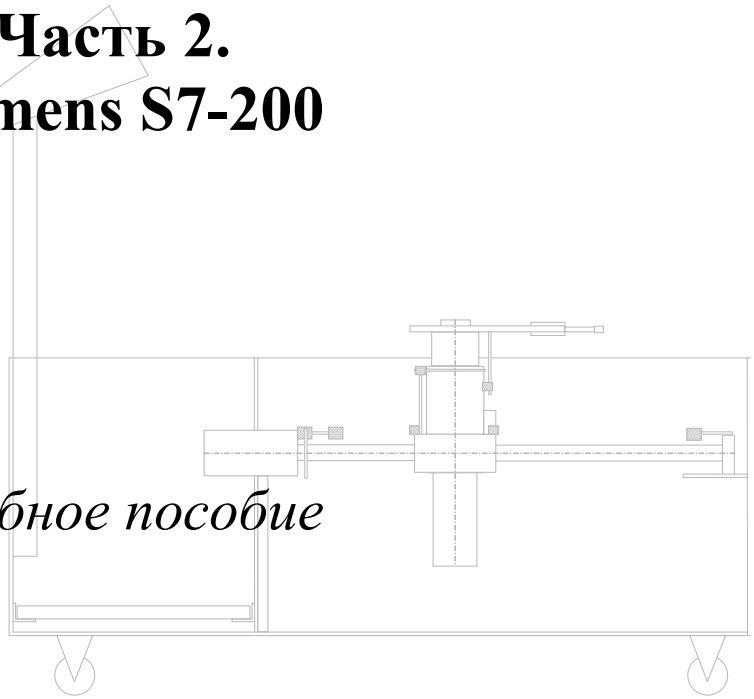
Федеральное агентство по образованию

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Серия «Учебно-методический комплекс дисциплины»

А.Н. Рыбалев

Программируемые логические контроллеры и аппаратура управления: лабораторный практикум
Часть 2.
Siemens S7-200



Учебное пособие

Благовещенск
2009

ББК ???

Печатается по решению
редакционно-издательского совета
энергетического факультета
Амурского государственного
университета

Рыбалев А.Н. Программируемые логические контроллеры и аппаратура управления: лабораторный практикум. Часть 1. Siemens S7-200. Учебное пособие. – Благовещенск: Амурский гос. ун-т, 2009.

Пособие предназначено для студентов специальности 220301 «Автоматизация технологических процессов и производств», изучающих дисциплины «Технические средства автоматизации», «Автоматическое управление энергетическими установками», «Автоматизация технологических процессов» и выполняющих лабораторные работы по данным дисциплинам. Может быть также использовано при выполнении курсовых и дипломных проектов.

Рецензенты: А.В. Бушманов, заведующий кафедрой информационных и управляющих систем АмГУ, канд. техн. наук, доцент;
А.И. Яшин, главный инженер Благовещенской ТЭЦ, канд. техн. наук, доцент.

В авторской редакции

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Учебное пособие содержит теоретические сведения, задания и контрольные вопросы к лабораторным работам по следующим дисциплинам специальности 220301 «Автоматизация технологических процессов и производств»:

«Технические средства автоматизации»;

«Автоматическое управление энергетическими установками»;

«Автоматизация технологических процессов».

Пособие может также использоваться как базовый материал для разработки лабораторных работ по дисциплинам «Микропроцессорные системы управления» и «Интегрированные системы проектирования и управления».

Лабораторные работы «начального уровня», посвященные знакомству с аппаратурой и программным обеспечением, в упрощенном виде могут выполняться в рамках изучения дисциплины «Практикум по контрольно-измерительным приборам и автоматике», изучаемой студентами специальности 220301 на III курсе. Методические указания к данным работам являются переработанными версиями разделов «Первые шаги» руководств к контроллерам и системам программирования.

Пособие состоит из трех частей.

В настоящей второй части пособия рассматриваются лабораторные работы с применением контроллера Siemens S7-200. В первых двух работах студенты знакомятся с системой программирования Siemens Step 7-Micro/Win, осваивают языки программирования контроллеров LD, FBD, ST в реализации данной системы. В следующих четырех работах рассматриваются системы контроллерного управления частотным асинхронным электроприводом на основе преобразователя частоты ABB ACS 300, в том числе система программно-логического управления механизмом циклического действия, системы измерения и регулирования скорости и угла поворота привода с применением скоростного счетчика Siemens S7-200. В последних трех работах контроллер используется для управления лабораторным электромеханическим роботом-манипулятором с тремя кинематическими парами и механизмом, имитирующим рабочий орган. В ходе выполнения лабораторных работ разрабатываются системы программного управления движением с контролем положения, самообучения по результатам ручного управления, компьютерного (супервизорного) управления с применением SCADA-системы Trace Mode.

В приложении приведено краткое техническое описание используемых при проведении лабораторных работ приборов и устройств. Материал дополняет теоретические сведения, приведенные в каждой работе, и может использоваться при выполнении курсовых и дипломных проектов по специальности 220301.

Разработка учебного пособия «Программируемые логические контроллеры и аппаратура управления» является очередным этапом многолетней работы, проводимой на кафедре автоматизации производственных процессов и электротехники по совершенствованию лабораторной базы специальности 220301.

Стенды Siemens S7-200 и ABB ACS 300 подарены кафедре Инновационно-техническим центром АмГУ по инициативе его бывшего руководителя, к сожалению ныне покойного, Козлова А.В. На кафедре стенды были модифицированы под потребности в лабораторных работах.

Лабораторный робот-манипулятор был сконструирован покойным профессором Контесом В.Д. В разные годы под руководством Редозубова Р.Д. и Рыбалева А.Н. над модернизацией робота и созданием систем управления работали выпускники Пашин А.Ю., Ушаков А.С., Кокин Р.А., Корякин А.С., Синдеев С.С., Рахимов А.А.

Огромную работу по монтажу лабораторных стендов выполнил высококвалифицированный рабочий кафедры АПП и Э Харьков В.П. Во многом благодаря ему внешний вид и содержание стендов соответствуют самым высоким требованиям.

ЛАБОРАТОРНЫЕ РАБОТЫ

1. «Первые шаги в Siemens STEP 7- Micro/WIN»

Цель работы: получить элементарные навыки работы в системе программирования контроллеров Siemens S7 200 STEP 7- Micro/WIN, ознакомиться с основами языка программирования LAD.

Порядок работы

Вызов STEP 7-Micro/WIN

Щелкните на символе STEP 7- Micro/WIN, чтобы открыть новый проект. На рис. 1 показан новый проект.

Обратите внимание на навигационную панель. С помощью символов на навигационной панели вы можете открывать отдельные элементы проекта STEP 7-Micro/WIN. Щелкните на символе Communications на навигационной панели, чтобы вызвать диалоговое окно «Communications [Обмен данными]».

Это диалоговое окно используется для установки связей для STEP 7-Micro/WIN.

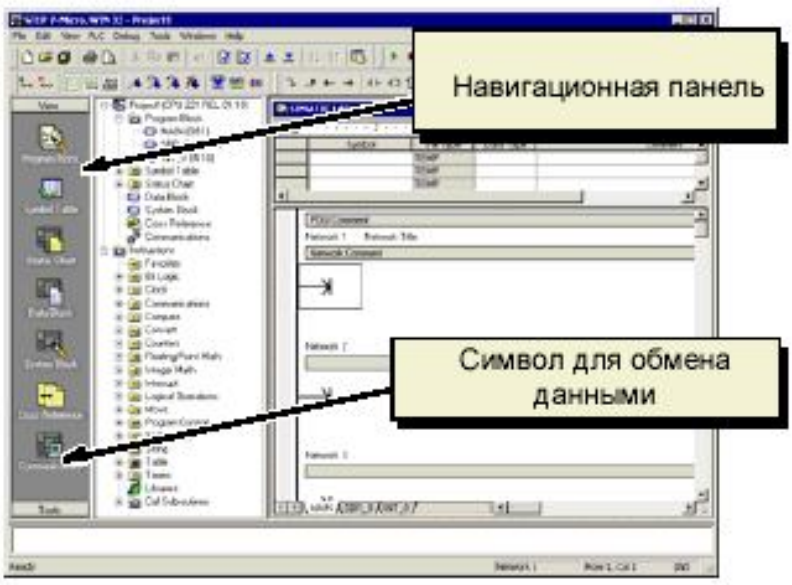


Рис. 1. Новый проект

Проверка параметров обмена данными для STEP 7-Micro/WIN

Проект-пример использует для STEP 7-Micro/WIN и преобразователя интерфейсов RS232/RS485. настройки по умолчанию. Эти настройки проверяются следующим образом:

1) проверьте, чтобы адрес PLC в диалоговом окне Communications был установлен на 2;

2) проверьте, чтобы в качестве интерфейса для сетевых параметров был установлен кабель PC/PPI (COM1 или COM2);

3) проверьте, чтобы для скорости передачи (transmission rate) было установлено значение 9.6 Кбит/с.

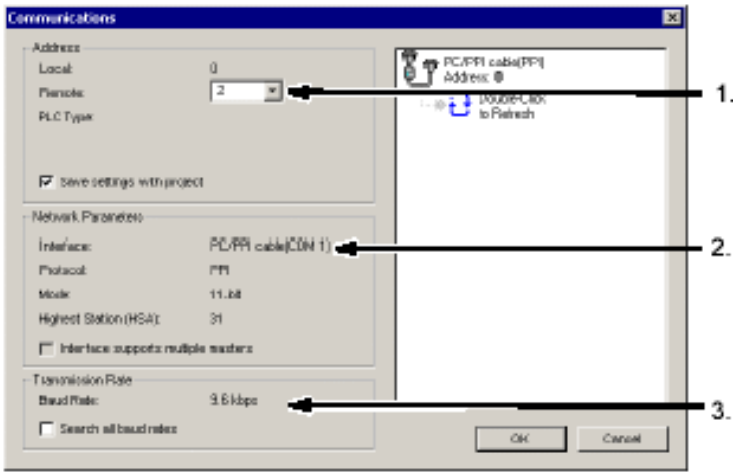


Рис. 2. Проверка параметров обмена данными

Установка связи с S7–200

В диалоговом окне Communications [Обмен данными] установите связь с CPU S7–200.

В диалоговом окне Communications щелкните дважды на кнопке Double click to refresh [Дважды щелкните для обновления]. STEP 7-Micro/WIN ищет станцию S7–200 и отображает символ CPU для подключенной станции S7–200.

Выберите S7–200 и щелкните на OK. Если STEP 7-Micro/WIN не находит ваше CPU S7–200, проверьте настройки параметров для обмена данными и повторите эти шаги. После установления связи с S7–200 вы готовы к созданию и загрузке программы-примера.

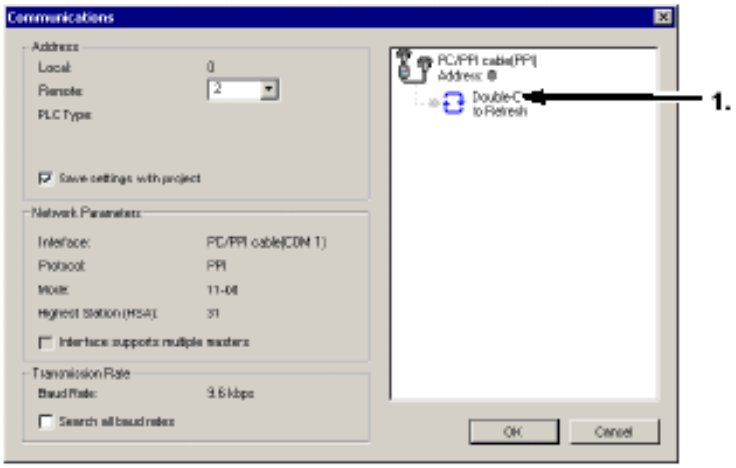


Рис. 3. Установка связи с PLC

Создание программы-примера

Ввод этого примера программы управления поможет вам понять, как просто работать со STEP 7-Micro/WIN. Эта программа содержит шесть команд в тех сегментах (Network) и образует из них очень простой таймер, сам запускается и сам себя сбрасывает.

Команды для этого примера программы введите в редакторе LAD. Следующий пример показывает всю программу в виде контактного плана (LAD) и в виде списка команд (STL, AWL). Комментарии к сегменту в STL-программе

объясняют логику для каждого сегмента. Импульсная диаграмма показывает, как программа работает.

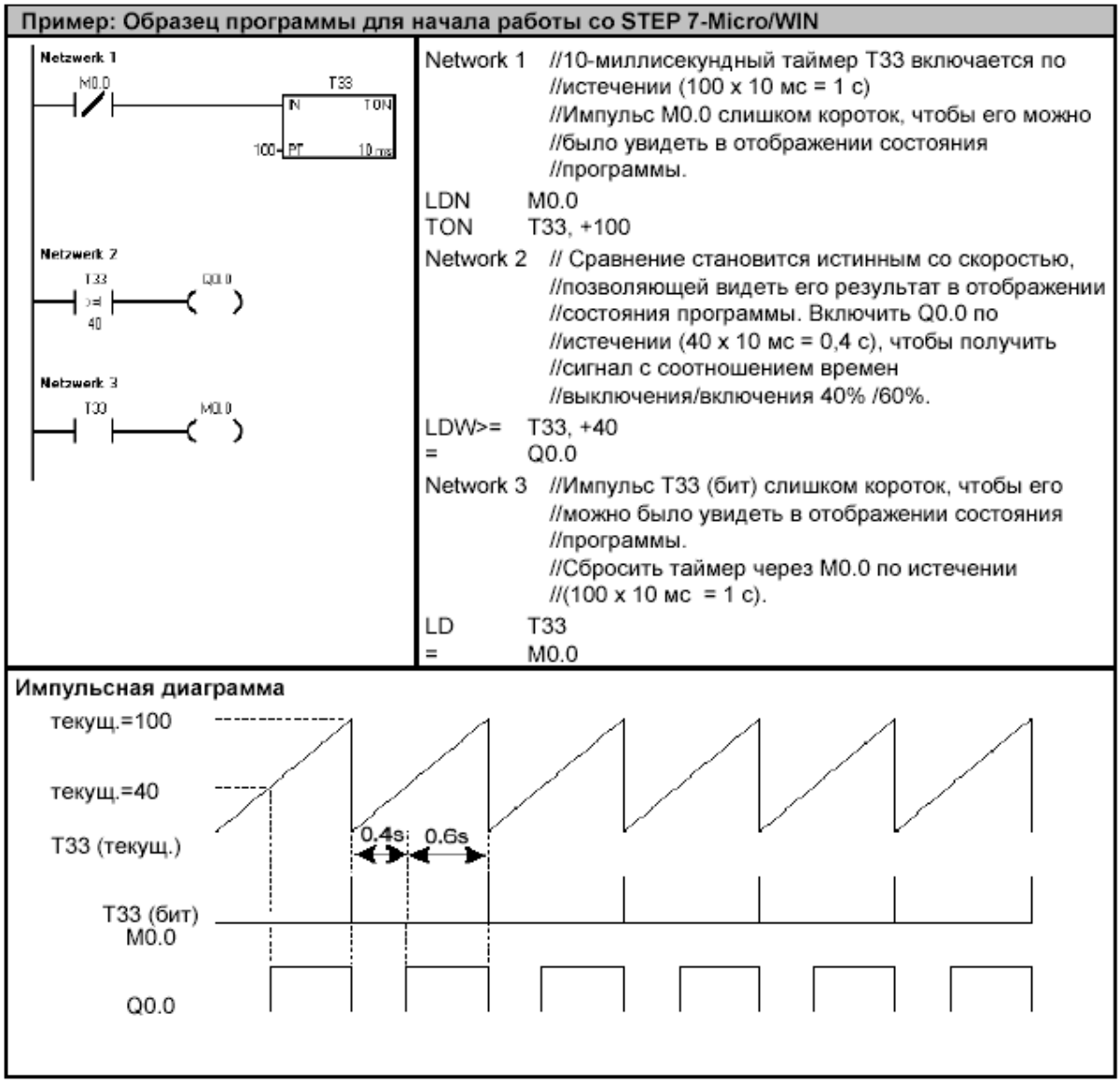


Рис. 4. Программа - пример

Вызов редактора программ

Чтобы открыть редактор программ, щелкните на символе Program Block [Программный блок], см. рис. 1. Обратите внимание на дерево команд и редактор программ. Дерево команд используется для вставки команд контактного плана (LAD) в сегменты редактора программ путем буксировки команд с помощью мыши из дерева команд в сегменты.

