контроллера подается сигнал логической единицы.

**Задача 3.** Составить схему управления, которая будет передавать УЧПУ через выход Q0.2 контроллера сигнал логической единицы, если датчик ограждения не сработал.

Револьверная головка состоит из двух узлов. Первый — резцедержатель, рассчитанный на установку, включающую до 8 инструментов. В рабочем положении в каждый момент времени может находиться только один инструмент, закрепленный в резцедержателе. Номер позиции резцедержателя, находящейся в текущий момент в рабочем положении, фиксируется датчиком, который выдает сигнал в двоичном коде по трем сигнальным линиям, подключенным ко входам [I1.0, I0.7, I0.6] =  $I_d$  контроллера, где вход I0.6 — младший разряд датчика. Соответствие сигналов датчика и позиции резцедержателя описано в табл. 2.

Таблица 2

Позиция резцедержателя, находящегося в рабочем положении

Разряд датчика	1	2	3	4	5	6	7	8
I0.6	0	1	0	1	0	1	0	1
I0.7	0	0	1	1	0	0	1	1
I1.0	0	0	0	0	1	1	1	1

Вращение резцедержателя осуществляется реверсивным приводом постоянного тока. Коммутация цепей мотора вращения осуществляется выходными сигналами Q0.5 и Q0.6 контроллера: Q0.5 — вращение в сторону увеличения номера позиции резцедержателя, находящегося в рабочем положении; Q0.6 — вращение в обратную сторону.

На макете включение вращения мотора по часовой или против часовой стрелки отображается зажиганием одного из двух светодиодов. Показания датчика положения резцедержателя задаются вручную тремя переключателями.

Вторым узлом револьверной головки является механизм зажима — разжима резцедержателя. Соответствующая функция осуществляется стопором, перемещаемым штоком гидроцилиндра, который управляется двухпозиционным электрогидрозолотником. При подаче напряжения на обмотку золотника с выхода Q0.7 контроллера, шток гидроцилиндра перемещается, осуществляя фиксацию резцедержателя. При выключении выхода Q0.7 происходит расфиксация резцедержателя. Зажим — разжим обслуживают два датчика, подключенные ко входам I1.1 и I1.2 контроллера. Подача сигнала логической единицы на вход I1.1 с одного датчика соответствует окончанию процесса разжима, а подача сигнала логической единицы на вход I1.2 с другого датчика — окончанию зажима резцедержателя.

На макете подача напряжения на обмотку электрогидрозолотника отображается зажиганием светодиода, а срабатывание путевых конечных выключателей штока гидроцилиндра имитируется двумя тумблерами.

**Задача 4.** Спроектировать схему управления револьверной головкой, работающей в двух режимах: ручном и автоматическом. Выбор режима осуществляется тумблером пульта управления, подключенным ко входу I0.0 контроллера. Логический ноль на этом входе соответствует заданию ручного режима работы, логическая единица — автоматического. Индикацию режима вывести на пульт управления через два выхода Q0.0 и Q0.1 контроллера. В ручном режиме схема управления принимает сигналы управления с кнопок, подключенных ко входам I0.1 и I0.2 контроллера. Если нажата первая из этих кнопок, то резцедержатель должен повернуться на одну позицию в сторону увеличения номера инструментального гнезда, попадающего в рабочее положение. Если нажата кнопка, соединенная с входом I0.2 контроллера, то резцедержатель поворачивается на одну позицию в другую сторону — в сторону уменьшения номера инструментального гнезда, попадающего в рабочее положение. Схема управления должна вырабатывать сигнал готовности, который отсутствует все время, пока она выполняет полученную команду. Таким образом, сигнал готовности снимается, когда поступает любая из двух команд, и выставляется, когда резцедержатель зафиксирован в новом положении. Сигнал готовности выводится на индикацию на цифровом пульте через выход Q0.2 контроллера. Переход из ручного режима управления в автоматический может происходить только при наличии сигнала готовности.



В автоматическом режиме сменой инструмента управляет УЧПУ. Во-первых, УЧПУ подает на входы [I0.3, I0.4 и I0.5] =  $I_u$  контроллера в двоичном коде номер гнезда инструмента, которое необходимо вывести в рабочее положение. Во-вторых, УЧПУ подает на вход I1.3 короткий импульс положительной полярности, во время которого схема управления должна считать с входов I0.3...I0.5 контроллера информацию о том, куда перемещать резцедержатель. Схема управления, считав код гнезда, снимает сигнал готовности, расфиксирует и начинает вращать резцедержатель (если это требуется) по кратчайшему пути, чтобы выполнить задание УЧПУ. После позиционирования резцедержателя, схема управления фиксирует его и выдает сигнал готовности на УЧПУ. Теперь схема управления готова выполнять следующую команду УЧПУ или обрабатывать команду перехода в ручной режим работы.

Задачи автоматизации токарного станка могут быть решены при помощи следующих команд контроллера:

- команды блока Bit Logic — normally open, normally close, output;
- команды блока Counters — Count Up (CTU);
- команды блока Integer Math — Subtract Integer (Sub\_I), Increment Word (Inc\_W), Decrement Word (Dec\_W);
- команды блока Move — Move Word (Mov\_W).

### **2.3.2. Автоматизация механизма смены инструмента**

На сложных станках одной из основных систем автоматики является система инструментального обеспечения. Задача управления этой системой, как правило, возлагается на программируемый логический контроллер (далее — контроллер). Предлагается разобраться в том, как работает один из вариантов механизма смены инструмента, обрабатывающего центра и написать управляющие программы для него, которые будут выполняться контроллером. Команды на выполнение тех или иных действий могут поступать как с пульта управления станка в ручном режиме работы станка, так и от устройства числового программно-го управления (УЧПУ) в автоматическом режиме работы станка.

Стенд механизма смены инструмента изображен на рис. 4. Задачи по автоматизации его узлов могут быть решены как на самом стенде, так и с использованием программы «Механизм смены инструмента».

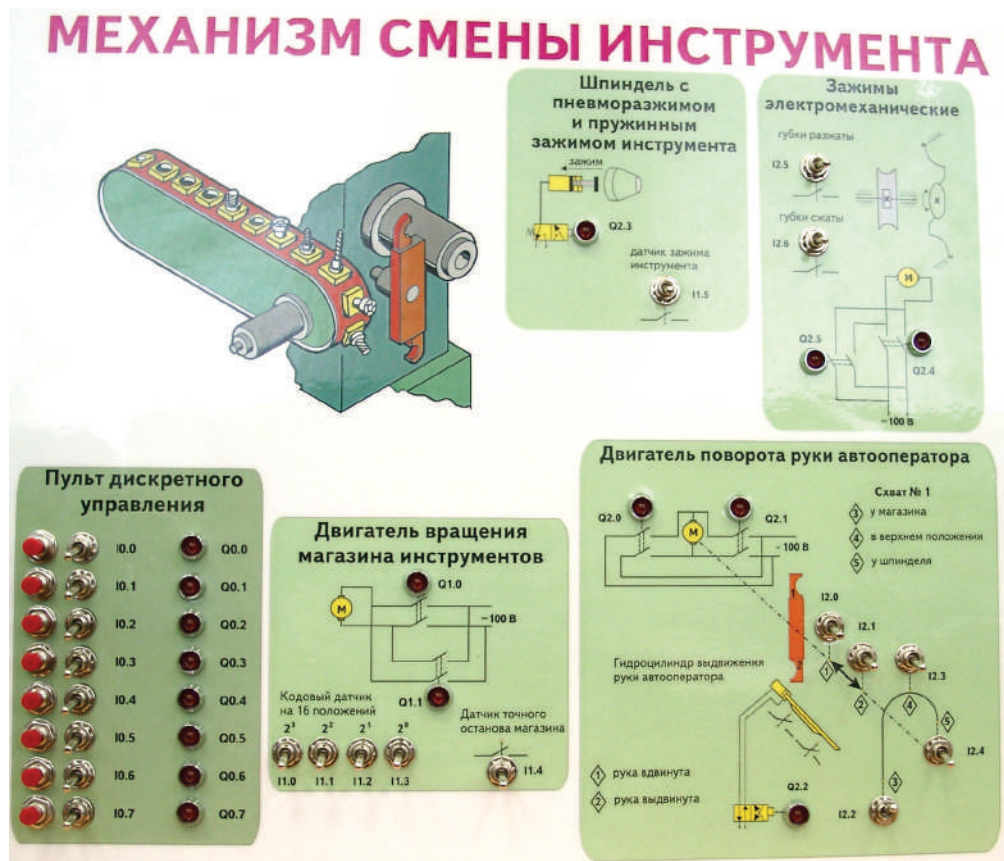


Рис. 4. Стенд механизма смены инструмента

Программа-имитатор механизма смены инструмента снабжена переключателем «Стоп» типа check-box, который позволяет остановить работу приводов имитируемых механизмов, чтобы воспроизвести, например, их заклинивание.

Требуемое усилие зажима инструмента в пневмопатроне обеспечивается пружиной. Зажим инструмента фиксируется срабатыванием датчика, подключенного ко входу I1.5 контроллера, т. е. подачей на этот вход сигнала логической единицы. Разжим инструмента в патроне осуществляется с помощью пневмоцилиндра. Когда с выхода Q2.3 контроллера подается управляющее напряжение на катушку пневмозолотника, шток пневмоцилиндра перемещается так, что сжимает пружину патрона, это позволяет вынимать и вставлять инструмент в патрон. При снятии управляющего напряжения с катушки

электропневмозолотника, шток пневмоцилиндра перестает противодействовать пружине и под действием последней инструмент зажимается в патроне. Если в патрон не вставлен инструмент, то датчик зажима зафиксирует достигнутое усилие сжатия губок патрона и выдаст сигнал логической единицы на вход I1.5 контроллера.

На макете подача напряжения на катушку электрогидрозолотника отображается зажиганием светодиода, срабатывание датчика зажима инструмента имитируется включением тумблера.

**Задача 1.** Разработать схему управления патроном, работающую по следующему алгоритму: при подаче команды разжима инструмента логической единицей на вход I0.0 контроллера, схема управления выдает сигнал логической единицы на катушку электропневмозолотника через выход Q2.3 контроллера. При подаче сигнала зажима инструмента в патроне логическим нулем на вход I0.0 контроллера, схема управления выдает сигнал логического нуля на катушку электропневмозолотника через выход Q2.3 контроллера. Кроме того, схема управления должна вырабатывать сигнал готовности на выходе контроллера Q0.0. Логика выдачи сигнала такова: при подаче на вход контроллера любой из команд зажима или разжима патрона, схема управления снимает сигнал готовности, а выставляет его после того, как датчик зажима инструмента подтвердит выполнение команды. Для разжима инструмента — это сигнал логического нуля, поданный на вход I1.5 контроллера, для зажима — сигнал логической единицы, поданный на тот же вход контроллера.