Символы на панели инструментов предоставляют возможность быстрого вызова команд меню.

После ввода и сохранения программы вы можете загрузить ее в S7–200.

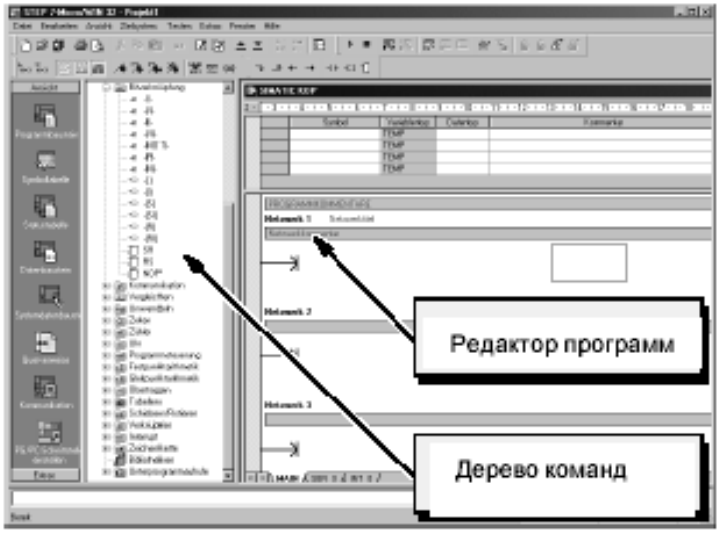


Рис. 5. Окно STEP 7- Micro/WIN

Ввод сегмента (Network) 1: Запуск таймера

Если M0.0 выключен (0), поток сигнала передается для запуска таймера. Для ввода контакта для M0.0:

1) дважды щелкните на символе BitLogic [Битовая логика] или один раз щелкните на знаке плюс (+) для отображения битовых логических операций;

2) выберите размыкающий контакт;

3) удерживая в нажатом состоянии левую кнопку мыши, перетащите этот контакт в первый сегмент;

4) щелкните на «???» над контактом и введите следующий адрес: M0.0;

5) нажмите клавишу Return, чтобы ввести адрес для контакта.

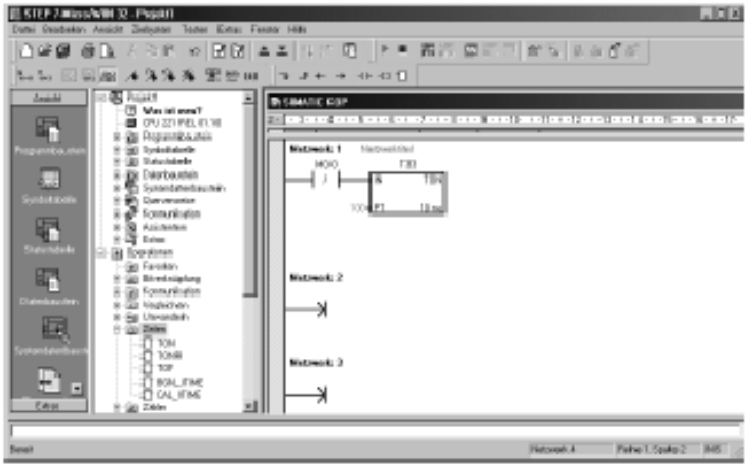


Рис. 6. Сегмент 1

Чтобы ввести таймерную команду для T33:

1) дважды щелкните на символе Timers [Таймеры], чтобы отобразить таймерные команды;

2) выберите TON (On–Delay Timer – таймер с задержкой включения);

3) удерживая в нажатом состоянии левую кнопку мыши, перетащите этот таймер в первый сегмент;

4) щелкните на «???» над таймерным блоком и введите следующий номер таймера: T33;

5) нажмите клавишу Return, чтобы ввести номер таймера и перевести фокус на параметр (PT) для задания предустановленного времени;

6) введите для предустановленного времени следующее значение: 100;

7) нажмите клавишу Return, чтобы ввести это значение.

Ввод сегмента 2: Включение выхода

Если значение таймера для T33 больше или равно 40 (40 раз по 10 миллисекунд, или 0,4 секунды), то контакт пропускает поток сигнала для включения выхода Q0.0 модуля S7–200.

Для ввода команды сравнения:

1) дважды щелкните на символе компаратора (Compare), чтобы отобразить команды сравнения. Выберите команду >=I (больше или равно для целых чисел);

2) удерживая в нажатом состоянии левую кнопку мыши, перетащите эту команду сравнения во второй сегмент;

3) щелкните на «???» над контактом и введите адрес для значения таймера: T33;

4) нажмите клавишу Return, чтобы ввести номер таймера и перевести фокус на другую величину, которая должна сравниваться со значением таймера;

5) введите следующую величину для сравнения со значением таймера: 40;

6) нажмите клавишу Return, чтобы ввести это значение.

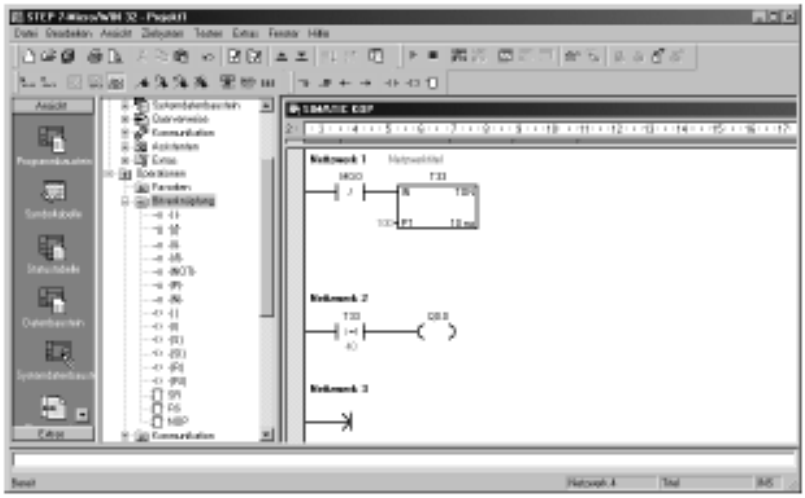


Рис. 7. Сегмент 2

Для ввода команды включения выхода Q0.0:

1) дважды щелкните на символе Bit Logic [Битовая логика], чтобы отобразить битовые логические операции, и выберите выходную катушку;

2) удерживая в нажатом состоянии левую кнопку мыши, перетащите эту катушку во второй сегмент;

3) щелкните на «???» над катушкой и введите следующий адрес: Q0.0;

4) нажмите клавишу Return, чтобы ввести этот адрес для катушки.

Ввод сегмента 3: Сброс таймера

Когда таймер достигает предустановленного значения (100) и включает таймерный бит, контакт для T33 включается. Поток сигнала от этого контакта включает бит памяти M0.0. Так как таймер активизируется нормально замкнутым контактом для M0.0, то изменение состояния M0.0 с выключенного (0) на включенное (1) сбрасывает таймер.

Чтобы ввести контакт для таймерного бита T33:

1) выберите из команд битовой логики замыкающий контакт;

2) удерживая в нажатом состоянии левую кнопку мыши, перетащите этот контакт в третий сегмент;

3) щелкните на «???» над контактом и введите адрес таймерного бита: T33;

4) нажмите клавишу Return, чтобы ввести этот адрес для контакта.

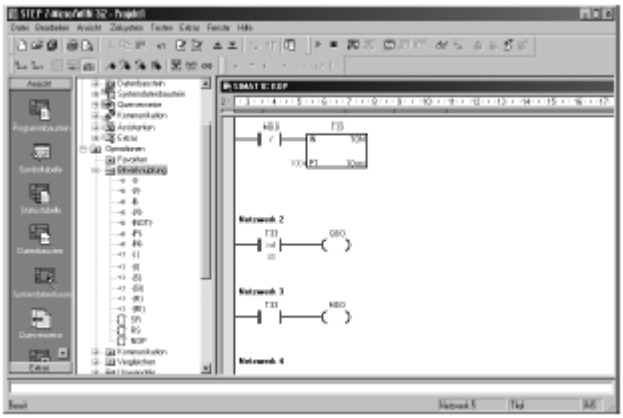


Рис. 8. Сегмент 3

Чтобы ввести катушку для включения M0.0:

1) выберите из команд битовой логики выходную катушку;

2) удерживая в нажатом состоянии левую кнопку мыши, перетащите эту выходную катушку в третий сегмент;

3) дважды щелкните на «???» над катушкой и введите следующий адрес: M0.0;

4) нажмите клавишу Return, чтобы ввести этот адрес для катушки.

Сохранение примера проекта

После ввода трех сегментов с командами вы закончили ввод программы. Когда вы сохраняете эту программу, вы создаете проект, который включает в себя тип CPU S7–200 и другие параметры. Для сохранения проекта:

1) выберите из строки меню команду **File > Save As [Файл > Сохранить как]**;

2) в диалоговом окне Save As [Сохранить как] введите имя для проекта;

3) для сохранения проекта щелкните на OK.

STEP 7-MICRO/WIN позволяет осуществлять программирование контроллера на трех языках программирования:

Ladder (LAD) – язык релейно-контактных схем (им мы воспользовались при написании программы-примера);

STL – язык инструкций, напоминающий ассемблер;

FBD – язык функциональных блоков.

С помощью меню View(Вид) Вы можете переключаться между языками. Посмотрите, как выглядит написанная Вами программа в представлении STL и FBD.

После сохранения проекта вы можете загрузить программу в S7–200.

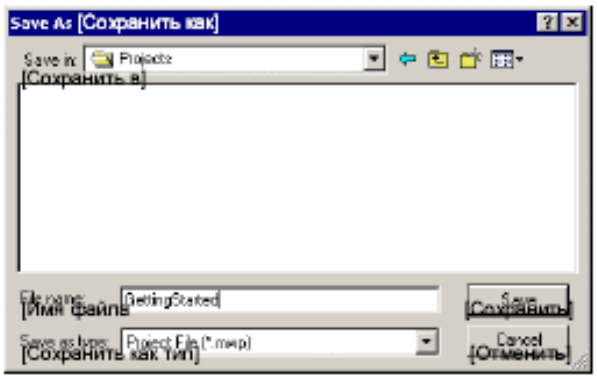


Рис. 9. Сохранение примера проекта

Загрузка программы-примера

Каждый проект STEP 7-Micro/WIN ставится в соответствие модели CPU (CPU 221, CPU 222, CPU 224, CPU 224XP или CPU 226). Если тип CPU, установленный в проекте, не соответствует подключенному CPU, то STEP 7-Micro/WIN указывает на это несоответствие и требует от вас проведения соответствующих мероприятий. Для этого примера выберите в этом случае «Continue Download [Продолжить загрузку]».

Для загрузки программы щелкните на символе Download [Загрузить], находящемся на панели инструментов, или выберите команду меню **File > Download [Файл > Загрузить]**.

Для загрузки элементов программы в S7–200 щелкните на OK.

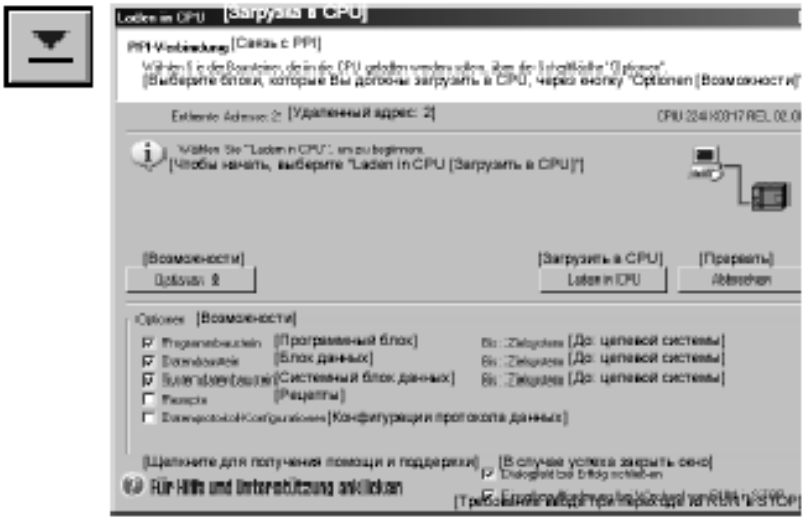


Рис. 9. Загрузка программы

Если ваш S7–200 находится в режиме RUN, то появится сообщение с требованием перевести S7–200 в STOP. Для перевода S7–200 в STOP щелкните на Yes [Да].

Перевод S7–200 в режим RUN

Чтобы STEP 7-Micro/WIN мог перевести CPU S7–200 в режим RUN, переключатель режимов S7–200 должен находиться в положении TERM или RUN. При переводе S7–200 в режим RUN S7–200 исполняет программу.

Щелкните на символе RUN, находящемся на панели инструментов, или выберите команду меню **PLC > RUN [ПЛК > RUN]**.

Щелкните на OK, чтобы изменить режим работы S7–200.

Когда S7–200 переходит в режим RUN, светодиод для Q0.0 включается и выключается по мере исполнения программы в S7–200.

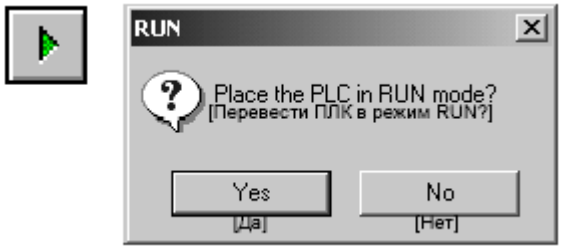


Рис. 10. Перевод S7–200 в режим RUN

Вы можете наблюдать за работой программы, выбрав команду меню **Debug > Program Status [Отладка > Состояние программы]**. STEP 7-Micro/WIN отображает значения для команд. Для остановки программы переведите S7–200 в режим STOP, щелкнув на символе STOP или выбрав команду меню **PLC > STOP [ПЛК > STOP]**.

Содержание отчета

Программа контроллера в представлениях LAD, STL и FBD и временная диаграмма ее работы.

Контрольные вопросы

1. Опишите основные интерфейсы STEP 7-MICRO/WIN.
2. На каком языке программирования изначально была написана программа-пример? Что представляет собой данный язык?
3. Опишите работу элементов программы: нормально открытого и нормально закрытого контактов, операции сравнения, катушки и таймера.
4. Сопоставьте элементам программы LAD строки инструкций на языке STL.
5. Опишите инструкции языка STL: LD , LDN , LDW , TON, =.

2. Разработка и реализации программы управления светофорами на базе Siemens S7-200

Цель работы: на основе программы, реализованной в ходе выполнения предыдущей работы, разработать и реализовать программу управления двумя светофорами (управляющими движением на перекрестке в двух взаимно перпендикулярных направлениях). (Очевидно, что другие два светофора повторяют логику работы первых двух, поэтому, управляя работой двух светофоров, мы тем самым управляем работой четырех.)

Порядок работы

«Коммутация» секций светофоров производится выходами
Q2.0 – «красный» первого светофора;
Q2.1 – «желтый» первого светофора;
Q2.2 – «зеленый» первого светофора;
Q2.3 – «красный» второго светофора;
Q2.4 – «желтый» второго светофора;
Q2.5 – «зеленый» второго светофора.

Цикл работы светофоров следующий, см. табл. №1 (для ускорения работы программы не будем придерживаться реальных временных параметров):

Таблица 1. Цикл работы светофоров

№	1	2	3	4	5
Интервал:	2 сек.	1 сек.	2 сек.	1 сек.	2 сек.
1-ый светофор:	красный	красный, желтый	зеленый	желтый	красный
2-ой светофор:	зеленый	желтый	красный	красный, желтый	зеленый

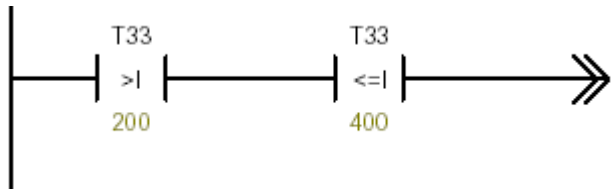
Время полного цикла составляет, таким образом, 6 сек.

При программировании следует пользоваться рекомендациями:

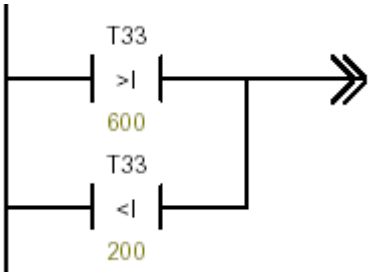
1. Network1 и Network3 (см. предыдущую работу), отвечающие за запуск и сброс таймера, можно оставить без изменения, за исключением предустановленного значения таймера, которое нужно привести в соответствие со временем полного цикла.

2. «Вместо» Network2 необходимо реализовать логику включения секций светофоров, используя операции сравнения «меньше», «больше», «меньше или равно», «больше или равно». При этом если реализуется логика «И», «контакты» должны располагаться последовательно, если логика «ИЛИ», – параллельно.

Примеры:



– если (T33 > 200) **и** (T33<=400)….



– если (T33 > 600) **или** (T33<200)….

Для реализации логики можно использовать несколько секций, например, Network2, Network4, Network5 и т. д.

Содержание отчета

Программа контроллера в представлениях LAD, STL и FBD и временная диаграмма ее работы.

Контрольные вопросы

1. Опишите работу секций программы в представлении LAD.
2. Опишите работу секций программы в представлении STL.
3. Опишите работу секций программы в представлении и FBD.

3. Разработка и реализации программы управления частотно-управляемым электроприводом механизма циклического действия

Задание: разработать программу управления частотно-управляемым электроприводом на основе преобразователя частоты ABB ACS300

Теоретические сведения

Лабораторный стенд «Частотно-управляемый электропривод на основе ПЧ ABB ACS300»

Лабораторный стенд «Частотно-управляемый электропривод на основе ПЧ ABB ACS300» (рис. 1) имеет в своем составе асинхронный двигатель, преобразователь частоты ABB ACS300 и органы управления.

Рис. 1. Частотно-управляемый электропривод
на основе ПЧ ABB ACS300

Органы управления (1) состоят из пяти выключателей и потенциометра. Выключатели подключены к дискретным входам преобразователя частоты DI1 – DI5 (рис. 2). Потенциометр служит для ручного задания скорости вращения и подключен контактам 1– 3, как показано на рис. 2.

В результате модернизации стенда все цепи управления преобразователем дополнительно выведены на гнезда, расположенные в коробке (2). Таким образом, появилась возможность управления преобразователем от внешнего устройства (в нашем случае контроллера Siemens S7-200). Сопряжение стенда с контроллером осуществляется посредством лабораторных проводов.

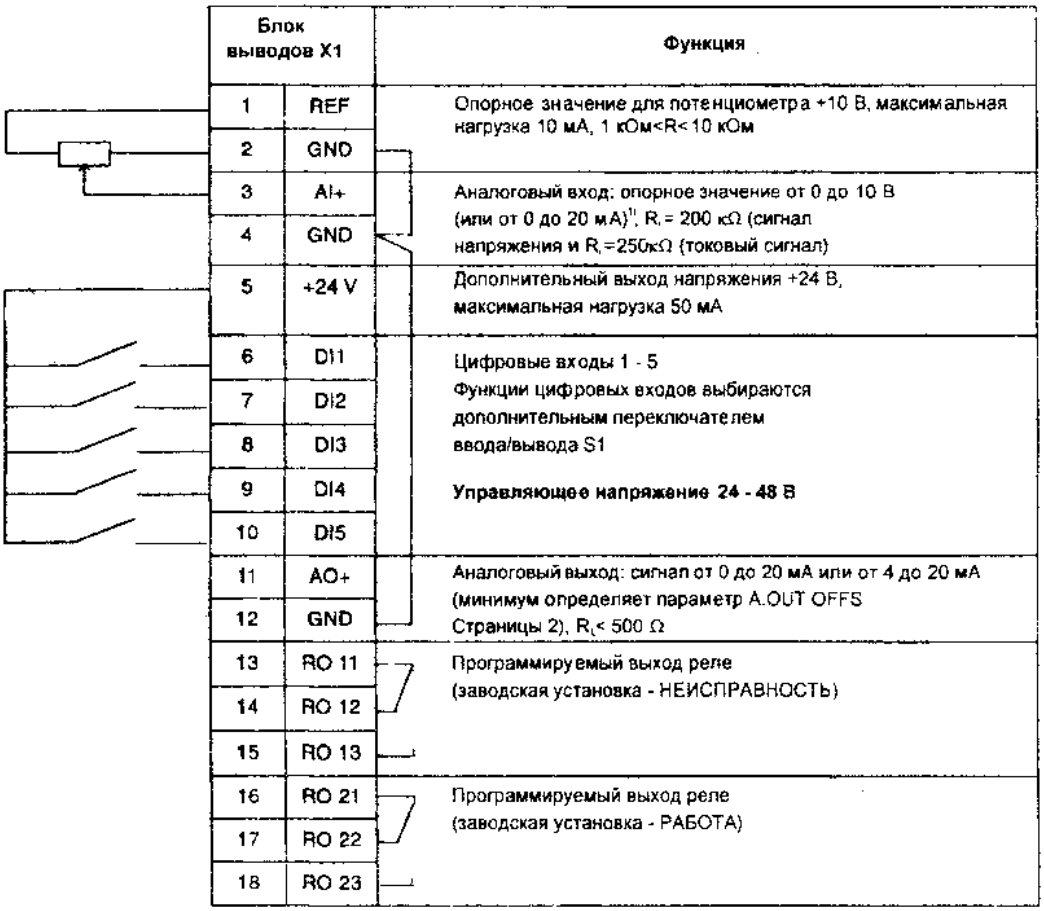


Рис. 2. Внешние цепи преобразователя частоты

Реакция ПЧ на сигналы, поступающие на его дискретные входы, определяется положением специального переключателя (рис 3), расположенного под панелью управления и задающего режим управления ПЧ, а также параметрами программирования. В данной работе используется стандартное распределение («Стандартный режим»), устанавливаемое на заводе изготовителе, согласно которому

вход DI1 отвечает за пуск/останов привода (замыкание контакта на рис. 2 приводит к пуску);

вход DI2 отвечает за реверс, т.е. изменение направления вращения (замыкание контакта на рис. 2 приводит к реверсу);

вход DI3 отвечает за выбор первой фиксированной скорости;

вход DI4 отвечает за выбор второй фиксированной скорости;

вход DI5 отвечает за выбор скорости развертки частоты при разгоне и торможении.

При одновременной активации входов DI3 и DI4 производится выбор третьей фиксированной скорости. Если оба входа неактивны, скорость регулируется потенциометром.

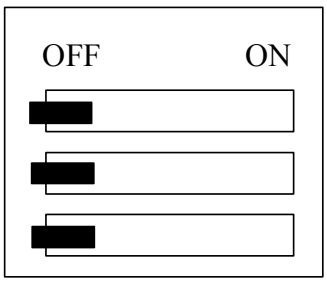


Рис.3. Положение переключателя в стандартном режиме

Имеется два варианта стандартного режима: собственно «Стандартный режим» и «Стандартный режим 2». Выбор между ними осуществляется с помощью параметра программирования PARAM SET страницы параметров PAGE3. Режимы отличаются лишь назначением входа DI5. В нашем случае следует убедиться, что PARAM SET = 1.

Задание фиксированных скоростей осуществляется с помощью параметров CON f1, CON f2, CON f3 станицы параметров PAGE2.

Для перевода системы управления преобразователем частоты в режим установки параметров нажмите кнопку панели управления «*». Эта же кнопка используется для перехода между страницами (по ссылке) и для входа в режим установки конкретного параметра. Для переходов между параметрами, а также выбора их значений используются кнопки «Уменьшение» и «Увеличение».

Управление преобразователем частоты может осуществляться как внешним устройством, так и с помощью органов панели управления. При первом подключении к сети преобразователь по умолчанию переходит в режим внешнего управления. Именно этот режим и используется при выполнении лабораторной работы. Если по каким-либо причинам необходимо перевести преобразователь в режим локального управления, следует нажать и удерживать в течение двух секунд кнопку «Remote».

Организация внешних цепей контроллера Siemens S7-200

Внешний вид стенда «Siemens S7-200» показан на рис. 4.

На стенде расположены:

1 – короб с проводкой;

2 – CPU Siemens 226;

3 – модуль расширения EM222;

4 – блок гнезд и выключателей для организации входных цепей;

5 – блок гнезд для организации выходных цепей;

6 – автоматический выключатель;

7 – группы коротко соединенных гнездовых разъемов.

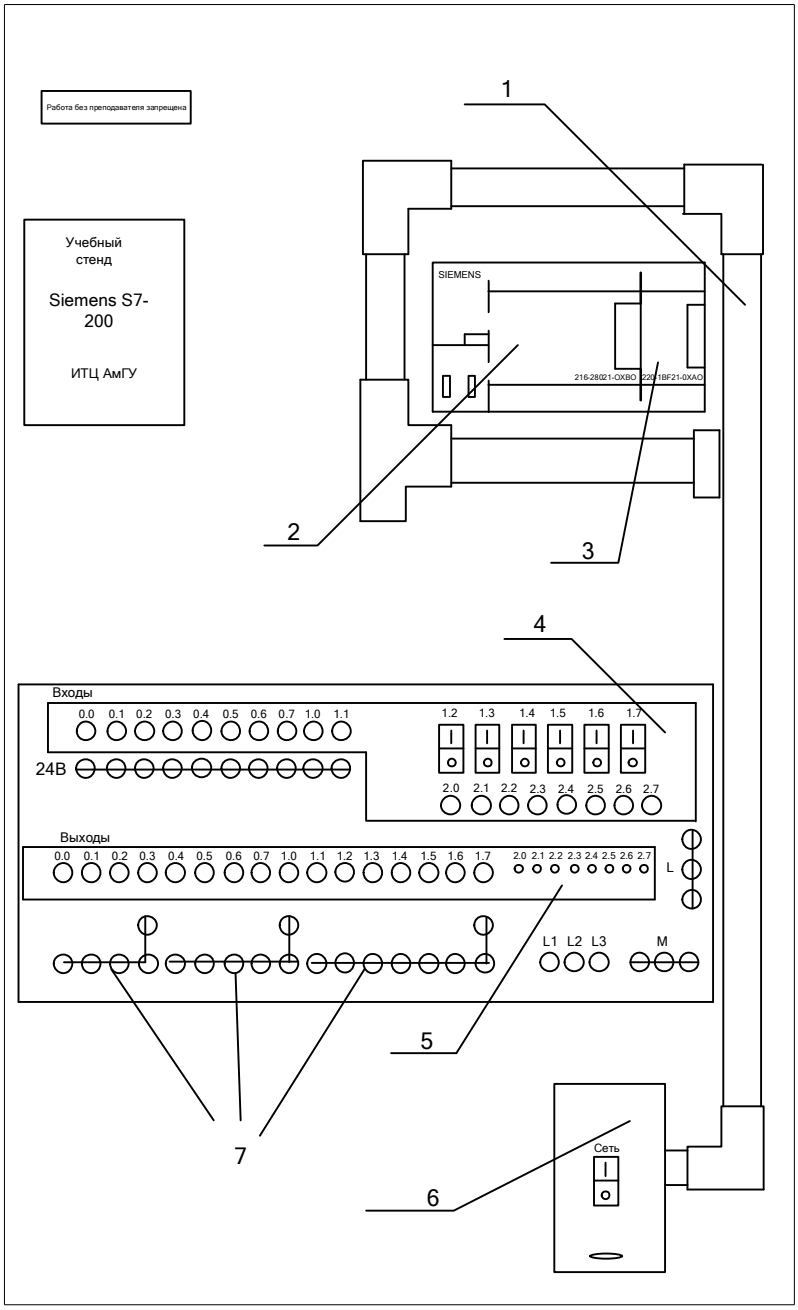


Рис. 4. Внешний вид стенда

На рис. 5 показан один из вариантов схемы внешних соединений CPU, приведенный в его техническом описании.