Губки электромеханического зажима инструмента руки автооператора приводятся в движение реверсивным приводом постоянного тока. Включение мотора привода для зажима или разжима губок схвата осуществляется подачей выходных сигналов контроллера Q2.4 и Q2.5 соответственно. Состояние губок «сжаты — разжаты» контролируется двумя датчиками, подключенными ко входам I2.6 и I2.5 контроллера соответственно.

На макете вращение мотора индицируется светодиодами, которые показывают состояние контакторов, управляемых выходами Q2.4 и Q2.5 контроллера. Срабатывание датчиков зажима и разжима моделируется с помощью двух тумблеров.

**Задача 2.** Разработать схему управления электромеханическим зажимом губок схвата руки автооператора, работающую по следующему алгоритму: при подаче команды зажима логической единицей на вход I0.3 контроллера, схема управления включает вращение мото-

ра зажима через выход Q2.4 и прекращает его вращение, когда срабатывает датчик зажима. Датчик зажима представляет собой конечный выключатель, подключенный ко входу I2.6 контроллера. Выдача сигнала логической единицы этим датчиком соответствует состоянию губок «зажаты». При подаче команды разжима на вход I0.4 контроллера, схема управления включает вращение мотора в обратном направлении через выход Q2.5. Вращение должно прекратиться при срабатывании датчика разжима, работающего аналогично датчику зажима, но подключенного ко входу I2.5 контроллера. Кроме того, схема управления должна вырабатывать сигнал готовности на выходе контроллера Q0.1 следующим образом: когда поступает сигнал зажима или разжима, сигнал готовности снимается и выставляется, когда выполнение команды успешно завершено.

Узел руки автооператора состоит из двух механизмов: выдвижения руки автооператора и ее поворота. Узел предназначен для переноса инструмента. Рука автооператора имеет два зажима, работа которых описана в предыдущем пункте. Один из вариантов смены инструмента осуществляется следующим образом:

- рука автооператора поворачивается по часовой стрелке на  $90^\circ$  и двумя своими зажимами захватывает инструмент в патроне и магазине;
- рука выдвигается, извлекая инструменты из патрона и магазина;
- рука поворачивается на  $180^\circ$  против часовой стрелки;
- рука задвигается, завершая таким образом взаимную замену инструментов в патроне и магазине;
- рука автооператора поворачивается против часовой стрелки на  $90^\circ$  и возвращается в исходное состояние.

Движение выдвижения и задвижения руки автооператора осуществляется гидроцилиндром. Подача напряжения с выхода Q2.2 контроллера на катушку электрогидрозолотника приводит к выдвижению руки, снятие напряжения — к задвижению руки. Два конечных выключателя, подключенных ко входам I2.0 и I2.1 контроллера, фиксируют состояние руки «задвинута» и «выдвинута» соответственно. Вращение руки автооператора осуществляется с помощью реверсивного электропривода постоянного тока. Подача напряжения с выхода Q2.0 контроллера на соответствующий контактор в цепи коммутации мотора приводит к вращению руки автооператора по часовой стрелке. Вращение руки

автооператора против часовой стрелки осуществляется при включении выхода Q2.1 контроллера. Конечный выключатель, подключенный ко входу I2.3 контроллера, выдает сигнал логической единицы, когда рука автооператора находится в вертикальном положении. Два конечных выключателя, подключенных ко входам I2.2 и I2.4 контроллера, выдают сигналы логической единицы, когда схват № 1 руки автооператора находится у магазина или у патрона соответственно.

На макете состояние электрогидрозолотника управления гидроцилиндром выдвижения — задвижения руки автооператора отображается светодиодом. Работа конечных выключателей задвижения и выдвижения руки моделируется с помощью двух тумблеров. Состояние мотора поворота руки автооператора отображается двумя светодиодами, показывающими, какой из контакторов в цепи электромотора включен. Срабатывание конечных выключателей, положения руки автооператора моделируется с помощью двух тумблеров.

**Задача 3.** Составить схему управления, обеспечивающую обмен инструментом между патроном и текущим гнездом магазина инструментов. Команда смены инструмента поступает на вход контроллера I0.7. Схема должна вырабатывать сигнал готовности и выводить его на выход контроллера Q0.2. Логика синтеза сигнала готовности следующая: при поступлении команды смены инструмента, схема управления снимает сигнал готовности и начинает цикл смены инструмента. После завершения цикла схема управления выставляет сигнал готовности. Прием и начало обработки команды смены инструмента может производиться при условии выдачи схемой управления сигнала готовности.

Магазин инструментов представляет собой накопитель замкнутого ленточного типа, имеющий 16 гнезд для хранения инструмента. Лента с инструментом может вращаться в обоих направлениях. Магазин имеет датчик точного останова любого из гнезд в позиции перегрузки инструмента. Кодовый четырехразрядный двоичный датчик выдает в двоичном коде информацию о том, какое звено сейчас находится в зоне перегрузки. Останов вращения магазина производится по сигналу датчика точного останова. Последний подключен ко входу I1.4 контроллера; кодовый датчик подключен ко входам  $[I1.0...I1.3] = I_d$  контроллера: младший разряд датчика подключен ко входу I1.3, старший — к I1.0. При нахождении нулевого гнезда в позиции перегрузки, кодовый датчик формирует двоичный код, равный нулю. При нахождении гнез-



да № 1 в позиции перегрузки, кодовый датчик формирует двоичный код, равный единице. Перемещение магазина осуществляется реверсивным приводом постоянного тока. Мотор коммутируется для вращения по часовой стрелке (в сторону увеличения номера гнезда, оказывающегося в позиции перегрузки) посредством выдачи логической единицы на выход Q1.0 контроллера. При включении выхода Q1.1 контроллера срабатывает контактор, обеспечивающий вращение магазина и инструментов в обратном направлении.

На макете срабатывание каждого из контакторов, что соответствует вращению мотора и магазина в прямом или обратном направлении, отображается соответствующим светодиодом. Работу кодового датчика имитируют набором требуемой по логике работы магазина инструментов комбинацией включения — выключения четырех тумблеров. Работу датчика точного останова имитируют отдельным тумблером.

**Задача 4.** Спроектировать схему управления магазином инструментов, осуществляющую перемещение заданного гнезда инструментальной ленты в позицию перегрузки в соответствии со следующим алгоритмом. Сперва УЧПУ выдает на входы контроллера  $[I0.1 \dots I0.4] = I_{cy}$  двоичный код гнезда, которое (гнездо) необходимо переместить в позицию перегрузки. Далее УЧПУ дает короткий импульс логической единицы на вход I0.5 контроллера. Этот импульс служит сигналом, по которому схема управления должна осуществить смену инструмента. Схема управления должна, вращая мотор магазина, кратчайшим путем переместить к позиции перегрузки гнездо с заданным УЧПУ номером. Схема управления должна вырабатывать сигнал готовности и подавать его на выход Q0.1 контроллера. Сигнал снимается при поступлении команды от УЧПУ и выставляется, когда команда выполнена. Отметим, что схема управления принимает команду от УЧПУ на обработку только в состоянии готовности.

**Задача 5.** Составить схему управления всем механизмом смены инструмента. На вход схемы управления (входы I0.1... I0.4 контроллера) поступает от УЧПУ код ячейки магазина, с которой должен быть произведен обмен инструмента в патроне, и короткий сигнал «пуск» (вход I0.0 контроллера), инициирующий эту смену (см. задачу 4). Схема управления должна вырабатывать сигнал готовности для УЧПУ (выход Q0.0 контроллера). Если схема управления находится в состоянии готовности, то она выполняет цикл смены инструмента: перемещает нужную ячейку магазина к позиции перегрузки; расфиксирует

патрон; разжимает схваты автооператора; выполняет обмен инструментом с помощью автооператора, сжимая и разжимая схваты и поворачивая руку автооператора, когда необходимо; убирает автооператор в исходное вертикальное положение.

Сигнал готовности снимается при поступлении команды от УЧПУ и выставляется, когда команда выполнена. (Подробности, включая адреса используемых входов и выходов контроллера, см. в предыдущих задачах.)

**Задача 6.** Условие задачи 5 усложняется следующим образом: схема управления должна отводить на выполнение каждой элементарной операции не более заданного количества времени — вращение магазина — 10 с, разжим или зажим инструментов в схвате — 3 с, выдвижение или задвижение руки автооператора — 3 с, поворот руки автооператора в любом направлении — на  $90^\circ$  — 3 с, на  $180^\circ$  — 5 с. Если схема управления фиксирует, что определенная операция не выполнена за отведенное время, то она должна выставить сигнал ошибки соответствующего узла. Если вращение магазина не уложилось в 10 с, то мотор вращение выключается, при этом включается индикатор ошибки — выход контроллера Q0.4, а сигнал готовности остается снятым. Для схвата сигнал ошибки выводится на выход Q0.5 контроллера; для цилиндра выдвижения — задвижения автооператора — на выход Q0.6 контроллера; для мотора поворота руки автооператора — на выход Q0.7 контроллера. Сигнал готовности выставляется схемой управления только в том случае, если все операции при смене инструмента были выполнены за время не больше заданного.

Задачи автоматизации механизма смены инструмента могут быть решены при помощи следующих команд контроллера:

- команды блока Bit Logic — normally open, normally close, output;
- команды блока Integer Math — Subtract Integer (Sub\_I);
- команды блока Move — Move Word (Mov\_W);
- команды блока Timers — On-Delay Timer (TON).

### 2.3.3. Управление механизмами участка механообработки

Участок механообработки (рис. 5) включает в себя два фрезерных станка, одну контрольно-измерительную машину (КИМ), стол загрузки заготовок, два стола выгрузки: один для годных деталей, второй для брака — и автоматизированную тележку, которая перемещает за-

готовки и детали между перечисленными выше объектами. Управляет участком программируемый логический контроллер (далее — контроллер), который взаимодействует при необходимости с системами числового программного управления (СЧПУ) обоих станков и системой управления КИМ.