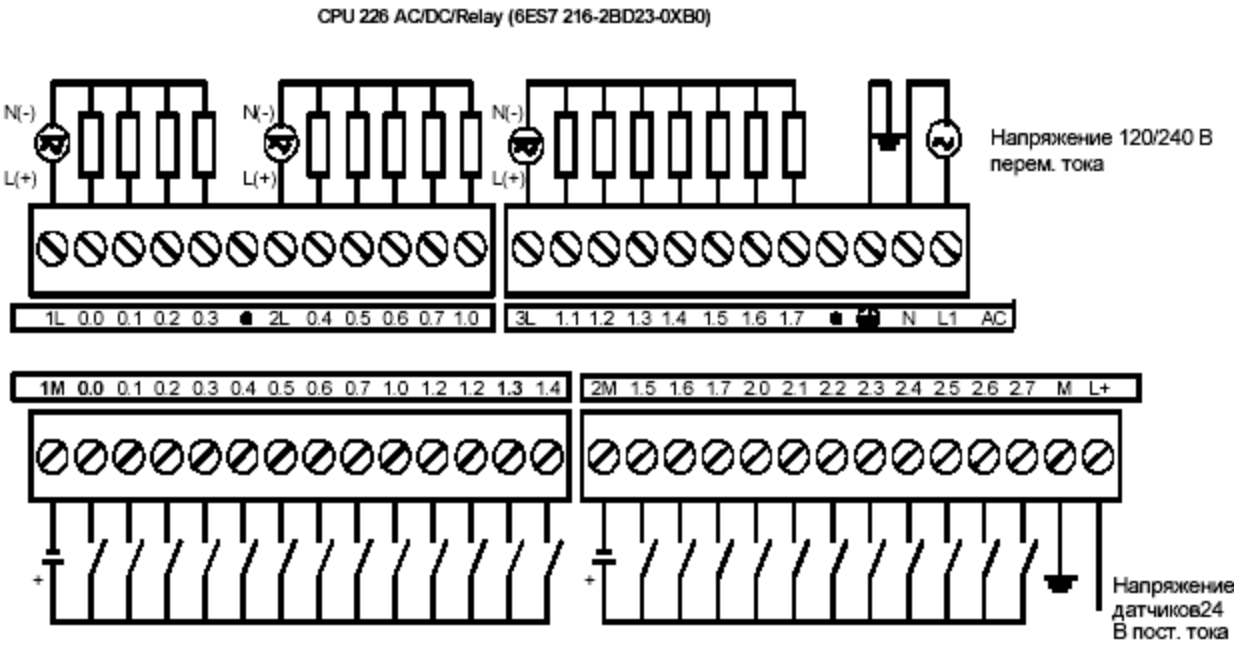


Рис. 5. Вариант схемы соединений CPU

К входам контроллера подключены датчики типа «сухой контакт». Для питания входных и выходных цепей используются независимые источники питания.

В стенде питание входных цепей организовано с использованием внутреннего источника питания (рис. 6), для этого:

клеммы 1М и 2М соединены с клеммой М («–» источника питания);

клемма L+ соединена с общим проводом для датчиков, который выведен на ряд гнездовых разъемов «24 В» и полюсами выключателей.

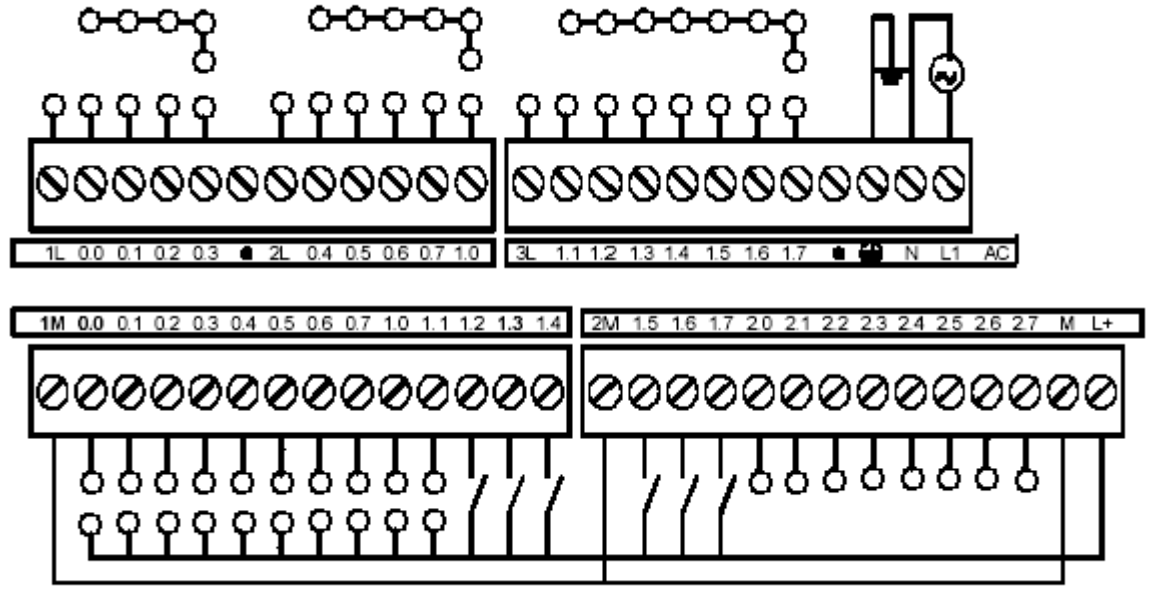


Рис. 6. Разводка внешних цепей CPU на стенде

Питание выходных цепей может осуществляться как от внутреннего источника, так и от внешнего.

Клеммы 1L, 2L, 3L соединены с соответствующими гнездами стенда (рис. 4), клеммы M и L+ источника питания соединены с группами гнезд (по три гнезда в группе). Это сделано для того, чтобы можно было организовать питание выходных цепей от внутреннего источника. Для этого следует соединить гнезда 1L, 2L, 3L с гнездами L+, а группы коротко соединенных гнездовых разъемов 7 (см. рис. 4) с гнездами M (или наоборот). Тогда становится возможным подключение нагрузок выходных цепей индивидуальными двухпроводными кабелями (без общего провода).

Внешние цепи модуля расширения EM222 в работе не рассматриваются.

Для подключения контроллера к преобразователю частоты следует:

соединить клемму 1L контроллера к клемме +24 В(5) преобразователя;

соединить клеммы Q0.0, Q0.1, Q0.2 контроллера к клеммам 6,7,8 преобразователя.

При этом функцию ключей, изображенных на рис. 2, будут выполнять контакты выходных реле контроллера, соединяющие при активации соответствующих выходов клеммы Q0.0, Q0.1, Q0.2 с клеммой 1L.

Программа работы

Постановка задачи

Предполагается, что электропривод приводит в движение подъемный механизм, который работает в цикле:

1) подъем на максимальной скорости;
2) при срабатывании путевого выключателя – переход на пониженную скорость, соответствующую частоте питания двигателя, равной 5 Гц;
3) при срабатывании конечного выключателя – остановка;
4) пауза 2 сек.;
5) спуск на максимальной скорости;
6) при срабатывании путевого выключателя – переход на пониженную скорость;
7) при срабатывании конечного выключателя – остановка;
8) пауза 1 сек.
9) переход к шагу 1.

Входными сигналами для контроллера будут являться сигналы следующих устройств:

выключатель запуска процесса (на панели стенда), связанный с входом I1.2;

два конечных и два путевых выключателя электропривода, имитируемых выключателями. Выключатели связаны с входами контроллера:

I1.3 – конечный выключатель в нижнем положении;

I1.4 – путевой выключатель, срабатывание которого при движении вниз должно приводить к переходу на пониженную скорость;

I1.5 – путевой выключатель, срабатывание которого при движении вверх должно приводить к переходу на пониженную скорость;

I1.6 – конечный выключатель в верхнем положении;

Выходами контроллера (входами преобразователя частоты) будут:
сигнал запуска привода, выход Q0.0 ;
сигнал реверса, выход Q0.1;
сигнал перехода на пониженную скорость, выход Q0.2.

Требуется:
изучить описание ПЧ и «запрограммировать» его на отработку описанных сигналов;
соединить входы ПЧ с выходами контроллера;
разработать и ввести в контроллер программу управления;
продемонстрировать работу системы.

Указания:
Программа управления будет состоять из основной программы и подпрограммы. Основной программе по нажатию кнопки «Пуск» будет вызываться подпрограмма.

Логика работы подпрограммы:

Контроль движения вверх и вниз:
Если не установлен бит Q0.1 (т.е. осуществляется движение вверх) и не включен конечный выключатель в верхнем положении I1.6,
ИЛИ,
если установлен бит Q0.1 (т.е. осуществляется движение вниз) и не включен конечный выключатель в нижнем положении I1.3, включаем выход Q0.0 (пускаем привод).

Контроль выдержек времени:
Если не установлен Q0.1 (т.е. ранее осуществлялось движение вверх) и включен конечный выключатель в верхнем положении I1.6, запускаем таймер №1.
Если сработал таймер №1, устанавливаем бит Q0.1 (подготавливая привод к движению вниз) и сбрасываем бит перехода на пониженную скорость Q0.2 (подготавливая привод к движению с максимальной скоростью);
При этом начнется движение вниз согласно логике контроля движения вверх и вниз.
Если установлен Q0.1 (т.е. ранее осуществлялось движение вниз) и включен конечный выключатель в нижнем положении I1.3, запускаем таймер №2.
Если сработал таймер №2, сбрасываем бит Q0.1 (подготавливая привод к движению вверх) и сбрасываем бит перехода на пониженную скорость Q0.2 (подготавливая привод к движению с максимальной скоростью);
При этом начнется движение вверх согласно логике контроля движения вверх и вниз.

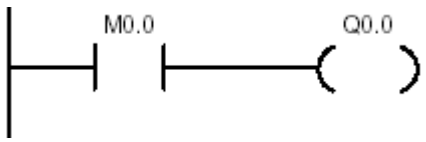
Контроль скорости:
Если не установлен Q0.1 (движение вверх) и сработал путевой выключатель I1.5,
ИЛИ,

если установлен Q0.1 (движение вниз) и сработал путевой выключатель I1.4, устанавливаем выход контроллера Q0.2 (тем самым, осуществляя переход на пониженную скорость).

Здесь использована следующая терминология (примеры поясняются гипотетическим кодом на языке С):

1. «Включить» – использовать «обыкновенную» «катушку» –( )– .

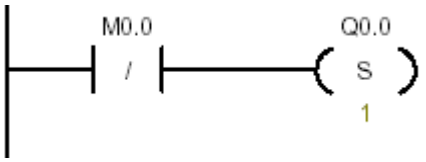
Пример: *если установлен бит M0.0, включить выход Q0.0 (если не установлен – выключить!)* :




```
if (M0.0)
    Q0.0 = 1;
else
    Q0.0 = 0;
```

2. «Установить» – использовать «катушку» с запоминанием состояния «1» –(S)–.

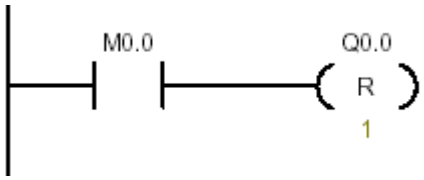
Пример: *если сброшен бит M0.0, установить выход Q0.0 (в противном случае – выход не изменять!)*:




```
if (!M0.0)
    Q0.0 = 1;
```

3. «Сбросить» – использовать «катушку» с запоминанием состояния «0» –(R)–.

Пример: *если установлен бит M0.0, сбросить выход Q0.0 (в противном случае – выход не изменять!)*:




```
if (M0.0)
    Q0.0 = 0;
```

Под катушками –(S)– и –(R)– указывается число бит, начиная с адреса, указанного над катушками, которые нужно установить в единицу или сбросить в нуль соответственно.

Содержание отчета

1. Принципиальная электрическая схема соединений цепей управления контроллера, преобразователя частоты, органов управления и выключателей, имитирующих путевые и концевые выключатели подъемника. На схеме должны быть приведены только используемые входы и выходы контроллера и преобразователя.

2. Программа контроллера в представлениях LAD, STL и FBD и временная диаграмма ее работы.

Контрольные вопросы

1. Опишите лабораторный стенд «Частотно-управляемый электропривод на основе ПЧ ABB ACS300», укажите назначение его элементов.
2. Опишите внешние цепи управления преобразователя частоты.
3. Опишите назначение дискретных входов преобразователя частоты в стандартном распределении функций.
4. Опишите лабораторный стенд «Siemens S7-200», укажите назначение его элементов.
5. Опишите разводку внешних цепей контроллера на стенде.
6. Опишите работу секций программы в представлении LAD.
7. Опишите работу секций программы в представлении STL.
8. Опишите работу секций программы в представлении и FBD.

4. Разработка и реализация программы измерения скорости электропривода

Цель работы: освоение техники работы с прерываниями и скоростными счетчиками Siemens S7-200.

Теоретические сведения

Описание установки

Установка состоит из частотно-управляемого электропривода на основе преобразователя частоты ABB ACS300 и контроллера Siemens S7-200.

Электропривод оснащен оптическим датчиком скорости/угла поворота, который установлен на клеммной коробке двигателя (рис. 1).

Рис. 1. Размещение датчика скорости/угла поворота

Электрическая принципиальная схема датчика показана на рис. 2.

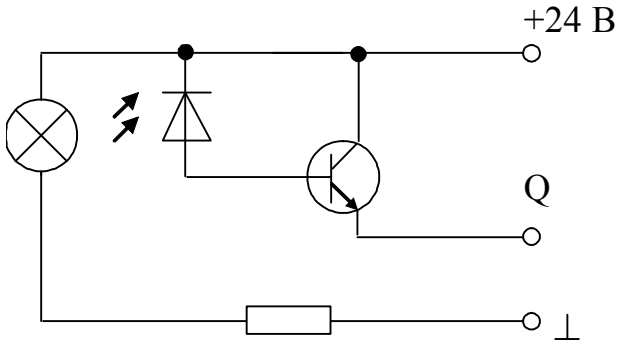


Рис. 2. Принципиальная схема датчика

Схема содержит источник света – электрическую лампу, ток в цепи которой ограничен специально подобранным сопротивлением, фотодиод и транзистор. При попадании потока света на фотодиод, последний увеличивает свою проводимость, вследствие чего открывается транзистор, и потенциал выхода Q приближается к +24 В. В противном случае потенциал Q близок к нулю.

Для сопряжения датчика с контроллером следует соединить лабораторными проводами

гнездо +24 В датчика – с гнездом +24 В контроллера,

гнездо «⊥» – с гнездом «GND» контроллера,

выход Q датчика – с каким-либо входом контроллера (рекомендуемый вход I0.0).

Световой поток, поступающий на фотодиод, перекрывается специальным диском, установленным на валу двигателя и вращающимся вместе с ним. Диск имеет 60 отверстий. Таким образом, на один оборот вала приходится 60 импульсов выходного напряжения. Поскольку двигатель имеет две пары полюсов, его синхронная частота вращения при частоте питающего напряжения 50 Гц равна 1500 об/мин или 25 об/сек (25 Гц). При такой частоте вращения частота следования импульсов составит 25×60 = 1500 Гц. Период следования импульсов равен 1/1500 ≈ 0,000667 сек или 0,667 мс. В таких условиях использовать «обыкновенный» программный счетчик, анализирующий входной сигнал в каждом программном цикле контроллера, невозможно: на больших частотах счетчик не в состоянии «уследить» за поступающими импульсами.

Традиционная архитектура ПЛК подразумевает, что максимальное количество элементов системы работает относительно независимо от центрального процессора. В частности, в контроллере Siemens S7-200 имеются так называемые *скоростные счетчики*, осуществляющие счет импульсов, поступающих на входы контроллера, независимо от CPU. Скоростные счетчики реализуют идею аппаратного решения проблемы счета и взаимодействуют с CPU асинхронным способом.

Скоростные счетчики

На рис. 3 приведены команды управления скоростными счетчиками.

Определение режима работы скоростного счетчика HDEF

Команда определения режима работы скоростного счетчика (HDEF) устанавливает режим работы для определенного скоростного счетчика (HSCx). Выбором режима определяются датчик тактовых импульсов, направление и функции запуска и сброса скоростного счетчика.

Запуск скоростного счетчика HSC

Команда активизации скоростного счетчика (HSC) конфигурирует и управляет режимом работы скоростного счетчика через сигнальные состояния битов специальной памяти HSC. Параметр N определяет номер скоростного счетчика. Допустимые операнды для команд HDEF и HSC приведены в табл. 1.

Скоростные счетчики могут быть сконфигурированы на двенадцать различных режимов работы (табл. 2).

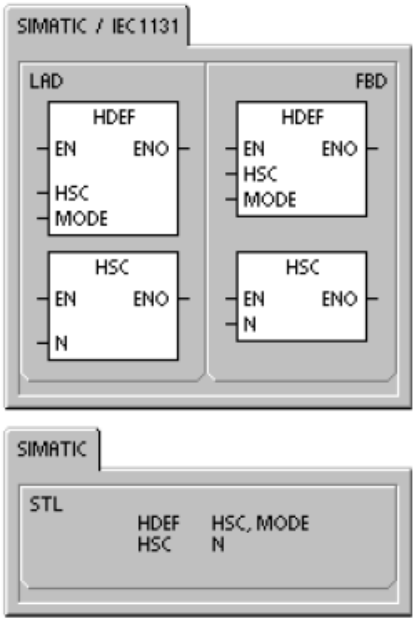


Рис. 3. Команды управления скоростными счетчиками

Каждый счетчик имеет специализированные входы, которые поддерживают такие функции, как датчик тактовых импульсов, управление направлением, сброс и запуск. Для двухфазных счетчиков оба датчика тактовых импульсов могут работать со своей максимальной скоростью. В квадратурных режимах (A/B-счетчики) предоставляется возможность выбора однократной (1x) или четырехкратной (4x) скорости счета. Все счетчики работают с максимальной скоростью, не создавая помех друг другу.

Таблица 1. Допустимые операнды для скоростных счетчиков

Входы/выходы	Типы данных	Операнды
HSC, MODE	BYTE	константа
N	WORD	константа

CPU 226 поддерживают шесть скоростных счетчиков: от HSC0 до HSC5.

В большинстве случаев применения скоростной счетчик загружается первым из нескольких предустановленных значений, и желаемые выходы активизируются на интервал времени, в течение которого текущее значение счетчика меньше текущего предустановленного значения. Счетчик настроен таким образом, что, когда текущее значение счетчика становится равным предустановленному значению, или при появлении сброса происходит прерывание.

Когда при равенстве текущего значения счетчика и предустановленного значения происходит прерывающее событие, загружается новое предустановленное значение, и устанавливается следующее состояние для выходов. Когда происходит событие, вызывающее прерывание по сбросу, то устанавливаются первое предустановленное значение и первые состояния выходов, и цикл повторяется.

Так как прерывания происходят со значительно меньшей частотой, чем та, с которой считает скоростной счетчик, то может быть реализовано точное управление быстрыми операциями при относительно малом воздействии на общий цикл обработки программы ПЛК. Метод подключения прерываний по-

зволяет выполнять каждую загрузку нового предустановленного значения в отдельной программе обработки прерывания, что упрощает управление состоянием. (В качестве альтернативы, все события, вызывающие прерывания, могут быть обработаны и в одной единственной программе обработки прерываний.).

Описание различных скоростных счетчиков

Все счетчики в одном и том же режиме работают одинаково. Имеется четыре основных вида счетчиков: однофазный счетчик с внутренним управлением направлением, однофазный счетчик с внешним управлением направлением, двухфазный счетчик с 2 тактовыми входами и квадратурный счетчик с фазами A и B. Не каждый счетчик поддерживает все режимы. Каждый счетчик можно использовать: без входов сброса и пуска, со сбросом, но без пуска, или с входами пуска и сброса.

Когда активизируется вход сброса, он сбрасывает текущее значение и сохраняет его сброшенным, пока не деактивизируется сброс.

Когда активизируется вход пуска, он разрешает счетчику считать. Если вход пуска деактивизирован, текущее значение счетчика остается постоянным, а тактовые события игнорируются.

Если сброс активизируется, когда пуск неактивен, то сброс игнорируется, а текущее значение не изменяется. Если вход пуска становится активным, когда активен вход сброса, текущее значение сбрасывается.

Перед использованием скоростного счетчика вы должны с помощью команды HDEF (**H**igh Speed Counter **Def**inition – определение скоростного счетчика) выбрать его режим. С помощью бита памяти первого цикла SM0.1 (этот бит включен в течение первого цикла обработки программы, а затем выключается) вызывается фрагмент кода, который содержит команду HDEF.

Программирование скоростного счетчика

Для программирования скоростного счетчика требуется выполнить следующие основные задачи:

1) определить счетчик и режим;
2) настроить управляющий байт;
3) установить текущее (начальное) значение;
4) задать предустановленное (целевое) значение;
5) назначить и разблокировать программу обработки прерываний;
6) активизировать скоростной счетчик.

Определение режимов и входов счетчика

Для определения режимов и входов счетчика используется команда определения скоростного счетчика HDEF.

В таблице 3 показаны входы для таких функций скоростных счетчиков, как генератор тактовых импульсов, управление направлением, сброс и запуск. Один и тот же вход не может быть использован для двух разных функций, но любой вход, не используемый текущим режимом скоростного счетчика, может быть использован для другой цели.

Например, если HSC0 работает в режиме 1, который использует I0.0 и I0.2, то I0.1 может быть использован, например, для прерываний по фронту сигнала или для обслуживания HSC3.

Таблица 2. Входы функций скоростных счетчиков

Режим	Описание	Входы			
	HSC0	I0.0	I0.1	I0.2	
	HSC1	I0.6	I0.7	I1.0	I1.1
	HSC2	I1.2	I1.3	I1.4	I1.5
	HSC3	I0.1			
	HSC4	I0.3	I0.4	I0.5	
	HSC5	I0.4			
0	Однофазный счетчик с внутренним управлением направлением	Датчик тактовых импульсов			
1		Датчик тактовых импульсов		Сброс	
2		Датчик тактовых импульсов		Сброс	Пуск
3	Однофазный счетчик с внешним управлением направлением	Датчик тактовых импульсов	Направление		
4		Датчик тактовых импульсов	Направление	Сброс	
5		Датчик тактовых импульсов	Направление	Сброс	Пуск
6	Двухфазный счетчик с 2 тактовыми входами	Датчик тактовых импульсов для прямого направления	Датчик тактовых импульсов для обратного направления		
7		Датчик тактовых импульсов для прямого направления	Датчик тактовых импульсов для обратного направления	Сброс	
8		Датчик тактовых импульсов для прямого направления	Датчик тактовых импульсов для обратного направления	Сброс	Пуск
9	Квадратурный счетчик с фазами A и B	Датчик тактовых импульсов A	Датчик тактовых импульсов B		
10		Датчик тактовых импульсов A	Датчик тактовых импульсов B	Сброс	
11		Датчик тактовых импульсов A	Датчик тактовых импульсов B	Сброс	Пуск
12	Режим счета 12 поддерживают только HSC0 и HSC3. HSC0 считает количество импульсов, выдаваемых Q0.0. HSC3 считает количество импульсов, выдаваемых Q0.1.				

Примеры режимов HSC

Временные диаграммы на рисунках 4 – 8 показывают, как работает каждый счетчик в соответствии с режимом.

Рис. 4. Пример работы в режимах 0, 1 или 2

Рис. 5. Пример работы в режимах 3, 4 или 5

Когда используются режимы счета 6, 7 или 8, и в течение 0,3 микросекунды друг за другом появляется нарастающий фронт на тактовых входах счета вперед и счета назад, скоростной счетчик может рассматривать эти события как происходящие одновременно.

Если это происходит, то текущее значение не меняется и не отображается изменение в направлении счета. Если между поступлениями нарастающих фронтов на тактовые входы счета вперед и счета назад проходит больше 0,3 микросекунды, то скоростной счетчик воспринимает эти события отдельно. В этом случае ошибки не происходит, и счетчик сохраняет правильное счетное значение.

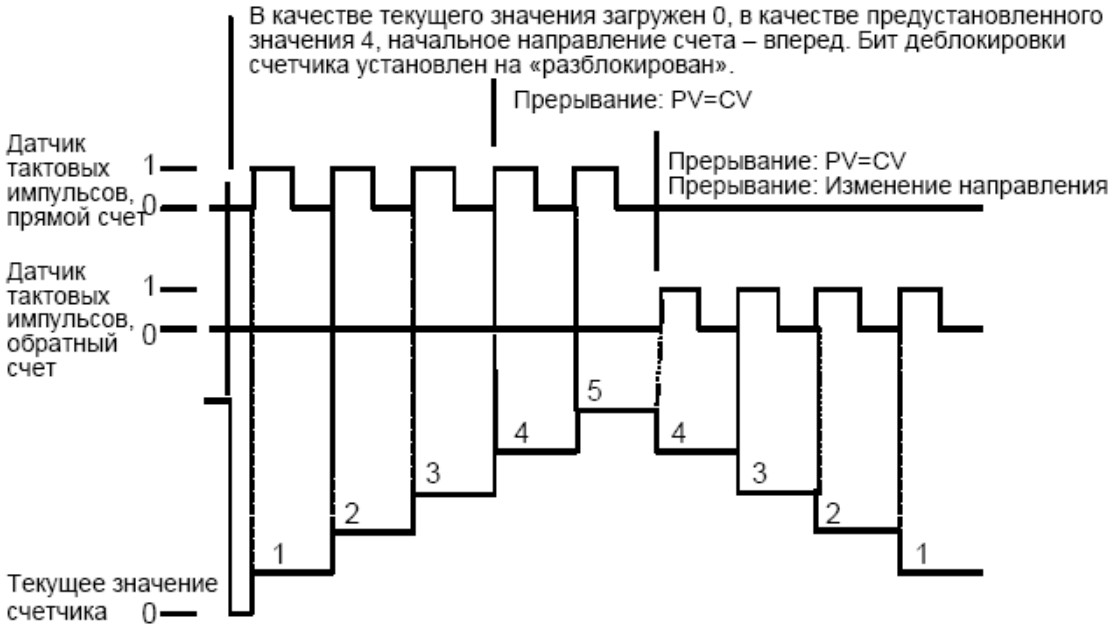


Рис. 6. Пример работы в режимах 6, 7 или 8

Рис. 7. Пример работы в режимах 9, 10 или 11 (квадратурный режим, однократная скорость)

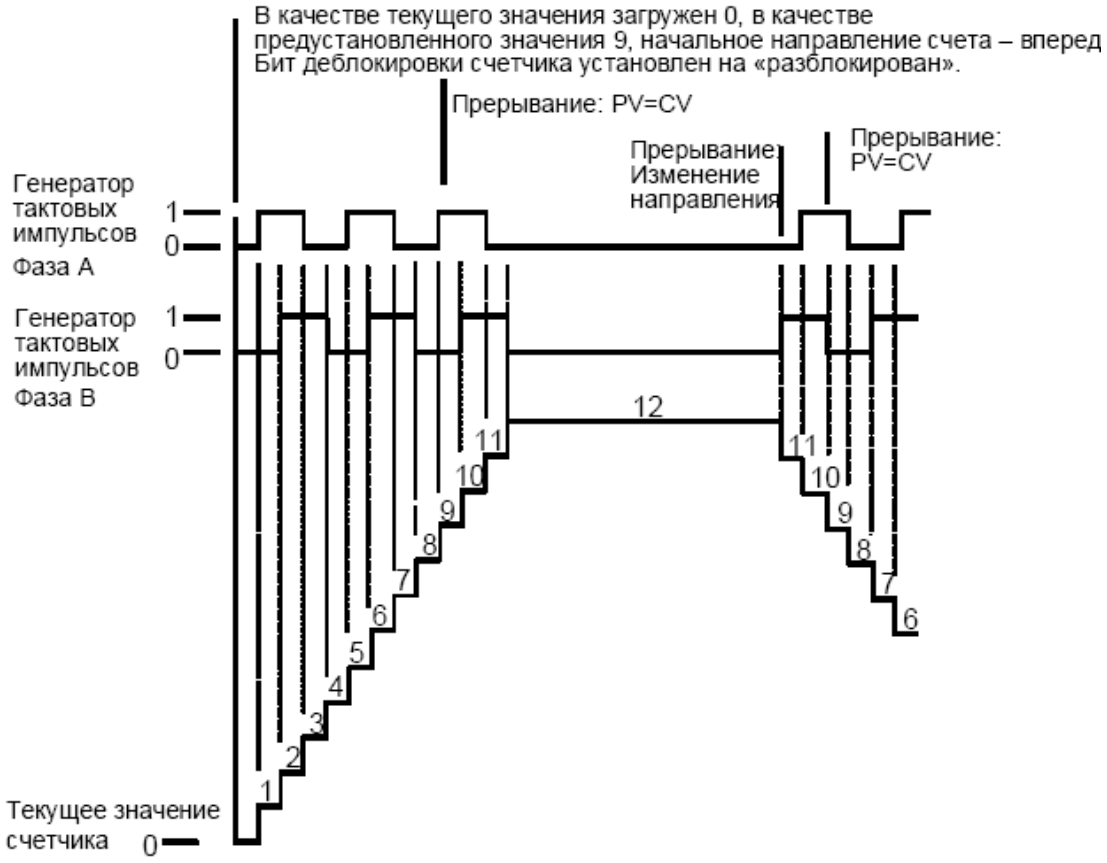


Рис. 8. Пример работы в режимах 9, 10 или 11 (квадратурный режим, четырехкратная скорость)

Четыре счетчика имеют три управляющих бита, которые используются для конфигурирования активного состояния входов сброса и пуска и для выбора односкоростного или четырехскоростного режима счета (только для квадратурных счетчиков). Эти биты находятся в управляющем байте соответствующего счетчика и используются только тогда, когда выполняется команда HDEF. Эти биты определены в табл. 3.