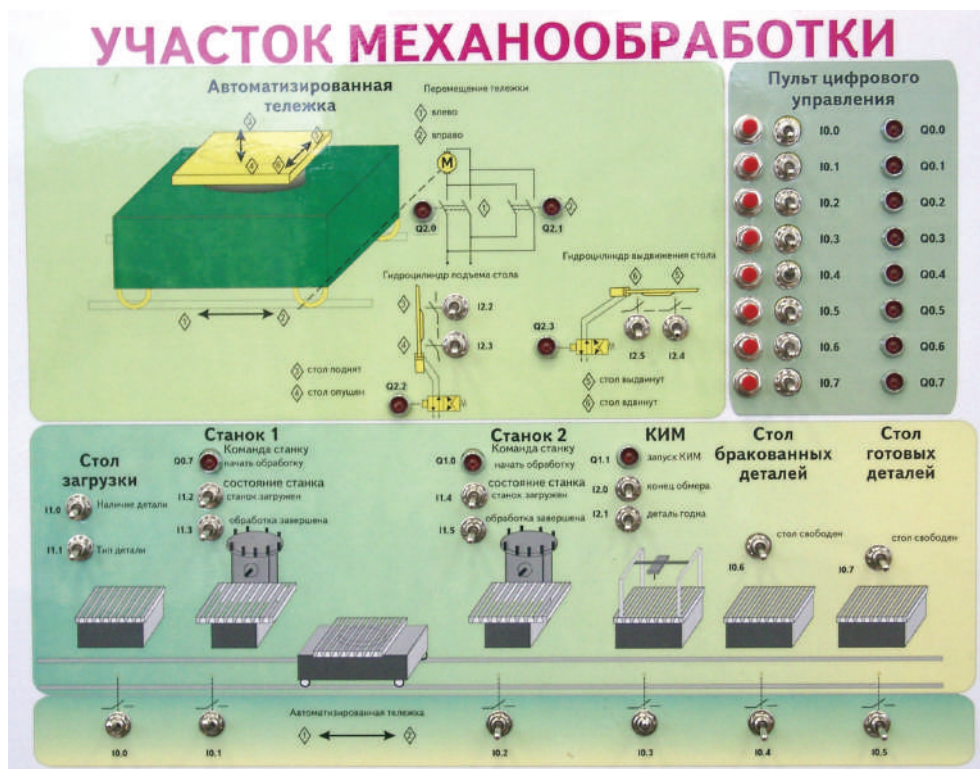


Рис. 5. Стенд участка механообработки

Автоматизация участка может быть выполнена на учебном стенде или при помощи программы «Участок механообработки». В программном имитаторе участка механообработки имеется всплывающее меню, которое появляется после щелчка правой кнопкой мыши по окну программы. Всплывающее меню позволяет положить — убрать заготовку на стол заготовок и выбрать тип заготовки.

Станок участка механообработки имеет собственную систему управления, в нее загружаются требуемые управляющие программы. Станок представляет собой объект управления, который может находиться в трех состояниях: станок не загружен заготовкой, станок загружен



заготовкой и на нем ведется ее обработка, станок загружен, и на нем находится готовая деталь. Взаимосвязь между СЧПУ станка и системой управления всего участка, т. е. контроллером, осуществляется с помощью трех сигналов. Два сигнала подаются от станка на входы И1.2 и И1.3 (И1.4 и И1.5 — для второго станка) контроллера управления участком. Присутствие первого сигнала означает, что станок загружен: на нем находится заготовка или готовая деталь. Если на станке нет заготовки ни в каком виде (т. е. обработанном или необработанном), то на вход И1.2 (И1.4) контроллера выдается сигнал логического нуля.

Второй сигнал выдается станком, когда обработка заготовки на нем завершена и с него можно снять готовую деталь. Этот сигнал снимается СЧПУ станка, как только на станок выдается команда «Начать обработку» с выхода контроллера Q0.7 (Q1.0).

Сигнал с выхода Q0.7 (Q1.0) является третьим из сигналов, которыми обменивается станок и система управления участком. Когда последняя, т. е. контроллер, хочет инициировать обработку заготовки на станке, тогда ему необходимо подать сигнал логической единицы на станок с выхода Q0.7 (Q1.0) контроллера. Аналогичные три сигнала взаимодействия контроллера со станком 2 подключены к контроллеру через входы И1.4 и И1.5 и выход Q1.0.

На макете выдача команды «Начать обработку» отображается светодиодом, а выработка сигналов «Станок загружен» и «Обработка завершена» имитируется с помощью двух тумблеров.

**Задача 1.** Разработать схему управления, которая осуществит обмен сигналами между контроллером (т. е. системой управления участком механообработки) и станком. По нажатию кнопки «Пуск» на пульте управления (вход И0.0 контроллера) схема управления должна выдать сигнал «Начать обработку» через выход Q0.7 контроллера при условии, что станок загружен и обработка на нем завершена, т. е. на входе И1.2 присутствует логическая единица, а на входе И1.3 — логический ноль. При поступлении команды «Начать обработку» станок начинает обработку заготовки и снимает сигнал «Обработка завершена». Изменение этого сигнала имитируется переключением тумблера. Этим же тумблером имитируется конец обработки заготовки на станке, после чего схема управления должна снять сигнал «Начать обработку».

Начальное состояние станка соответствует выключенному тумблеру «Станок загружен» и включенному «Работа завершена». Если станок загружен новой заготовкой и на его систему управления (СЧПУ) по-

ступила команда «Начать обработку», то СЧПУ должна снять сигнал «Обработка завершена» и начать обработку. После того как обработка будет завершена, должен быть выставлен одноименный сигнал. После этого станок может быть разгружен (при этом сигнал «Обработка завершена» не изменяется), затем загружен новой заготовкой. Это имитируется включением сигнала «Станок загружен» и снятием сигнала «Обработка завершена». Теперь, при поступлении очередного сигнала «Начать обработку», станок готов обрабатывать новую заготовку.

Станок загружается заготовкой. Этому соответствует включенное состояние тумблера «Станок загружен» и по-прежнему выключенное состояние тумблера «Обработка завершена». Следующим должно наступить состояние, когда заготовка обработана. Этому соответствует включенное состояние тумблеров «Станок загружен» и «Обработка завершена». Завершающее в цикле обработки детали на станке состояние соответствует включенному состоянию тумблера «станок загружен» и выключенному — «обработка завершена». Далее возможна загрузка станка и осуществление очередного цикла обработки заготовки на нем.

Контрольно-измерительная машина (КИМ), как и станок, имеет собственную систему управления. На описываемом участке КИМ запускается от системы управления всем участком — от контроллера. Цикл работы КИМ описывается следующим образом: в КИМ загружается деталь, подлежащая измерению, и подается сигнал выполнения обмера логической единицей с выхода Q1.1 контроллера. После того как измерительный цикл заканчивается, КИМ выдает на вход I2.0 контроллера сигнал «Конец обмера», а на вход I2.1 контроллера — сигнал логической единицы, если деталь годна, и сигнал логического нуля, если деталь признана бракованной по результатам обмера.

При поступлении сигнала «Запуск КИМ» с входа Q1.1 контроллера, система управления КИМ сбрасывает в логический ноль сигналы «Конец обмера» и «Деталь годна» на входах I2.0 и I2.1 контроллера.

На макете выдача контроллером команды «Запуск КИМ» отображается светодиодом, а выработка сигналов «Конец обмера» и «Деталь годна» имитируется с помощью двух тумблеров.

**Задача 2.** Разработать схему управления, которая осуществит управление КИМ. По нажатию кнопки «Пуск» (кнопка на пульте, подключенная ко входу I0.3 контроллера), при наличии сигнала с КИМ «Конец обмера», схема управления должна выдать сигнал Q1.1 «За-

пуск КИМ». При этом КИМ снимает сигнал «Конец обмера» с входа I2.0 контроллера. После завершения своей работы КИМ посылает сигнал «Конец обмера» на вход I2.0 контроллера. Пока последний сигнал истинен, схема управления может считать с входа I2.1 контроллера сигнал годности детали. При получении сигнала «Конец обмера», схема управления контроллера должна снять сигнал «Запуск КИМ».

Для выключения сигнала «Запуск КИМ» в схеме управления можно использовать инструкцию, выделяющую передний фронт сигнала, — Positive Transition.

Автоматизированная тележка перемещается по рельсам в двух направлениях от стола загрузки заготовок до стола готовых деталей, имея шесть точек позиционирования, которые отмечены конечными выключателями, подключенными ко входам контроллера I0.0 ... I0.5. Тележка приводится в движение мотором постоянного тока, коммутируемым двумя контакторами, подключенными к выходам Q2.0 и Q2.1. При включении выхода Q2.0 тележка перемещается от стола загрузки к столу готовых деталей. При включении выхода Q2.1 тележка перемещается в обратном направлении. При выключенных обоих контакторах тележка находится в состоянии покоя.

Кроме привода перемещения, на тележке имеется стол для транспортировки заготовок и деталей от стола загрузки к станкам, КИМ и столам бракованных и готовых деталей. Транспортный стол имеет два гидравлических привода для своего подъема — опускания и выдвигания — втягивания.

Рассмотрим циклы загрузки и разгрузки транспортного стола тележки, являющиеся одинаковыми при работе тележки как с любым из трех столов участка, так и с любым из двух станков и с КИМ. Начальное состояние транспортного стола следующее: он опущен (конечный выключатель I2.3 включен) и вдвинут (конечный выключатель I2.5 включен). Это состояние поддерживается выдачей «нулевых» сигналов с выходов Q2.2 и Q2.3 контроллера на соответствующие электрогидрозолотники.

Цикл загрузки транспортного стола заготовкой или деталью осуществляется следующим образом: транспортный стол выдвигается подачей управляющего сигнала Q2.3 на электрогидрозолотник гидроцилиндра выдвигания стола. После срабатывания конечного выключателя I2.4, фиксирующего конец выдвигания, необходимо поднять стол. Для этого выдается сигнал Q2.2 с выхода контроллера на элек-

трогидрозолотник гидроцилиндра подъема транспортного стола. Сигнал Q2.3 при этом не снимается, т. к. иначе стол начнет вдвигаться. Срабатывание конечного выключателя I2.2 отмечает момент окончания подъема стола. Теперь стол необходимо выдвинуть снятием сигнала Q2.3 с выхода контроллера и после срабатывания конечного выключателя I2.5 опустить его снятием сигнала Q2.2. Цикл заканчивается, когда срабатывает конечный выключатель I2.3. Стол оказывается в исходном положении загруженным заготовкой или деталью.

Цикл разгрузки транспортного стола осуществляется в следующем порядке. Сперва стол поднимается. Для этого необходимо выдать единичный сигнал с выхода Q2.2 контроллера. Когда сработает конечный выключатель I2.2, необходимо выдать единичный сигнал с выхода Q2.3 контроллера (не снимая сигнал Q2.2). Стол начнет выдвигаться. Как только сработает конечный выключатель I2.4, необходимо опустить стол, сняв сигнал Q2.2. Окончание опускания стола будет зафиксировано срабатыванием конечного выключателя I2.3. В этот момент необходимо снять сигнал Q2.3 для того, чтобы выдвинуть назад пустой стол. Окончание цикла фиксируется срабатыванием конечного выключателя I2.5. При этом стол оказывается в начальном положении без заготовки или детали. Сигналы Q2.2 и Q2.3 в конце цикла (как и в начале) равны логическому нулю.

Отметим, что штоки гидроцилиндров подъема и выдвижения стола ограничены в перемещении механически, т. е. работают по упорам. Конечные выключатели лишь информируют контроллер, что движение в определенном направлении закончено: транспортный стол достиг крайней точки по одной из осей своего перемещения. При этом если необходимо, чтобы шток соответствующего цилиндра оставался в достигнутом конечном положении, управляющий им сигнал не должен меняться. В результате этого шток не движется, но поджимается к упору.

На макете включение-выключение контроллером контакторов в силовой цепи привода перемещения тележки и электрогидрозолотников гидроцилиндров подъема и выдвижения стола отображается соответствующими четырьмя светодиодами. Срабатывание конечных выключателей, фиксирующих подъем, опускание, выдвижение и втягивание транспортного стола, имитируется с помощью соответствующих четырех тумблеров.

**Задача 3.** Автоматизированная тележка в начальном положении находится у стола загрузки. Транспортный стол опущен и находится

во вдвинутом состоянии. Необходимо написать управляющую программу для контроллера, которая обеспечит следующий алгоритм работы тележки: по сигналу «Пуск цикла» с кнопки I0.6 тележка должна переместиться к станку № 1 и остановиться. При нажатии на кнопку «Продолжение цикла» I0.7 тележка должна продолжить движение и остановиться у КИМ. Далее, после очередного нажатия на кнопку «Продолжение цикла», тележка должна доехать и остановиться у стола готовых деталей. Следующее нажатие на кнопку «Продолжение цикла» должно привести к перемещению тележки в обратном направлении к столу загрузки. Около него тележка должна остановиться. Цикл перемещения тележки окончен. Она оказалась в начальном положении и готова к исполнению очередного такого же цикла.

При нажатии на кнопку «Пуск цикла» необходимо зажигать светодиод Q0.0 на пульте управления. Светодиод должен гаснуть, когда отработка цикла перемещения тележки закончится.

**Задача 4.** Автоматизированная тележка в начальном положении находится у стола загрузки. Транспортный стол опущен и находится во вдвинутом состоянии. Необходимо написать управляющую программу для контроллера, которая обеспечит следующий алгоритм работы тележки в ручном режиме управления. После каждого поступления сигнала «Пуск вправо» I0.6 с пульта управления, тележка должна переместиться к ближайшему справа от нее объекту по пути ее перемещения и остановиться. Например, если тележка находилась у стола загрузки, то она должна переместиться к станку № 1 и остановиться. Если команда на перемещение поступает тогда, когда тележка находится в крайнем правом своем положении, то схема управления не должна на нее реагировать, т. е. тележка должна оставаться в этом же положении.

Аналогично указанному должна реагировать схема управления и соответственно тележка на поступление с пульта управления команды «Пуск влево» I0.7. Только теперь тележка должна перемещаться до ближайшего объекта слева по пути ее движения. Во время перемещения тележки схема управления не должна реагировать на любые поступающие команды с пульта управления вплоть до окончания перемещения тележки.

**Задача 4а.** Добавьте в схему управления цепь индикации состояния тележки: если тележка перемещается и, следовательно, не готова воспринимать команды с пульта, то должен быть включен выход Q0.0, в противном случае — должен быть выключен.

**Задача 5.** Условие этой задачи аналогично условию предыдущей задачи, однако появляется дополнительное условие. Схема управления теперь должна реагировать на одно нажатие любой из кнопок пульта управления во время движения тележки. Алгоритм работы всей системы таков: если во время движения тележки с пульта поступила команда, то схема управления запоминает ее и, после выполнения текущего перемещения, выполняет перемещение, соответствующее поступившей во время движения команды.

Таким образом, каждый раз, когда схема управления выполняет последнее заданное движение, она готова принять одну следующую команду. Другими словами, необходимо предусмотреть буфер команд единичной длины. Также следует обеспечить индикацию заполнения буфера команд: когда буфер пуст, выход Q0.1 выключен, когда — полон, выход Q0.1 включен.

**Задача 6.** Составить для контроллера схему управления участком механообработки в автоматическом режиме. Необходимо обеспечить следующий порядок функционирования участка. Начальное состояние объектов участка: тележка находится у стола загрузки, ее транспортный стол вдвинут и опущен; станки первый и второй не загружены, обработка на них завершена; КИМ находится в состоянии конца обмера; столы готовых и бракованных деталей свободны. Начало работы участка инициируется поступлением команды «Пуск автоматической работы» с пульта управления на вход I0.6 контроллера. При этом тележка циклически:

- загружает деталь, подлежащую обработке со стола загрузки, при наличии этой детали нем. При отсутствии детали происходит ожидание сигнала «Наличие детали», после чего происходит загрузка;
- в зависимости от типа загруженной детали везет ее к станку № 1 («тип детали» = «лог. 0») или к станку № 2 («тип детали» = «лог. 1»);
- выгружает деталь на станок, дает команду станку начать обработку;
- ждет завершения обработки;
- загружает обработанную деталь и везет ее к КИМ;
- выгружает деталь на КИМ и дает команду запуска КИМ;
- дожидается конца обмера и загружает деталь с КИМ;
- в зависимости от того, годная деталь или нет, везет ее к столу бракованных или готовых деталей;



- если стол свободен, то деталь перегружается на стол; если стол не свободен, то при появлении сигнала «Стол свободен», деталь перегружается;
- по окончании разгрузки, тележка возвращается к столу загрузки, и все описанные выше действия повторяются.

Циклы выполняются бесконечно.

Выход Q0.0 контроллера используется для индикации состояния участка механообработки: включенное состояние выхода обозначает режим автоматической работы участка, выключенный — состояние останова участка.

Задачи по автоматизации участка механообработки могут быть решены при помощи команд блока Bit Logic: Normally Open, Normally Close, Not, Positive Transition, Negative Transition, Output.

## 2.4. Автоматизация робототехники

---

Системы управления промышленными роботами классифицируют по принципу управления (программное — адаптивное), способу управления (разомкнутые — замкнутые), типу управления (цикловые — позиционные — контурные), способу программирования (аналитическое программирование — программирование обучением — программирование самообучением).

Системы программного управления требуют строгой определенности и постоянства параметров выполняемой задачи. Управляющая программа таких систем содержит объем информации, не изменяющийся в процессе работы, поэтому среда манипулирования робота должна быть организованной, т. е. все предметы, инструменты и объекты, с которыми взаимодействует робот в процессе выполнения рабочих операций, должны находиться на определенных местах и иметь строго определенную пространственную ориентацию.

Системы адаптивного управления не содержат полной информации о параметрах и условиях выполняемой задачи. Управляющая программа этих систем обычно включает в себя информацию о начальном и конечном положениях рабочего органа манипулятора с набором алгоритмов поведения робота в зависимости от возможных состояний внешней среды, а сенсорное обеспечение робота позволяет автомати-

чески корректировать программные действия на основе получаемой информации путем соответствующего изменения управляющих воздействий, т. е. реагировать на изменение параметров и условий работы изменением алгоритма управления.

Способ управления характеризует использование обратных связей, как правило, по положению звеньев робота в системе управления (разомкнутые — без обратных связей и замкнутые — с обратной связью).

Тип системы управления определяет содержание командной информации, управляющей движением манипуляционных звеньев робота.

Цикловая система программного управления — система управления, в которой командная информация, содержащаяся в управляющей программе, включает в себя признак звена манипулятора и направление его движения. Цикловое управление является простейшим, обеспечивая в основном двухточечное позиционирование, которое осуществляется по жестким упорам, расположенным в крайних положениях, и применяется при выполнении роботом вспомогательных операций (при обслуживании станков, прессов, литейных машин и т. д.).

Позиционная система программного управления — система управления, в которой, в отличие от цикловой системы, командная информация содержит не только признак звена и направления движения, но и параметр, определяющий величину перемещения. Позиционное управление является более сложным, обеспечивая многоточечное позиционирование, для чего содержит информацию о положении звеньев непосредственно в управляющей программе. Последовательный ряд положений рабочего органа, отрабатываемых позиционной системой, может образовывать некоторую определенную траекторию, однако перемещение между соседними позициями в таких системах происходит по произвольной траектории. Позиционная система обладает большими технологическими возможностями и универсальностью, поэтому роботы с позиционной системой управления применяются для обслуживания основного оборудования (смена инструмента на механообрабатывающих станках и пр.), а также для выполнения основных технологических операций (адресная доставка груза, точечная сварка, пайка, клепка, сверловка печатных плат и пр.).

Контурная система программного управления — система управления, в которой командная информация (управляющая программа) содержит, кроме признака звена, направления и величины переме-



щения, еще и параметры траектории (контура), по которой осуществляется движение. Контурное управление обеспечивает перемещение рабочего органа по непрерывной траектории, обладает высокой универсальностью и значительными технологическими возможностями. Роботы с контурной системой управления применяются для выполнения, как правило, основных, а не вспомогательных технологических операций, например: окраски, контурной сварки, шлифовки сварных швов, газовой резки и т. д.

По способу программирования информации, обеспечивающей заданные действия робота, различают три основных метода: аналитическое программирование, метод обучения, метод самообучения.

Аналитическое программирование подразумевает заблаговременную подготовку управляющей программы. Расчет программы осуществляют либо с применением обычных средств вычислительной техники, либо автоматически с использованием ЭВМ и средств автоматического программирования, либо с помощью устройства управления самого робота. Аналитическое программирование применяют, когда обучение робота оператором оказывается слишком трудоемким, затруднено получение полной информации или невозможно присутствие оператора при программировании.

В настоящее время в робототехнике наиболее распространено программирование обучением. В зависимости от степени участия оператора этот метод программирования подразделяют на ручной, полуавтоматический и автоматический способы. Ручной способ обучения предусматривает непосредственное участие оператора на всех этапах программирования: при формировании программы, преобразовании и вводе информации. Полуавтоматический способ обучения характеризует участие оператора в формировании программы и преобразовании информации. Ввод информации обеспечивает устройство управления по сигналу оператора. Автоматическое обучение выполняется полностью управляющим устройством с применением ЭВМ.

Программировать самообучением можно роботов с развитым сенсорным аппаратом и адаптивным управлением. Программирование самообучением происходит без участия оператора: устройство управления роботом самостоятельно формирует рабочие программы с помощью систем автоматического программирования на основе информации, получаемой от информационно-измерительной, или сенсорной, системы.

### 2.4.1. Цикловая система управления промышленным роботом МП-9С

Пневматический робот МП-9 С (рис. 6) предназначен для автоматизации технологических процессов в промышленности. Исполнительное устройство робота способно осуществлять захват, перенос и установку детали по заданным координатам рабочей зоны.

Робот имеет 3 пневмопривода механизмов перемещения руки манипулятора: подъема — опускания, поворота влево — вправо, выдвижения — втягивания и пневмопривод схвата. Пневмоцилиндры приводов снабжаются энергией сжатого воздуха, поступающего из магистрали через соответствующие электропневмоклапаны Y1... Y7, управляемые контроллером.