Таблица 3. Активный уровень для управляющих битов сброса, пуска и выбора однократной или четырехкратной скорости

HSC0	HSC1	HSC2	HSC4	Описание (используются только при исполнении HDEF)
SM37.0	SM47.0	SM57.0	SM147.0	Активный уровень управляющего бита для сброса [1]: 0 = сброс активен при высоком уровне; 1 = сброс активен при низком уровне
---	SM47.1	SM57.1	---	Активный уровень управляющего бита для пуска [1]: 0 = пуск активен при высоком уровне; 1 = пуск активен при низком уровне
SM37.2	SM47.2	SM57.2	SM147.2	Выбор скорости счета для квадратурных счетчиков: 0 = 4-кратная скорость 1 = 1-кратная скорость
1 По умолчанию входы сброса и пуска активны при высоком уровне сигнала, а в квадратурных счетчиках скорость счета установлена четырехкратной (по отношению к частоте входного датчика тактовых импульсов).				

Необходимо установить эти управляющие биты в соответствии с желаемым состоянием до исполнения команды HDEF. В противном случае счетчик принимает конфигурацию, определенную по умолчанию для выбранного режима работы счетчика.

Если команда HDEF была выполнена, нельзя изменить настройку счетчика, не переведя сначала S7–200 в состояние STOP.

Настройка управляющего байта

Определив счетчик и режим его работы, можно программировать динамические параметры счетчика. Каждый скоростной счетчик имеет управляющий байт, который позволяет выполнить следующие действия:

разблокировать или заблокировать счетчик;

управлять направлением (только для режимов 0, 1 и 2) или устанавливать начальное направление счета для всех остальных режимов;

загружать текущее значение;

загружать предустановленное значение.

Проверка управляющего байта и соответствующих текущего и предустановленного значений производится при выполнении команды HSC. В таблице 4 описан каждый из этих управляющих битов.

Таблица 4. Управляющие биты для скоростных счетчиков

HSC0	HSC1	HSC2	HSC3	HSC4	HSC5	Описание
SM37.3	SM47.3	SM57.3	SM137.3	SM147.3	SM157.3	Бит управления направлением счета: 0 = обратный счет 1 = прямой счет
SM37.4	SM47.4	SM57.4	SM137.4	SM147.4	SM157.4	Записать направление счета в HSC: 0 = не актуализировать 1 = актуализировать направление
SM37.5	SM47.5	SM57.5	SM137.5	SM147.5	SM157.5	Записать новое предустановленное значение в HSC: 0 = не актуализировать; 1 = актуализировать предустановленное значение
SM37.6	SM47.6	SM57.6	SM137.6	SM147.6	SM157.6	Записать новое текущее значение в HSC: 0 = не актуализировать; 1 = актуализировать текущее значение
SM37.7	SM47.7	SM57.7	SM137.7	SM147.7	SM157.7	Разблокировка HSC: 0 = заблокировать HSC; 1 = разблокировать HSC

Установка текущего и предустановленного значений

Каждый скоростной счетчик имеет 32-битное текущее значение и 32- битное предустановленное значение. Оба значения являются целыми числами со знаком. Чтобы загрузить новое текущее или предустановленное значение, необходимо настроить управляющий байт и байты специальной памяти, содержащие текущее и/или предустановленное значение, а также выполнить команду HSC, чтобы новые значения были переданы в скоростной счетчик.

Таблица 5 описывает байты специальной памяти, используемые для хранения новых текущих и предустановленных значений.

В дополнение к управляющим байтам и байтам, содержащим новые текущие и предустановленные значения, текущее значение каждого скоростного счетчика может быть прочитано путем задания типа данных HC (текущее значение скоростного счетчика), за которым следует номер (0, 1, 2, 3, 4 или 5) счетчика, как показано в таблице 5. Текущее значение непосредственно дос-

тупно для операций чтения, но оно может быть записано только с помощью команды HSC.

Таблица 5. Новое текущее и новое предустановленное значение

Загружаемое значение	HSC0	HSC1	HSC2	HSC3	HSC4	HSC5
Новое текущее значение	SMD38	SMD48	SMD58	SMD138	SMD148	SMD158
Новое предустановленное значение	SMD42	SMD52	SMD62	SMD142	SMD152	SMD162

Таблица 6. Текущие значения скоростных счетчиков

Значение	HSC0	HSC1	HSC2	HSC3	HSC4	HSC5
Текущее значение	HC0	HC1	HC2	HC3	HC4	HC5

Адресация скоростных счетчиков (HC)

Для доступа к счетному значению скоростного счетчика указывается адрес этого счетчика с помощью типа памяти (HC) и номера счетчика (например, HC0). Текущее значение скоростного счетчика доступно только для чтения и может быть адресовано только как двойное слово (32 бита), как показано на рис. 9.

Рис. 9. Доступ к текущему значению скоростного счетчика

Назначение прерываний

Все режимы счетчиков поддерживают прерывание по равенству текущего значения HSC загруженному предустановленному значению. Режимы счетчиков, использующие вход внешнего сброса, поддерживают прерывание по активизации внешнего сброса. Все режимы счетчиков, кроме режимов 0, 1 и 2, поддерживают прерывание по изменению направления счета. Каждое из этих условий возникновения прерываний может быть заблокировано или разблокировано по отдельности. Полностью использование прерываний обсуждается в разделе о командах обмена данными и прерывания.

Байт состояния

Каждому скоростному счетчику поставлен в соответствие байт состояния, предоставляющий в распоряжение биты памяти, указывающие текущее направление счета, а также информацию о том, действительно ли текущее значение больше или равно предустановленному. Таблица 7 определяет эти биты состояния для каждого скоростного счетчика.

Биты состояния действительны только во время исполнения программы обработки прерывания скоростного счетчика. Цель контроля состояния скоростного счетчика состоит в том, чтобы разблокировать прерывания для событий, оказывающих воздействие на выполняемую операцию.

Таблица 7. Биты состояния для скоростных счетчиков

HSC0	HSC1	HSC2	HSC3	HSC4	HSC5	Описание
SM36.0	SM46.0	SM56.0	SM136.0	SM146.0	SM156.0	Не используются
SM36.1	SM46.1	SM56.1	SM136.1	SM146.1	SM156.1	Не используются
SM36.2	SM46.2	SM56.2	SM136.2	SM146.2	SM156.2	Не используются
SM36.3	SM46.3	SM56.3	SM136.3	SM146.3	SM156.3	Не используются
SM36.4	SM46.4	SM56.4	SM136.4	SM146.4	SM156.4	Не используются
SM36.5	SM46.5	SM56.5	SM136.5	SM146.5	SM156.5	Бит состояния текущего направления счета: 0 = обратный счет 1 = прямой счет
SM36.6	SM46.6	SM56.6	SM136.6	SM146.6	SM156.6	Бит состояния, указывающий, равно ли текущее значение предустановленному: 0 = не равно 1 = равно
SM36.7	SM46.7	SM56.7	SM136.7	SM146.7	SM156.7	Бит состояния, указывающий, больше ли текущее значение, чем предустановленное: 0 = меньше или равно 1 = больше

Пример инициализирующих последовательностей для скоростных счетчиков

В следующем описании инициализации и последовательности обработки в качестве примера используется HSC1. При описании инициализаций предполагается, что S7–200 только что переведен в режим RUN, и поэтому бит памяти первого цикла установлен. Если это не так, команда HDEF может быть выполнена только один раз для каждого скоростного счетчика после вхождения в режим RUN. Выполнение HDEF для скоростного счетчика во второй раз приводит к ошибке выполнения и не изменяет настройку счетчика по сравнению с тем, как она была выполнена для данного счетчика при первом выполнении HDEF.

Хотя приведенная далее последовательность показывает, как изменить направление, текущее и предустановленное значение по отдельности, можно изменить все эти настройки или любую их комбинацию в той же последовательности, устанавливая надлежащим образом SMB47, а затем выполняя команду HSC.

Следующие шаги описывают, как инициализировать HSC1 в качестве однофазного реверсивного счетчика с внутренним управлением направлением счета (режим 0, 1 или 2).

1. Используйте бит памяти первого цикла для вызова подпрограммы, в которой будет выполняться операция по инициализации. Когда вы используете вызов подпрограммы, следующие циклы эту подпрограмму не вызывают, что сокращает время цикла и делает программу более структурированной.

2. В подпрограмме инициализации загрузите SMB47 в соответствии с желаемой операцией управления. Например:

SMB47 = 16#F8 *дает следующие результаты:*

1) разблокирует счетчик;

2) записывает новое текущее значение;

3) записывает новое предустановленное значение;

4) устанавливает прямое направление счета;

5) настраивает входы пуска и сброса на активность при высоком уровне сигнала.

3. Выполните команду HDEF с входом HSC , установленным в 1, и входом MODE [режим], установленным в 0 при отсутствии внешнего сброса и пуска, 1 для внешнего сброса без пуска или 2 для внешнего сброса и пуска.

4. Загрузите SMD48 (двойное слово) желаемым текущим значением (загрузите 0, чтобы его очистить).

5. Загрузите SMD52 (двойное слово) желаемым предустановленным значением.

6. Чтобы распознавать равенство текущего и предустановленного значений, запрограммируйте прерывание, поставив в соответствие программе обработки прерывания прерывающее событие CV = PV (событие 13). Подробную информацию об обработке прерываний вы найдете в разделе, посвященном командам прерывания.

7. Чтобы распознавать внешний сброс, запрограммируйте прерывание, поставив в соответствие программе обработки прерывания прерывающее событие «внешний сброс» (external reset) (событие 15).

8. Для разблокировки прерываний выполните команду разрешения всех прерываний (ENI).

9. Выполните команду HSC, чтобы S7–200 запрограммировал HSC1.

10. Выйдите из подпрограммы.

Команды прерывания Siemens S7-200

Команды управления прерываниями контроллера приведены на рис.10

Разблокирование и блокирование прерываний

Команда разблокирования прерываний (ENI) разблокирует обработку всех назначенных прерывающих событий. Команда блокирования прерываний (DISI) блокирует обработку всех прерывающих событий.

Когда производится перевод в режим RUN, прерывания первоначально заблокированы. Находясь в режиме RUN, вы можете разблокировать все прерывания, выполнив команду разблокирования прерываний.

Выполнение команды блокирования прерываний запрещает обработку прерываний, однако активные прерывающие события и далее будут ставиться в очередь.

Условный возврат из программы обработки прерываний

Команда условного возврата из программы обработки прерываний (CRETI) может быть использована для возврата из программы обработки прерываний в зависимости от условия, задаваемого предшествующей логикой.

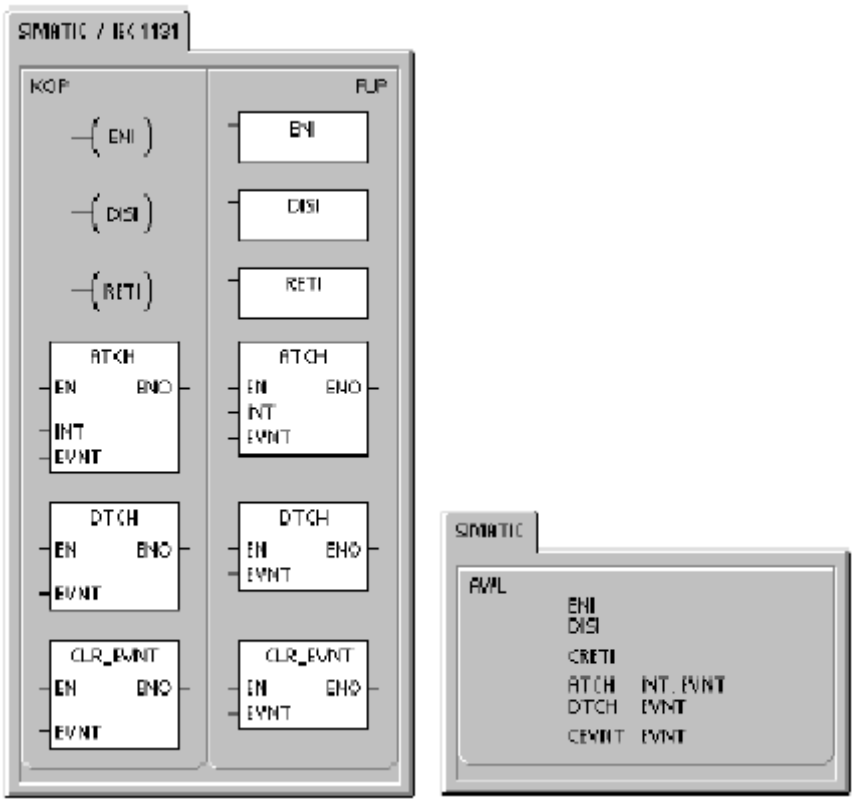


Рис. 10. Команды прерываний Siemens S7–200

Назначение прерывания

Команда назначения прерывания (ATCH) связывает прерывающее событие EVNT с номером программы обработки прерываний INT и разблокирует прерывающее событие.

Отсоединение прерывания

Команда отсоединения прерывания (DTCH) разрывает связь прерывающего события EVNT со всеми программами обработки прерываний и блокирует прерывающее событие.

Очистка прерывающих событий

Команда очистки прерывающих событий удаляет все прерывающие события типа EVNT из очереди прерываний. Эта команда используется для очистки очереди прерываний от нежелательных прерывающих событий. Если эта команда используется для удаления ложных прерывающих событий, вы должны отсоединить это событие перед удалением событий из очереди. Иначе после выполнения команды очистки прерывающих событий к очереди будут добавлены новые события.

Таблица 8. Допустимые операнды для команд прерывания

Входы/выходы	Типы данных	Операнды
INT	BYTE	константа (от 0 до 127)
EVNT	BYTE	константа (от 0 до 33)

Принцип действия команд назначения и отсоединения прерываний

Прежде чем программа обработки прерывания может быть вызвана, должно быть установлено соответствие между прерывающим событием и сегментом программы, который нужно выполнить, когда это событие происходит. Для организации связи между прерывающим событием (задаваемым номером прерывающего события) и сегментом программы (задаваемым номером программы обработки прерывания) используйте команду назначения прерывания (ATCH). 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 вы назначаете прерывающее событие программе обработки прерываний, это прерывание автоматически разблокируется. Если вы заблокировали все прерывания с помощью команды блокирования прерываний, то каждое возникновение прерывающего события ставится в очередь, пока прерывания не будут снова разблокированы с помощью команды разблокирования прерываний или не произойдет переполнение очереди прерываний.