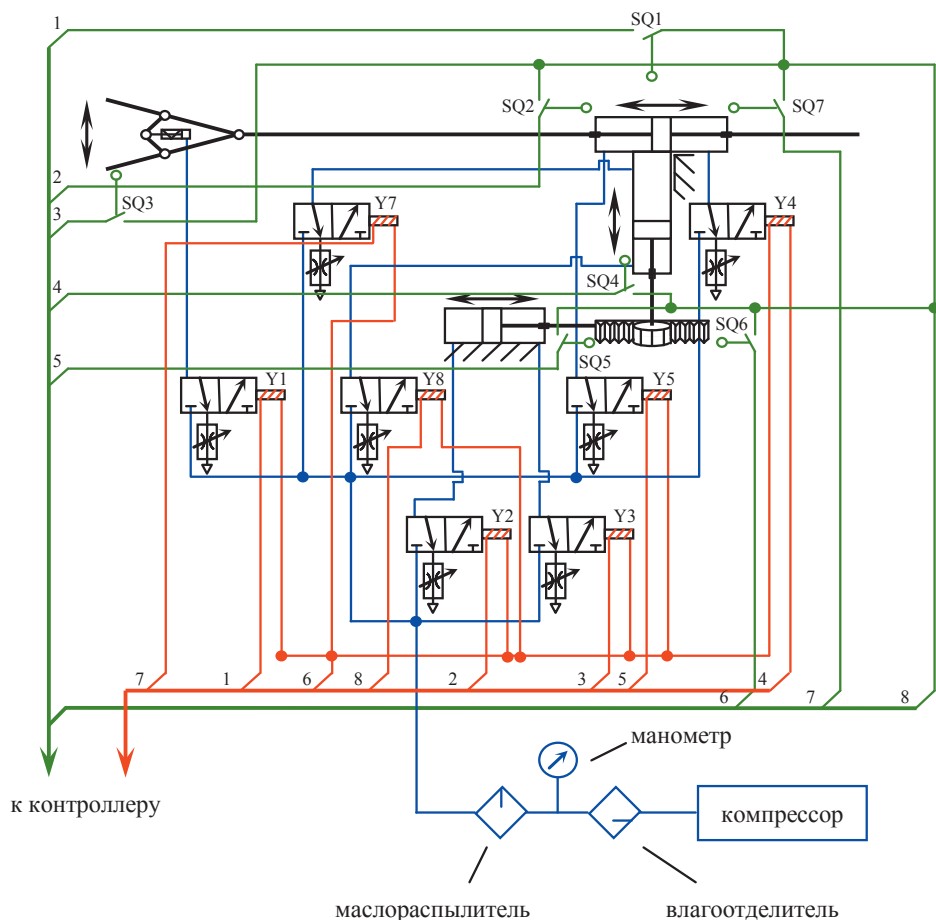


Рис. 6. Схема робота МП-9С

Воздух к магистрали подводится от поршневого компрессора через влагоотделитель и маслораспылитель. Для визуального контроля давления воздуха установлен манометр. Необходимо отметить, что пневмопривод робота выполняет перемещение звеньев с остановкой на упорах, поэтому в установившемся режиме все звенья манипулятора находятся в одном из крайних положений. Например, рука манипулятора может быть либо выдвинута до наезда на передний упор, либо вдвинута до заднего упора. В промежуточных положениях звенья бывают только в процессе перехода из одной позиции в другую. Для контроля положения звеньев робота используются конечные выключатели на герконовых элементах SQ1... SQ7, срабатывающие при достижении приводом робота соответствующего упора. Привод схвата робота снабжен возвратной пружиной, поэтому управляется одним электропневмоклапаном Y1 и контролируется только одним датчиком SQ3, срабатывающим при разжатии схвата.

Система управления роботом (рис. 7) построена на базе программируемого контроллера, в качестве которого выбран SIMATIC S7—200 фирмы Siemens. В состав системы также входит пульт управления, на котором размещены 6 тумблеров и одна кнопка, из них тумблер SA1 служит для выбора режима работы робота: ручного (программирование) или автоматического (отработка запрограммированных кадров). Тумблер SA6 определяет режим работы программы в автоматическом режиме: пошаговый или непрерывное (цикловое) выполнение программы. Остальные 4 тумблера используются для управления пневмоприводами робота в ручном режиме. Кнопка SB1 пульта управления является многофункциональной. В ручном режиме кратковременное нажатие этой кнопки приводит к записи кадра управляющей программы (УП), удержание кнопки в нажатом состоянии более 3 с вызывает обнуление счетчика числа кадров УП и, как следствие, сброс записанной ранее программы, что подтверждается загоранием соответствующего индикатора на пульте управления. В автоматическом режиме кнопка SB1 служит для запуска — останова исполнения программы. Выполнение каждого последующего кадра УП в пошаговом автоматическом режиме осуществляется после нажатия кнопки SB1. Конечные выключатели (датчики) робота, обмотки электропневмоклапанов и пульт управления подключены к соответствующим входам и выходам контроллера.

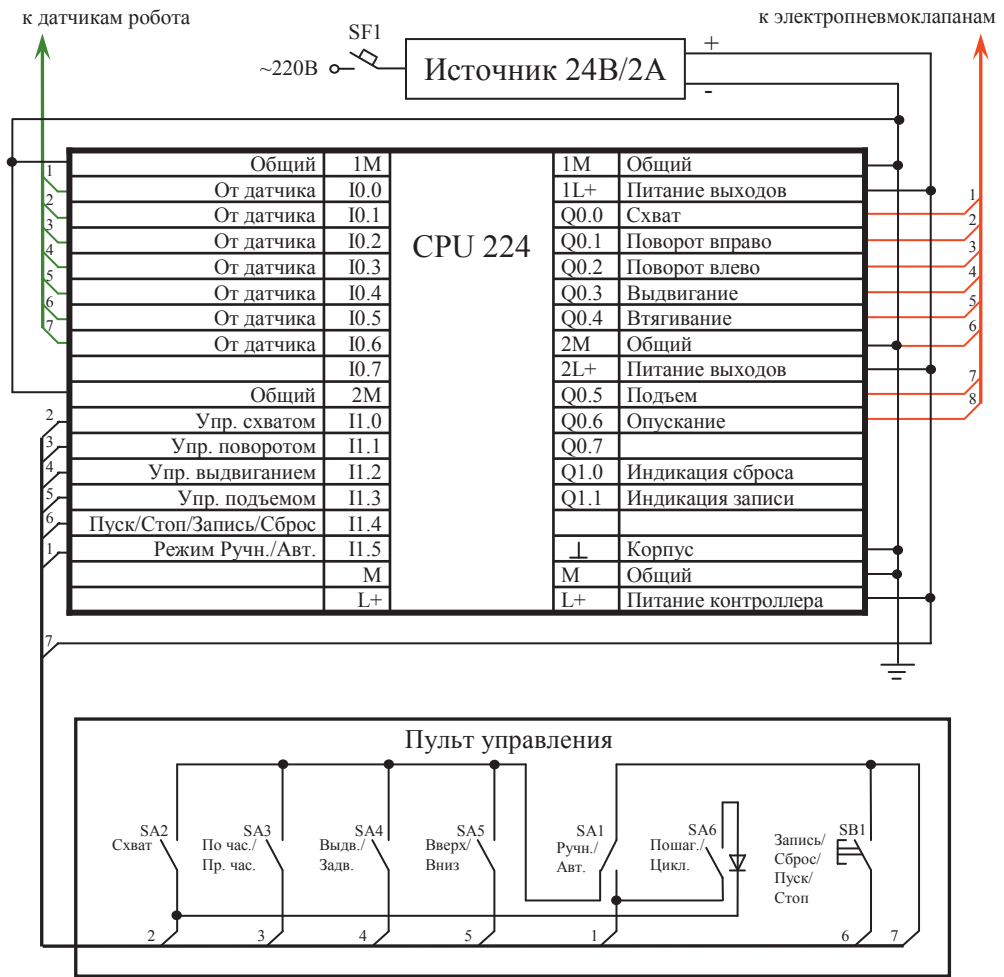


Рис. 7. Схема системы управления роботом МП-9С

Структура программного обеспечения контроллера показана на рис. 8, 9. Рабочая программа контроллера состоит из основной программы и ряда подпрограмм, которые в совокупности обеспечивают возможность программирования робота в режиме обучения и автоматического исполнения запрограммированных циклов работы.

В самом начале основной программы происходит чтение датчиков робота, далее, в зависимости от заданного режима работы (ручной или автоматический), вызывается либо подпрограмма ручного режима, либо подпрограмма автоматического режима.

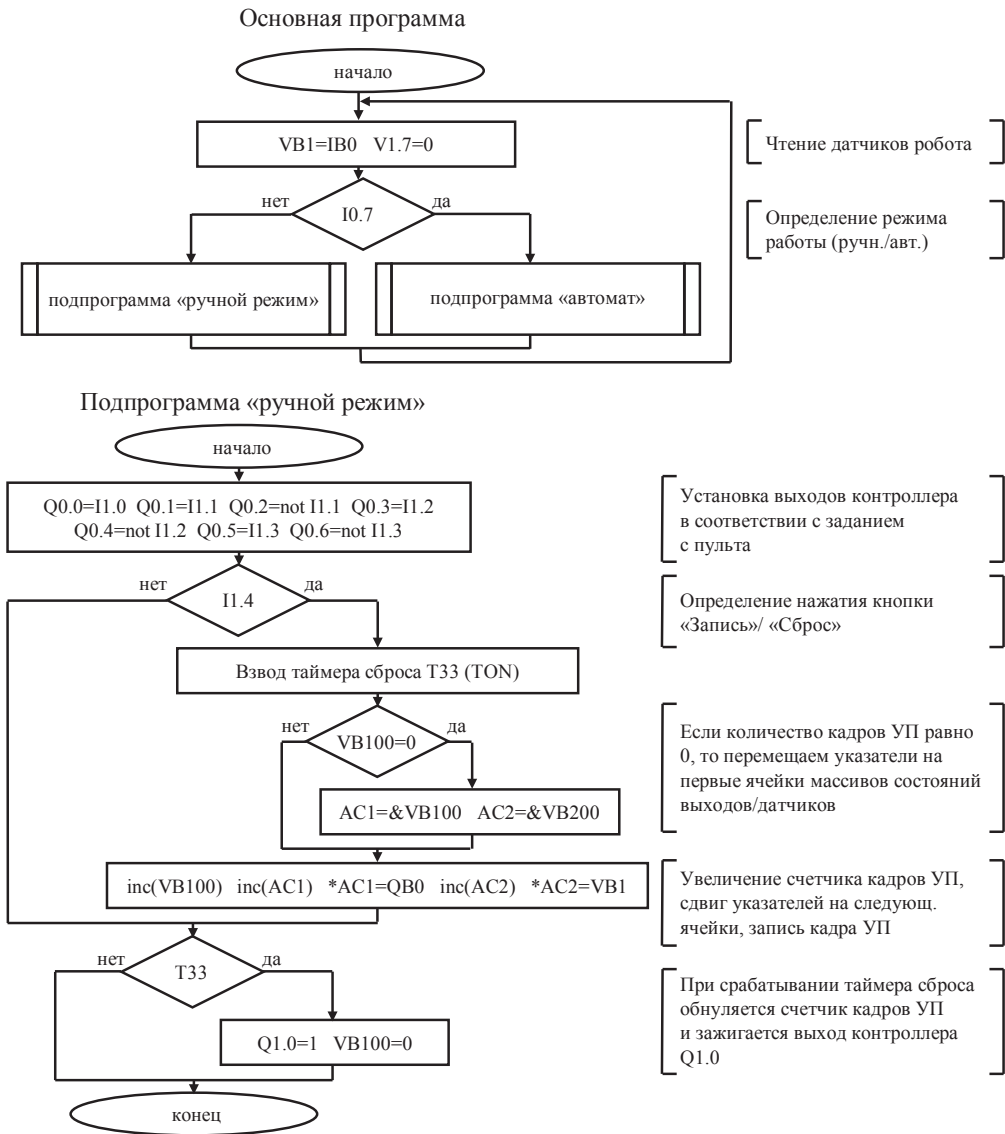


Рис. 8. Блок-схема основной программы и подпрограммы «Ручной режим»

В подпрограмме ручного режима осуществляется установка выходов контроллера, подключенных к обмоткам электропневмоклапанов, в соответствии с состоянием тумблеров SA2...SA5. После этого проверяется состояние кнопки SB1 и, если она нажата, запускается таймер сброса, в результате чего производится запись кадра УП (т. е. в памяти контроллера сохраняется состояние датчиков робота и выходов контроллера).

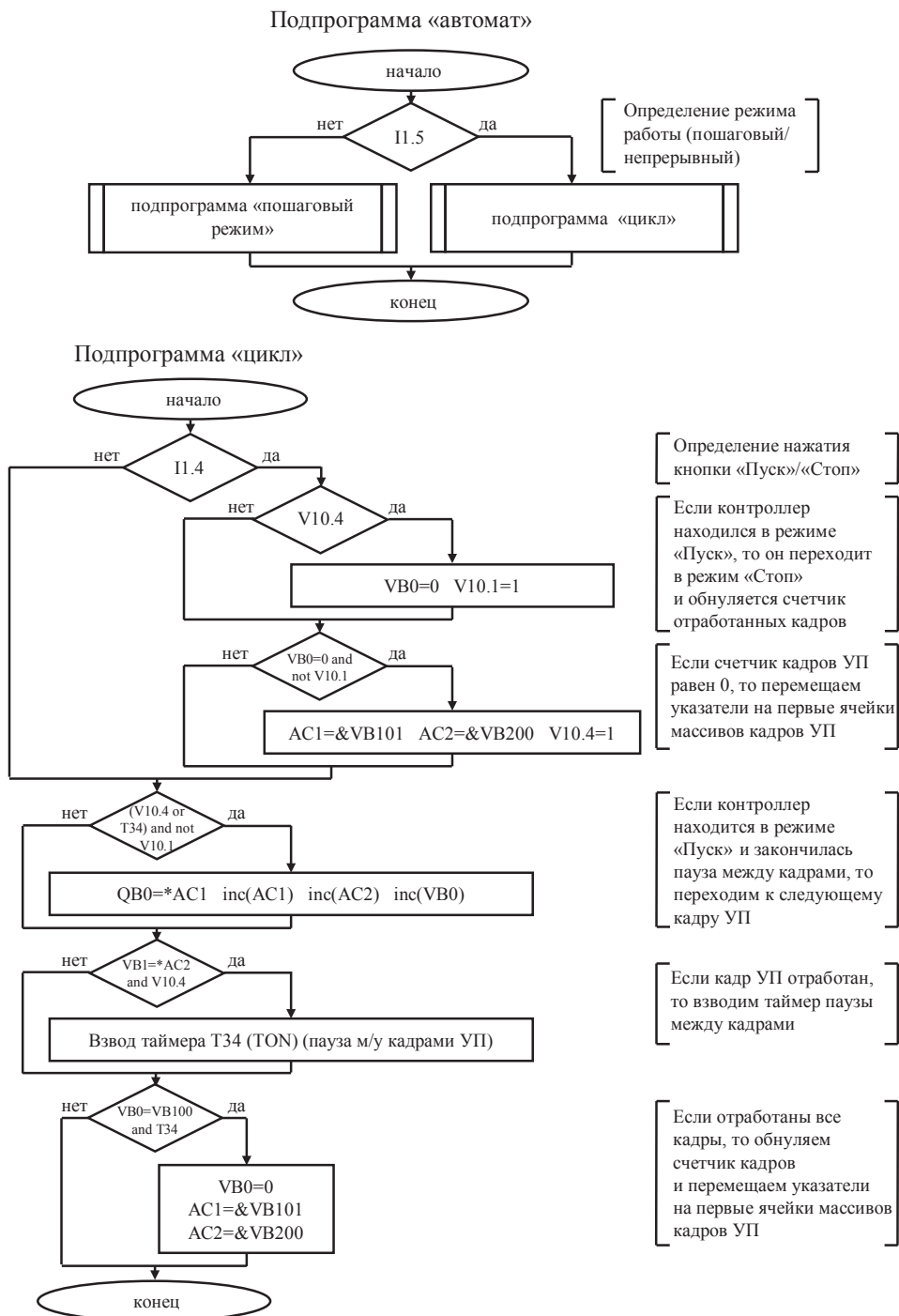


Рис. 9. Блок-схемы подпрограмм «Автомат» и «Цикл»

Следующим этапом подпрограммы ручного управления является проверка бита таймера сброса и, если он оказывается во взведенном состоянии, обнуление переменной, хранящей число кадров УП, (сброс программы).

Кадр управляющей программы предложено формировать из двух байтов (рис. 10): байта, хранящего информацию о состоянии выходов контроллера (т. е. задание на положение звеньев робота), и байта с информацией о состоянии датчиков робота, фиксирующих действительное положение звеньев робота.

Задачей подпрограммы автоматического режима работы является запуск подпрограммы пошагового режима или циклового (непрерывного) режима в зависимости от положения тумблера SA6.

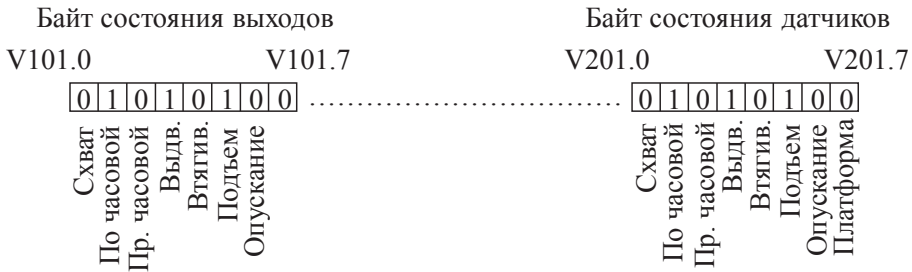


Рис. 10. Формат кадра управляющей программы

Подпрограмма пошагового режима работы функционирует следующим образом: при каждом нажатии кнопки SB1 состояния выходов выставляются в соответствии с кадром УП, а указатели AC1 (указатель на байт состояния выходов) и AC2 (указатель на байт состояния датчиков) перемещаются на следующий кадр УП (или на первый, если программа закончилась).

Подпрограмма циклового (непрерывного) режима обеспечивает непрерывное выполнение кадров УП после нажатия на кнопку SB1. Повторное нажатие кнопки SB1 приводит к останову выполнения программы. Исполнение программы всегда начинается с первого кадра УП. Контроль выполнения кадров осуществляется путем сравнения фактического состояния датчиков робота с состоянием, заданным в кадре УП. Кадр считается выполненным, если состояние датчиков робота совпадает с состоянием, заданным в кадре УП.

В программе использованы следующие переменные: VB0 — количество выполненных кадров; VB1 — состояние датчиков робота; VB100 —

число кадров программы; АС1, АС2 — указатели на ячейку с состоянием выходов и датчиков; V10 — промежуточная переменная, необходимая для закрепления функций «Пуск» — «Стоп» за одной кнопкой.

**Задача 1.** Разработать программу, управляющую роботом в ручном режиме.

**Задача 2.** Разработать программу, управляющую роботом в ручном и автоматическом режимах.

**Задача 3.** Разработать программу, управляющую роботом в ручном и автоматическом режимах. При программировании движений робота, вход I1.7 должен задавать паузу, выдерживаемую после выполнения кадра в автоматическом режиме. Длительность паузы должна составлять 0,5 с при I1.7 = 0, или 3 с при I1.7 = 1. В автоматическом режиме работы вход I1.7 не должен влиять на выполнение программы.

**Задача 4.** Разработать программу, управляющую роботом в ручном и автоматическом режимах. Вход I0.6 при работе в автоматическом режиме должен разрешать повтор выполнения технологической задачи роботом. Если при отработке кадров УП вход I0.6 переключился в ноль, то УП выполняется до последнего кадра, но переход к первому кадру и повторный запуск не осуществляются, пока не появится сигнал на входе I0.6.

**Задача 5.** Разработать программу, управляющую роботом в ручном и автоматическом режимах. Вход I0.7 при работе в автоматическом режиме должен разрешать выполнение технологической задачи роботом. Если при отработке кадров УП вход I0.7 переключился в ноль, то выполнение УП останавливается. Отработка УП может продолжаться только после появления сигнала на входе I0.7 и команды «Пуск» с пульта управления робота.

## 2.4.2. Позиционная система управления роботом ТУР-10

Промышленный робот ТУР-10 (технологический универсальный робот — 10 кгс) является представителем электромеханических роботов, работающих в сферической системе координат, и предназначен для обслуживания ряда технологических процессов. Манипулятор робота способен осуществлять захват, перенос и установку детали по заданным координатам рабочей зоны, а также может быть использован для перемещения технологического инструмента (краскопульты, клещей точечной сварки и др.) в пределах рабочей зоны.



Техническая характеристика робота ТУР-10:

- число степеней подвижности — 5;
- грузоподъемность — 10 кгс;
- тип управляющего устройства — позиционное;
- число одновременно управляемых движений — 5.

Схема робота приведена на рис. 11. Основными узлами робота являются: механизм поворота вокруг вертикальной оси, вертикальное звено (плечо), горизонтальное звено (рука), звено «сгиба» (кисть), звено вращения кисти. Схват в данной работе не используется. Звенья робота последовательно соединены друг с другом. В качестве привода звеньев манипулятора используется мотор-редуктор, содержащий двигатель постоянного тока и волновой редуктор. Передача движения к звеньям осуществляется через кривошипно-шатунные механизмы и цепные передачи.