Отдельные прерывающие события можно заблокировать разрывом связи между этим прерывающим событием и программой обработки прерывания с помощью команды отсоединения прерывания. Команда отсоединения возвращает прерывание в неактивное или игнорируемое состояние. Таблица 9 перечисляет различные типы прерывающих событий.

Исполнение программ обработки прерываний в S7–200

Программа обработки прерывания исполняется в ответ на соответствующее внутреннее или внешнее событие. После выполнения последней команды программы обработки прерывания управление возвращается в главную программу. Обработка прерываний обеспечивает быструю реакцию на определенные внутренние или внешние события.

Если программа обработки прерывания спроектирована короткой с точными спецификациями, то она будет быстро выполняться и не будет задерживать другие процессы на длительные промежутки времени. Если этого не сделать, то неожиданные условия могут вызвать ненормальную работу оборудования, управляемого главной программой. Для прерываний безусловно верна аксиома «чем короче, тем лучше».

В программе обработке прерывания нельзя использовать команды блокирования прерываний (DISI), разблокирования прерываний (ENI), определения режима работы скоростного счетчика (HDEF) и завершения обработки (END).

Системная поддержка прерываний

Так как прерывания могут оказывать влияние на контакты, катушки и аккумуляторы, то система сохраняет и перезагружает логический стек, аккумуляторные регистры и биты специальной памяти (SM), которые отображают состояние аккумуляторов и команд. Это позволяет избежать искажения главной программы пользователя из-за перехода в программу обработки прерывания и возвращения из нее.

Таблица 9. Прерывающие события.

Событие	Описание	CPU 221 CPU 222	CPU 224	CPU 224XP CPU 226
0	I0.0 Нарастающий фронт	да	да	да
1	I0.0 Падающий фронт	да	да	да
2	I0.1 Нарастающий фронт	да	да	да
3	I0.1 Падающий фронт	да	да	да
4	I0.2 Нарастающий фронт	да	да	да
5	I0.2 Падающий фронт	да	да	да
6	I0.3 Нарастающий фронт	да	да	да
7	I0.3 Падающий фронт	да	да	да
8	Порт 0 Символ принят	да	да	да
9	Порт 0 Передача завершена	да	да	да
10	Управляемое временем прерывание 0 SMB34	да	да	да
11	Управляемое временем прерывание 1 SMB35	да	да	да
12	HSC0 CV=PV (текущее значение = предустановленному)	да	да	да
13	HSC1 CV=PV (текущее значение = предустановленному)		да	да
14	HSC1 Изменение направления		да	да
15	HSC1 Внешний сброс		да	да
16	HSC2 CV=PV (текущее значение = предустановленному)		да	да
17	HSC2 Изменение направления		да	да
18	HSC2 Внешний сброс		да	да
19	PLS0 Прерывание по завершению отсчета количества импульсов PTO	да	да	да
20	PLS1 Прерывание по завершению отсчета количества импульсов PTO	да	да	да
21	Таймер T32 Прерывание CT=PT	да	да	да
22	Таймер T96 Прерывание CT=PT	да	да	да
23	Порт 0 Прием сообщения завершен	да	да	да
24	Порт 1 Прием сообщения завершен			да
25	Порт 1 Символ принят			да
26	Порт 1 Передача завершена			да
27	HSC0 Изменение направления	да	да	да
28	HSC0 Внешний сброс	да	да	да
29	HSC4 CV=PV (текущее значение = предустановленному)	да	да	да
30	HSC4 Изменение направления	да	да	да
31	HSC4 Внешний сброс	да	да	да
32	HSC3 CV=PV (текущее значение = предустановленному)	да	да	да
33	HSC5 CV=PV (текущее значение = предустановленному)	да	да	да

Совместное использование данных главной программой и программами обработки прерываний

Данные могут совместно использоваться главной программой и одной или несколькими программами обработки прерываний. Так как невозможно предсказать, когда S7–200 может сгенерировать прерывание, то желательно ограничить количество переменных, которые применяются как в программе обработки прерываний, так и в других местах программы. В результате действий программы обработки прерываний могут возникнуть проблемы непротиворечиво-

сти совместно используемых данных, когда выполнение команд вашей главной программы прерывается событиями, вызывающими прерывания. Чтобы гарантировать, что ваша программа обработки прерываний будет использовать только временную память и не перезапишет данные, используемые еще в какомлибо месте вашей программы, пользуйтесь таблицей локальных переменных программы обработки прерываний.

Существует ряд методов программирования, которые вы можете использовать, чтобы обеспечить корректное разделение данных между вашей главной программой и программами обработки прерываний. Эти методы или ограничивают способ доступа к совместно используемым ячейкам памяти, или препятствуют прерыванию последовательностей команд, использующих разделяемые ячейки памяти.

Для программы на STL, совместно использующей только одну переменную: если совместно используемые данные представляют собой одну переменную в виде байта, слова или двойного слова, и ваша программа написана на STL, то корректный доступ к совместно используемым данным может быть обеспечен сохранением промежуточных результатов операций над совместно используемыми данными только в тех адресах памяти или аккумуляторах, которые совместно не используются.

Для программы на LAD, которая совместно использует единственную переменную: если разделяемые данные представляют собой единственную переменную в виде байта, слова или двойного слова и ваша программа написана на LAD, то корректный совместный доступ может быть обеспечен установлением соглашения, что доступ к разделяемым ячейкам памяти может осуществляться только с помощью команд пересылки (MOVB, MOVW, MOVD, MOVR). В то время как многие команды LAD составлены из непрерываемых последовательностей команд STL, команды пересылки состоят из единственной команды STL, на исполнение которой не могут влиять прерывающие события.

Для программы на STL или LAD, совместно использующей несколько переменных: если разделяемые данные составлены из ряда связанных байтов, слов или двойных слов, то для управления исполнением программ обработки прерываний могут быть использованы команды блокировки/ разблокировки прерываний (DISI и ENI). В той точке вашей программы, где должны начаться операции с разделяемыми ячейками памяти, заблокируйте прерывания. Как только все действия, влияющие на совместно используемые ячейки памяти, завершены, вновь разблокируйте прерывания. В течение времени, когда прерывания заблокированы, программы обработки прерываний не могут выполняться и, следовательно, не имеют доступа к разделяемым ячейкам памяти; однако такой подход может привести к запаздыванию реакции на прерывающие события.

Вызов подпрограмм из программ обработки прерываний

Из программы обработки прерывания можно вызвать только один уровень вложенности подпрограмм. Аккумуляторы и логический стек совместно используются программой обработки прерывания и вызываемой подпрограммой.

Виды прерываний, поддерживаемых S7–200

S7–200 поддерживает следующие виды программ обработки прерываний:

прерывания коммуникационных портов: S7–200 генерирует события, которые позволяют вашей программе управлять коммуникационным портом;

прерывания по вводу/выводу: S7–200 генерирует события для различных изменений состояния различных входов-выходов. Эти события позволяют вашей программе реагировать на скоростные счетчики, вывод импульсов и на нарастающие или падающие фронты на входах;

прерывания, управляемые временем: S7–200 генерирует события, которые позволяют вашей программе реагировать через определенные интервалы времени.

Приоритет прерываний и постановки их в очередь

Прерывания обслуживаются S7–200 в порядке их возникновения с учетом соответствующей группы приоритета. В любой момент времени исполняется только одна программа обработки прерывания. Когда исполнение программы обработки прерывания начинается, программа исполняется до своего завершения. Она не может быть прервана другой программой обработки прерывания, даже если последняя имеет более высокий приоритет. Прерывания, возникающие во время обработки другого прерывания, ставятся в очередь для последующей обработки.

Подробнее о прерываниях смотрите в Руководстве пользователя.

Программа работы. Разработка программ измерения частоты вращения

Способы измерения

Принципиально существует два способа вычисления частоты вращения, основанных на подсчете поступающих импульсов:

1) определение числа импульсов за заданное время;

2) определение временного интервала, в течение которого поступает заданное количество импульсов.

В данной работе реализуем оба способа.

Определение числа импульсов за заданное время

Программа будет подсчитывать количество импульсов, поступающих с датчика в течение одной секунды. Отчет времени будет вести таймер, генерирующий прерывания каждую секунду.

Включите контроллер и запустите Step 7 MicroWin. Создайте новый проект. Удалите из проекта подпрограмму SBR_0.

В основной программе определите, сконфигурируйте и запустите скоростной счетчик HSC0, осуществите привязку программы обработки прерывания INT_0 к прерыванию от таймера и запустите таймер T32.

В первом цикле контроллера (если установлен бит SM0.1):

запишите в управляющий байт счетчика HSC0 число, обеспечивающее:

1) прямой счет;

2) актуализацию направления счета;

3) актуализацию предустановленного значения;

4) актуализацию текущего значения;

5) разблокирование HSC0.

остальные опции настройки несущественны. Запись байта осуществляется с помощью команды MOVB раздела инструкций «Переместить»;

с помощью команды HDEF определите скоростной счетчик HSC0 в режиме 0;

запишите по соответствующему адресу области специальной памяти новое текущее значение счетчика HSC0, равное 0. Запись двойного слова осуществляется с помощью команды MOVDW;

запишите по соответствующему адресу области специальной памяти новое предустановленное значение счетчика HSC0, равное достаточно большому числу, например, 1000000;

разрешите прерывания;

активизируйте скоростной счетчик HSC0;

обнулите текущее значение таймера T32, записав в переменную T32 нуль. Запись слова осуществляется с помощью команды MOVW;

присоедините программу обработки прерывания INT_0 к прерыванию №21 (прерывание от таймера T32).

На каждом цикле контроллера:

если активен сигнал на входе I1.2, запустите таймер T32; выдержка времени 1 сек.

Программа обработки прерываний INT_0 будет «отображать» количество импульсов, отсчитанное скоростным счетчиком HSC0, сбрасывать счетчик и таймер.

Если активен сигнал на входе I1.2:

запишите в некоторую переменную, например, VD0, текущее значение счетчика HSC0. Обратиться к текущему значению можно по имени HC0;

запишите по соответствующему адресу области специальной памяти новое текущее значение счетчика HSC0, равное 0;

активизируйте скоростной счетчик HSC0;

обнулите текущее значение таймера.

Программа закончена. Загрузите ее в контроллер и переведите контроллер в режим RUN. Включите электропривод. Установите частоту напряжения, равную 50Гц. Включите выключатель, связанный с входом контроллера I1.2.

В Step 7 MicroWin включите режим отображения значений переменных. Наблюдайте значение VD0. Если все было сделано правильно, переменная VD0 должна быть немногим менее 1500 (за 1 сек 1500 импульсов).

Проведите эксперименты и заполните следующую таблицу:

Таблица 10. Результаты экспериментов

Частота, Гц	Показания (VD0)	Скорость (об/мин)	Скольжение, %
10			
20			
30			

Частота, Гц	Показания (VD0)	Скорость (об/мин)	Скольжение, %
40			
50			

Примечание: скольжение определяется по формуле:

$s = (\omega - \omega_c)/ \omega_c$,

где ω_c – синхронная скорость двигателя для данной частоты питания.

Определение временного интервала, в течение которого поступает заданное количество импульсов

Программа будет подсчитывать количество временных отчетов таймера в течение времени поступления 1500 импульсов с датчика.

Включите контроллер и запустите Step 7 MicroWin. Создайте новый проект. Удалите из проекта подпрограмму SBR_0.

В основной программе определите, сконфигурируйте и запустите скоростной счетчик HSC0, осуществите привязку программы обработки прерывания INT_0 к прерыванию от скоростного счетчика и запустите таймер T32:

<u>В первом цикле контроллера (если установлен бит SM0.1):</u>

запишите в управляющий байт счетчика HSC0 число, обеспечивающее:

1) прямой счет;
2) актуализацию направления счета;
3) актуализацию предустановленного значения;
4) актуализацию текущего значения;
5) разблокирование HSC0.

остальные опции настройки несущественны;

с помощью команды HDEF определите скоростной счетчик HSC0 в режиме 0;

запишите по соответствующему адресу области специальной памяти новое текущее значение счетчика HSC0, равное 0;

запишите по соответствующему адресу области специальной памяти новое предустановленное значение счетчика HSC0, равное 1500;

разрешите прерывания;

активизируйте скоростной счетчик HSC0;

обнулите текущее значение таймера T32;

присоедините программу обработки прерывания INT_0 к прерыванию №12 (прерывание от HSC0).

<u>На каждом цикле контроллера:</u>

если активен сигнал на входе I1.2, запустите таймер T32; выдержка времени – максимально большая: 32767×1мс = 32,767 сек.

Программа обработки прерываний INT_0 будет «отображать» количество временных отчетов, отсчитанных таймером T32, сбрасывать счетчик и таймер:

Если активен сигнал на входе I1.2:

запишите в некоторую переменную, например, VW0, текущее значение таймера T32;

запишите по соответствующему адресу области специальной памяти новое текущее значение счетчика HSC0, равное 0;

активизируйте скоростной счетчик HSC0;

обнулите текущее значение таймера T32.

Программа закончена. Загрузите ее в контроллер и переведите контроллер в режим RUN. Включите электропривод. Установите частоту напряжения, равную 50Гц. Включите выключатель, связанный с входом контроллера I1.2.

В Step 7 MicroWin включите режим отображения значений переменных. Наблюдайте значение VW0. Если все было сделано правильно, переменная VW0 должна принять значение немногим большее 1000 (1500 импульсов за 1 сек).

Проведите эксперименты и заполните следующую таблицу:

Таблица 11. Результаты экспериментов

Частота, Гц	Показания (VW0)	Скорость (об/мин)	Скольжение, %
10			
20			
30			
40			
50			

Содержание отчета

1. Принципиальная электрическая схема соединений цепей управления контроллера, преобразователя частоты и органов управления. На схеме должны быть приведены только используемые входы и выходы контроллера и преобразователя.

2. Программы контроллера в представлениях LAD, STL и FBD и временные диаграммы их работы.

3. Результаты экспериментов в виде заполненных таблиц 10,11.

Контрольные вопросы

1. Опишите работу оптического датчика скорости / угла поворота электропривода.

2. Каково назначение скоростных счетчиков Siemens S7-200? Чем они отличаются от обычных программных счетчиков?

3. Опишите команды управления скоростными счетчиками.

4. Опишите известные Вам режимы работы скоростных счетчиков.

5. Опишите возможные варианты входов скоростных счетчиков и их назначение.

6. Опишите последовательность программирования скоростных счетчиков.