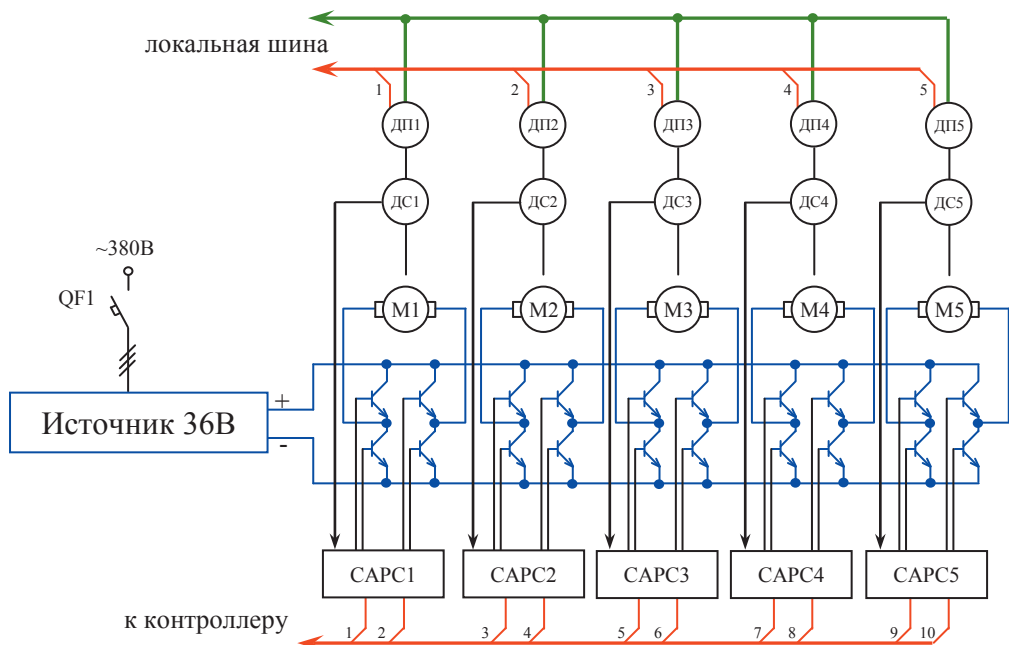


Рис. 11. Схема силовой части робота ТУР-10

Электроприводы выполнены на основе двигателей постоянного тока, питаемых от широтно-импульсных преобразователей с общим источником постоянного тока. Аналоговая система управления электроприводами состоит из однократно интегрирующих САР скорости

с тахогенераторными обратными связями. Задание на скорость формирует контроллер путем выдачи на входы САР аналогового сигнала через модули ЦАП. Информация о положении звеньев робота поступает в контроллер через модули дискретных входов от кодовых датчиков положения, формирующих 15-разрядный двоичный код. Датчики имеют входы чтения, активируемые сигналами контроллера при опросе датчика для получения информации о положении соответствующего звена, поэтому выходы датчиков объединены в локальную шину.

На пульте управления роботом (рис. 12) размещены: тумблер выбора режима работы SA7 (ручной — автоматический), тумблер выбора скорости перемещения звеньев в ручном режиме SA6 (пониженная скорость — нормальная скорость), многофункциональная кнопка SB1 (пуск — стоп — запись — сброс) и кнопки управления приводами робота в ручном режиме работы SA1...SA5 (вперед — назад). Кратковременное нажатие многофункциональной кнопки в ручном режиме работы приводит к записи кадра управляющей программы (УП). Удержание этой кнопки в нажатом состоянии более 3 с вызывает обнуление счетчика кадров УП и, как следствие, сброс записанной ранее программы, что подтверждается загоранием соответствующего индикатора на пульте управления. В автоматическом режиме работы эта кнопка служит для запуска — останова исполнения программы.

Структура программного обеспечения системы управления робота приведена на рис. 13...15. В основной программе (рис. 13), в зависимости от положения тумблера выбора режима работы SA7, вызывает-ся либо подпрограмма ручного режима, либо подпрограмма автоматического режима. До перехода в ручной режим происходит блокировка прерываний, это необходимо для останова расчета регуляторов положения, т. к. они реализованы в таймерном прерывании. В ручном режиме работы (рис. 14) осуществляется непрерывный опрос датчиков положения; их текущее состояние запоминается в буфере V1...V10. Нажатие кнопок управления приводами робота в этом режиме запускает соответствующие приводы в нужном направлении; выбор значения скорости осуществляется тумблером SA6 для всех приводов сразу. При кратковременном нажатии на кнопку SB1 происходит запись кадра УП: копирование буфера состояния датчиков в кадр УП, увеличение счетчика числа кадров VB100 на единицу и перемещение указателя AC1 на следующий кадр. Удержание кнопки SB1 более 3 с перемещает указатель на первый кадр.



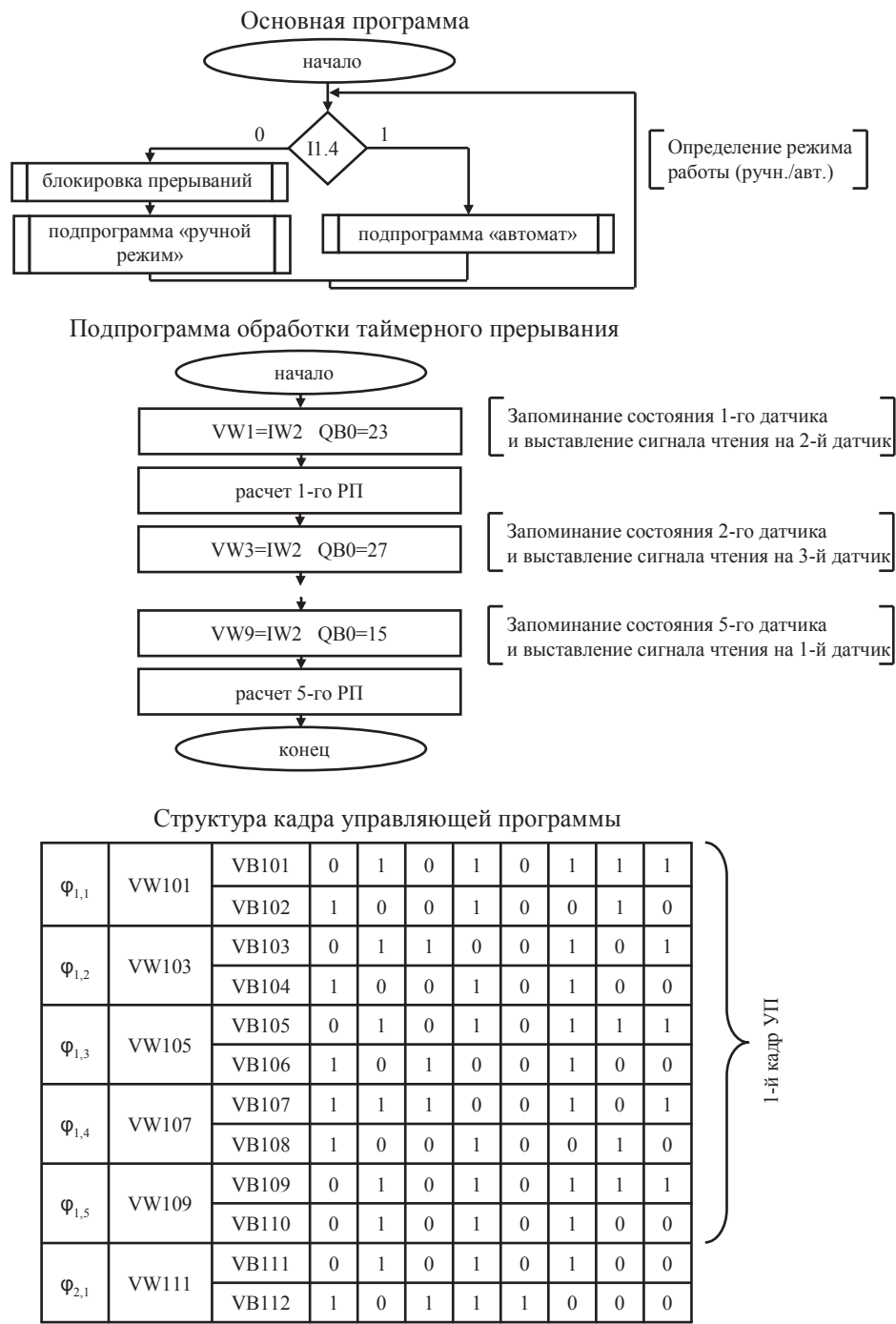


Рис. 13. Структура основной программы, подпрограммы обработки таймерного прерывания и кадра УП

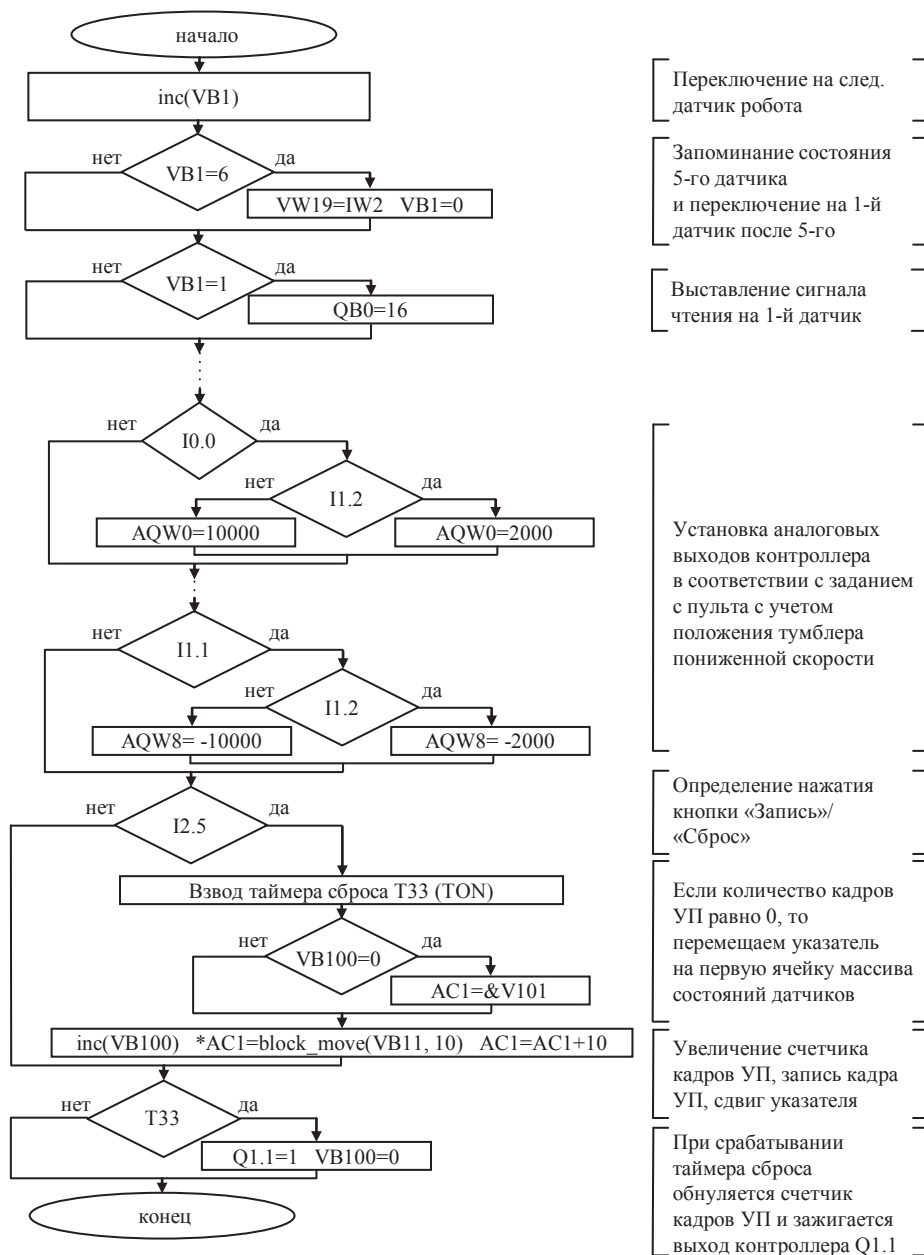


Рис. 14. Блок-схема подпрограммы «Ручной режим»

В автоматическом режиме работы (рис. 15) проверяется состояние кнопки SB1 (пуск — стоп); если она нажата, то, в зависимости от значения промежуточной переменной VB20, производится либо останов

робота (блокировка прерываний, выдача нулевых заданий на САР), либо запуск робота в работу с первого кадра программы (деблокировка прерываний, установка указателя кадров АС1 на первый кадр УП). Далее вычисляются ошибки всех контуров положения, и если все ошибки оказываются меньше допустимых, то кадр УП считается выполненным и запускается таймер Т34 для обеспечения паузы между кадрами.

После выполнения этих операций проверяется состояние таймера паузы Т34; если он сработал, то осуществляется переход к следующему кадру УП, а если программа закончилась, то к первому кадру.

Подпрограмма обработки таймерного прерывания включает в себя операции опроса датчиков положения, расчета контуров положения и выставления заданий на скорость. Опрос датчика положения заключается в выдаче сигнала запроса чтения на датчик путем установления соответствующего выхода контроллера (Q0.0...Q0.4) в 0. При этом остальные выходы контроллера должны находиться во взведенном состоянии, чтобы избежать попытки выдачи кода несколькими датчиками одновременно. Осуществить опрос датчика удобно, записав определенный код по адресу QB0. Данные коды и соответствующие им опрашиваемые датчики приведены ниже.

Контролируемый узел:	поворот кисти	Код: 23
	наклон кисти	27
	рука	30
	плечо	15
	поворот	29

Кадр УП занимает в памяти контроллера 10 байт (по 2 байта на каждое звено) и хранит в себе информацию о заданном положении всех звеньев робота.

В программе использованы следующие переменные: VB0 — номер текущего кадра; VB1...VB10 — переменные, хранящие текущие состояния датчиков робота; Т34 — таймер с задержкой включения, используемый для организации задержки между кадрами; VB100 — число кадров управляющей программы; VB101...VB2500 — массив состояний датчиков робота (кадры УП); АС1 — указатель, используемый для перемещения по кадрам УП; VB20 — переменная, необходимая для закрепления функций «Пуск» — «Стоп» за одной кнопкой; VB31...VB40 — буфер, используемый в автоматическом режиме для хранения текущего кадра УП (задание на положение).

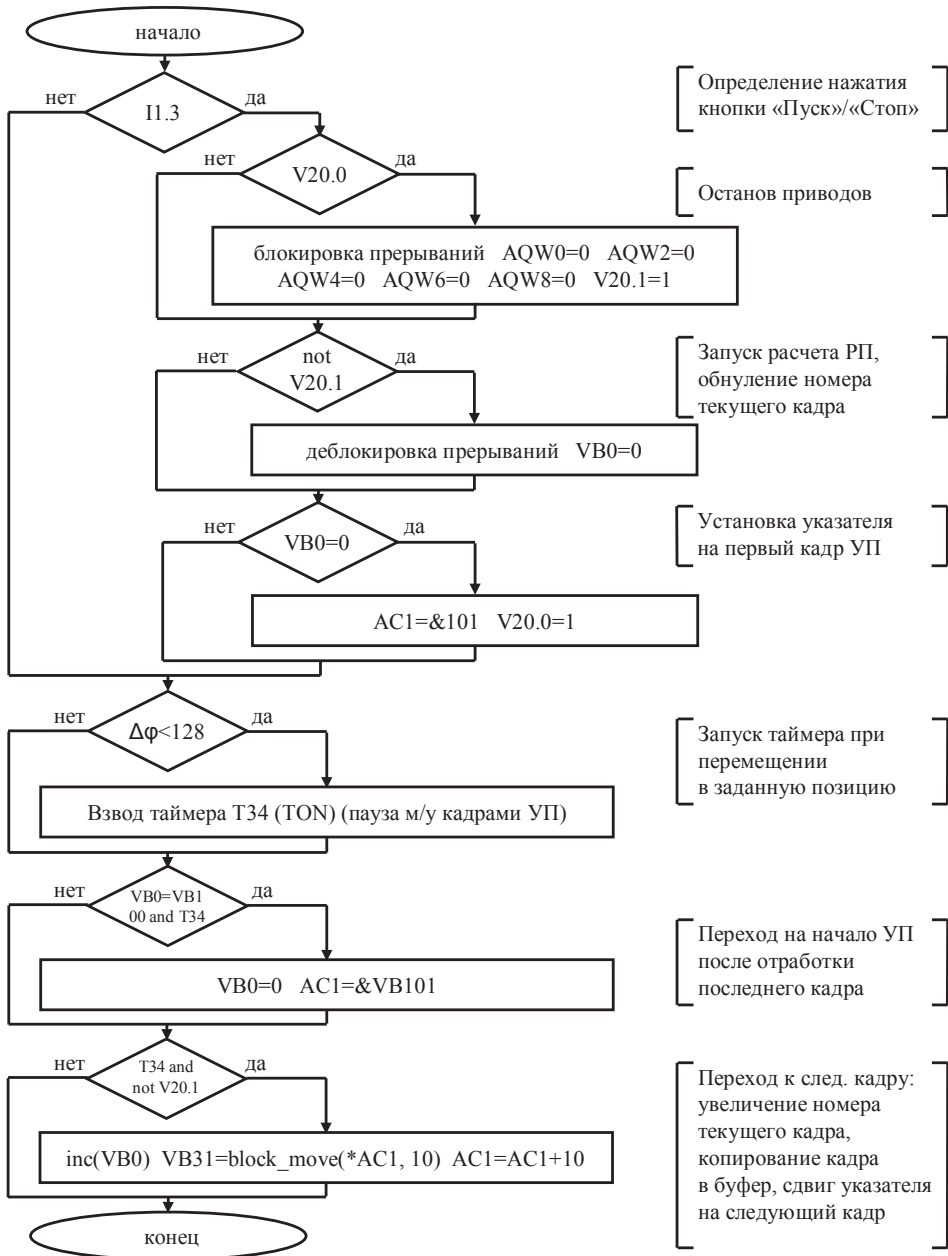


Рис. 15. Блок-схема подпрограммы «Автомат»

**Задача 1.** Создайте программу, управляющую роботом в ручном режиме. В качестве значения сигнала задания на скорость для приводов в режиме «Пониженная скорость» используйте числовой код +500

(движение «Вперед») или  $-500$  (движение «Назад»), в режиме «Нормальная скорость» используйте код  $+1500$  или  $-1500$  соответственно.

**Задача 2.** Создайте программу, управляющую роботом в ручном и автоматическом режимах.

**Задача 3.** Создайте программу, управляющую роботом в ручном и автоматическом режимах. Вход  $I1.5$  при программировании движений робота должен использоваться как вход, задающий паузу, выдерживаемую после отработки кадра. Длительность паузы должна составлять  $0,5$  с при  $I1.5 = 0$ , или  $3$  с при  $I1.5 = 1$ . В автоматическом режиме вход  $I1.5$  не должен влиять на работу программы.

**Задача 4.** Создайте программу, управляющую роботом в ручном и автоматическом режимах. Вход  $I1.5$  при работе в автоматическом режиме должен использоваться как вход, разрешающий выполнение программы. Если при выполнении кадров УП вход  $I1.5$  переключился в ноль, то УП выполняется до последнего кадра, но переход к первому кадру и повторный запуск не осуществляется.



---

## Библиографический список

---

1. Эмулятор программируемого контроллера : свидетельство РФ : № 2011616815 : опубл. 01.09.2011 / Зюзев А. М., Нестеров К. Е. Москва : Роспатент, 2011. 16 с.
2. Комплекс «Имитаторы устройств электроавтоматики станков»: свидетельство РФ : № 2011618825 : опубл. 14.11.2011 / Зюзев А. М., Нестеров К. Е. Москва : Роспатент, 2011. 8 с.
3. Петров И. В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования / под ред. проф. В. П. Дьяконова. Москва : СОЛОН-ПРЕСС, 2010. 256 с.
4. Программируемый контроллер S7—200. Системное руководство C79000-G7076-C233—02 [Электронный ресурс]. Вып. 2. Режим доступа: <http://www.automation.siemens.com>]. Загл. с экрана.
5. Шишов О. В. Современные технологии промышленной автоматизации : учебное пособие / О. В. Шишов. Москва ; Берлин : Ди-рект-Медиа, 2015. 368 с.

*Учебное издание*

**Нестеров Константин Евгеньевич**  
**Зюзов Анатолий Михайлович**

**ПРОГРАММИРОВАНИЕ  
ПРОМЫШЛЕННЫХ КОНТРОЛЛЕРОВ**

Редактор И. В. Меркурьева  
Верстка О. П. Игнатьевой

Подписано в печать 28.06.2019. Формат 70×100/16.  
Бумага офсетная. Цифровая печать. Усл. печ. л. 7,7.  
Уч.-изд. л. 4,6. Тираж 40 экз. Заказ 206

Издательство Уральского университета  
Редакционно-издательский отдел ИПЦ УрФУ  
620049, Екатеринбург, ул. С. Ковалевской, 5  
Тел.: +7 (343) 375-48-25, 375-46-85, 374-19-41  
E-mail: rio@urfu.ru

Отпечатано в Издательско-полиграфическом центре УрФУ  
620083, Екатеринбург, ул. Тургенева, 4  
Тел.: +7 (343) 358-93-06, 350-58-20, 350-90-13  
Факс: +7 (343) 358-93-06  
<http://print.urfu.ru>