7. Каково назначение управляющего байта скоростного счетчика.

8. Как устанавливаются новое текущее и новое предустановленное значение скоростного счетчика?

9. Какая информация хранится в байте состояния скоростного счетчика?

10. Опишите назначение прерываний и порядок работы контроллера при обработке прерываний.

11. Опишите команды прерываний Siemens S7-200.

12. Какие прерывающие события обрабатывались в программах, созданных при выполнении работы?

13. Опишите принципы (алгоритмы) определения скорости вращения электропривода, использованные в данной работе, их достоинства и недостатки

14. Опишите работу секций программы в представлении LAD.

15. Опишите работу секций программы в представлении STL.

16. Опишите работу секций программы в представлении и FBD.

5. Разработка и реализация системы регулирования частоты вращения электропривода

Цель работы: построение системы автоматического регулирования частоты вращения частотно-управляемого асинхронного электропривода.

Теоретические сведения

Система регулирования частоты вращения электропривода, разрабатываемая в данной лабораторной работе, будет использовать следующие элементы:

1) подсистему измерения частоты вращения вала двигателя;

2) подсистему воздействия на выходную частоту преобразователя частоты. Контроллер будет выдавать на дискретные входы преобразователя частоты сигналы «больше»/«меньше», получив которые, ПЧ будет увеличивать или уменьшать частоту;

3) подсистему задания и отображения скорости вращения электропривода, построенную на базе SCADA-системы Trace Mode.

1. Подсистема измерения скорости

Подсистема разработана при выполнении лабораторной работы № 4. Скорость измеряется путем определения числа импульсов, выдаваемых датчиком положения в течение заданного интервала времени (первый подход). Результат сохраняется в переменной VD0 (двойное слово).

2. Подсистема воздействия на выходную частоту ПЧ

Преобразователь частоты предусматривает несколько возможностей внешнего задания частоты питания двигателя, в том числе аналоговыми сигналами 0(2) – 10В и 0(4) – 20 мА. Однако, поскольку имеющаяся модель контроллера содержит только дискретные выходы, для управления выходной частотой в данной работе могут использоваться только дискретные входы ПЧ.

Изменять частоту с помощью сигналов на дискретных входах ПЧ можно в режиме «Мотор-потенциометр». Для перехода в данный режим из «Стандартного» (устанавливаемого на заводе-изготовителе) или любого другого необходимо демонтировать панель управления и лицевую панель преобразователя и установить переключатель в положение, показанное на рис. 1.

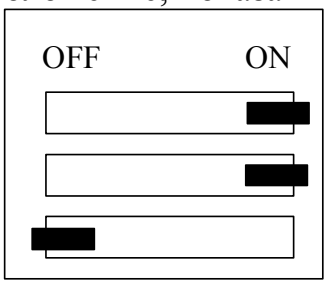


Рис.1. Положение переключателя в режиме «Мотор-потенциометр»

После установки переключателя установите панели на место.

Имеется два вида режима «Мотор-потенциометр»: собственно «Мотор-потенциометр» и «Мотор-потенциометр 2». Выбор между ними осуществляется

с помощью параметра программирования PARAM SET страницы параметров PAGE3.

В нашем случае следует установить PARAM SET = 1. Тогда назначение дискретных входов ПЧ будет соответствовать табл.1.

Таблица 1. Функции дискретных входов в режиме «Мотор-потенциометр»

Дискр. вход	Функция	Примечание
DI1	Старт	-
DI2	Реверс	-
DI3	Увеличение частоты	Ускорение определяется параметром ACC TIME2 (PAGE1)
DI4	Уменьшение частоты	Ускорение определяется параметром DEC TIME2 (PAGE1)
DI5	Постоянная скорость 1	-

Начальные значения частоты при пуске (реверсе) определяются параметром MIN FREQ (PAGE1). Торможение (при реверсе) и разгон производятся с ускорениями, соответствующими параметрам DEC TIME1 и ACC TIME1 (PAGE1). Значение параметра DEC TIME1 – время уменьшения частоты от максимальной, заданной параметром MAX FREQ (PAGE1), до минимальной, заданной параметром MIN FREQ. Это время может быть задано в диапазоне от 0,1 до 1800 сек. Аналогично, значение параметра ACC TIME1 – время увеличения частоты от минимальной до максимальной.

Торможение и разгон при подаче дискретных сигналов на входы DI4 и DI3 производятся с ускорениями, соответствующими параметрам DEC TIME2 и ACC TIME2 (PAGE1). Значения этих параметров имеют тот же смысл, что и значения параметров DEC TIME1 и ACC TIME1 соответственно.

Требуемые значения параметров MIN FREQ и MAX FREQ равны заводским установкам: 0 и 50 Гц. Параметры DEC TIME1 и ACC TIME1 не влияют на работу системы регулирования, поскольку определяют ускорение при пуске и реверсе. В данной работе пуск осуществляется при нулевой частоте, а реверс не используется. Выбрать значения параметров DEC TIME2 и ACC TIME2 предстоит в ходе выполнения лабораторной работы (для упрощения они будут равны друг другу).

Алгоритм работы программы контроллера, отвечающий за регулирование скорости, является простейшим релейным алгоритмом: если измеренная скорость электропривода меньше заданной (с учетом выбранной зоны нечувствительности Δ), контроллер подает сигнал «Увеличить частоту», если больше (при тех же условиях), – сигнал «Уменьшить частоту» (рис.2).

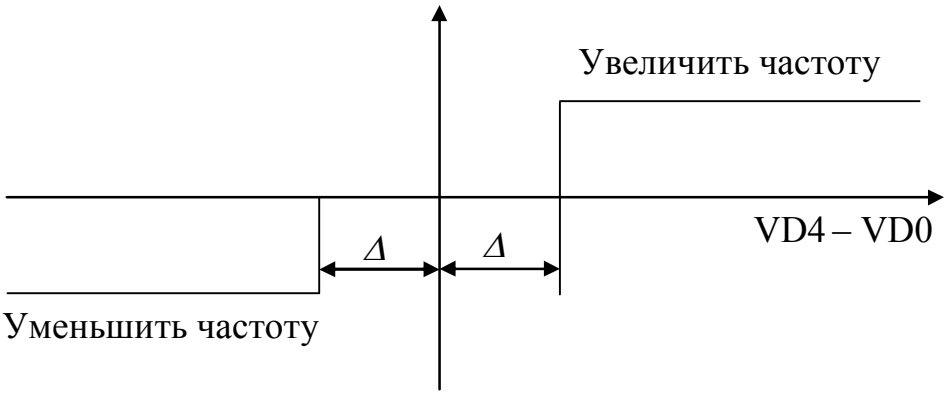


Рис. 2. Алгоритм регулирования частоты вращения

Измеренное значение скорости в виде числа импульсов датчика, поступивших в течение заданного времени, хранится в переменной VD0. В другой переменной типа двойное слово, например, в VD4, хранится заданное значение скорости, которое формируется и передается в контроллер SCADA-системой.

Величину зоны нечувствительности можно рассчитать исходя из следующих соображений.

Пусть частота вращения вала двигателя равна n об/мин, интервал времени между измерениями скорости t_u сек. Тогда значение VD0 равно $t_u \cdot n$. Заданная частота равна $n_{зад}$, причем $t_u(n_{зад} - n) > \Delta$. Контроллер дает команду преобразователю частоты увеличить частоту. Ускорение при увеличении частоты определяется параметром ACC TIME2, значение которого равно t_y сек/50 Гц. За время, равное периоду измерения скорости, выходная частота преобразователя вырастет на $50 \cdot t_u/t_y$ Гц. Приращение частоты вращения, если не учитывать скольжение двигателя, составит

$$\Delta n = 50 \cdot 60 \cdot t_u/(2 \cdot t_y) = 1500 \cdot t_u/t_y,$$

а приращение переменной VD0

$$\Delta_{VD0} = 1500 \cdot t^2_u / t_y.$$

Поскольку в разрабатываемой системе колебания скорости не желательны, необходимо, чтобы $\Delta_{VDO} < 2\Delta$. Из этого следует, что значение зоны нечувствительности можно определить из следующего неравенства:

$$\Delta > 750 \cdot t^2_u / t_y.$$

Ошибка регулирования частоты вращения

$$e = n_{зад} - n = \Delta / t_u > 750 \cdot t_u / t_y \text{ об/мин.}$$

Последнее выражение демонстрирует основной недостаток класса релейных систем регулирования, к которым принадлежит данная система: чем больше точность системы (меньше Δ) при заданном интервале измерения скорости t_u, тем меньше ее быстродействие (больше t_y).

В качестве ориентира можно использовать следующий вариант: $t_u = 0{,}1$ сек., $t_y = 10$ сек., тогда $\Delta = 1$ (Δ должно быть целым числом), $e = 10$ об/мин. Получим систему с временем регулирования не более 10 сек. и точностью отработки задания ±10 об/мин.

3. Подсистема задания и отображения скорости вращения электропривода

Подсистема строится на базе монитора реального времени Trace Mode. Экран монитора должен содержать средства ввода и изменения заданной частоты вращения и средства отображения текущей частоты. Дополнительно на экране могут быть размещены индикаторы увеличения и уменьшения частоты и другие элементы по усмотрению разработчиков.

Для обмена с контроллером Siemens S7-200 в Trace Mode существует драйвер, обеспечивающий поддержку протокола PPI (Point-to-Point Interface). Драйвер поддерживает чтение/запись данных во все области памяти контроллера – Stage, System Memory, Analog Input, Analog Output, Counter, Timer, High Speed Counter, Discrete Input, Discrete Output, Marker, Variable Memory. Для обмена данными используется последовательный порт компьютера и конвертер RS-232/RS-485. Драйвер оформлен в виде «драйвера для обмена по произвольному интерфейсу». Для корректной работы драйвера необходимо правильно произвести конфигурацию порта PPI в ПЛК и последовательного порта компьютера. Протокол PPI поддерживается в ПЛК серий S7-200.

Драйвер обеспечивает обмен данными как в режиме чтения, так и в режиме записи. Конфигурирование драйвера представляет собой указание параметров связи по последовательным портам. Для настройки драйвера используется утилита PPIconfig.exe. Каждой порт, указанный в списке, будет автоматически открыт драйвером. Страница конфигурации портов представлена на рис. 3.

Рис. 3. Страница конфигурации портов

В списке портов каждая строка состоит из 8-и параметров:

1) номер COM-порта. Повторное объявление одного и того же порта приведет к сообщению об ошибке при попытке сохранить конфигурацию;

2) скорость передачи данных (Baud Rate), от 300 bps до 115200 bps. Для устройств сети PPI по умолчанию принимается 9600 bps.;

3) число битов данных (Data Bits). По умолчанию установлено 8 бит;

4) контроль четности передачи (Parity), может принимать значения None, Odd или Even. По умолчанию для устройств сети PPI принимается Even;

5) количество стоп-битов (Stop Bits), 1 или 2. По умолчанию установлен 1 стоп-бит;

6) время тайм-аута для данного последовательного порта (в мс). По умолчанию – 1000 мс;

7) управление потоком. Используемый конвертер может требовать управления потоком. Для его корректной работы необходимо правильно указать сигналы (RTS, DTR), которые будут поданы перед каждой посылкой и сняты после ее отправки;

8) адрес Trace Mode в сети PPI. Согласно принципам обмена данными в сети PPI, каждое устройство должно иметь уникальный адрес.

Драйвер реализует обмен данными в сети Siemens PPI в режиме ведущего (Master) узла. Наличие нескольких ведущих узлов не поддерживается. Все остальные ПЛК должны быть ведомыми узлами. Архитектура сети – шина. Каждый узел в сети имеет свой уникальный адрес от 0 до 255. Режим работы сети – Polling Network. Ведущий узел отправляет запрос к ведомому, получает подтверждение о получении и через определенный промежуток времени запрашивает данные. Протокол PPI основан на электрическом интерфейсе RS-485. Поэтому подключение к стандартному последовательному порту ПК (RS-232) возможно только через конвертер RS-485/RS-232 согласно документации по коммуникационным портам ПЛК.

Для обмена данными необходимо создать каналы подтипа КОНТР_2 с дополнением к подтипу PPI. Каналы могут иметь вид представления H или F. Эта настройка определяет формат представления данных внутри системы Trace Mode. Возможен вариант, при котором будут прочитаны данные типа FLOAT в канал HEX. При этом дробная часть будет отброшена. Тип данных, которые читаются (записываются) в контроллер, определяется настройкой AREA. Тип канала (I или O) определяет его назначение – чтение данных или запись. Данные настройки каналов представлены на рис. 4.

Рис. 4. Каналы объекта

Эти каналы имеют следующие настройки:

1) RS# – номер последовательного порта. Значение 0 соответствует COM1, 1 – COM2 и т.д. Все используемые порты должны быть описаны на странице настроек портов в PPIconfig, а настройки сохранены в файл конфигурации. Номер порта задается в DEC-формате;

2) PLCaddr – адрес удаленного ПЛК в сети PPI. Уникальный параметр, определяющий устройство в сети. Диапазон значений – 0 .. 255. По умолчанию контроллер имеет адрес 2. Изменить эту настройку можно с помощью пакета программирования STEP7. Адрес ПЛК задается в DEC-формате;

3) Offset – смещение внутри области памяти контроллера. Смещение задается в байтах. Для получения правильных данных необходимо четко представлять принципы адресации в памяти контроллера и способы хранения данных.

Все данные, состоящие из нескольких байт, хранятся в памяти контроллера в формате Big Endian. Т.е. самый старший байт размещается первым, а самый младший – последним, пример на рис. 5.

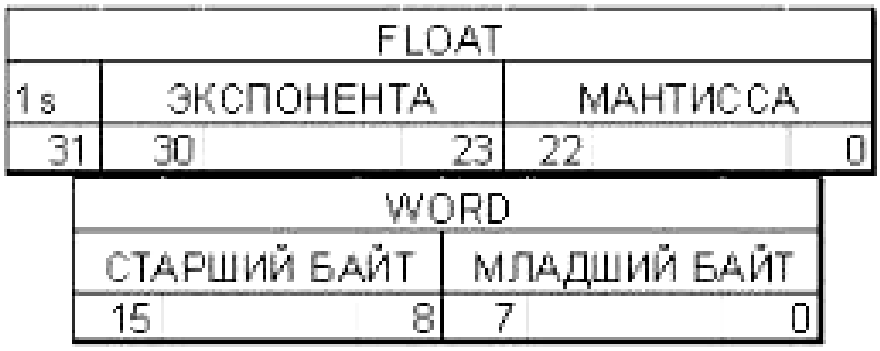


Рис. 5. Данные в памяти контроллера

Любая область памяти, кроме Timer, Counter и High Speed Counter, может рассматриваться как набор слов или чисел с плавающей точкой. Это определяется параметром AREA данного канала. В соответствии с этой настройкой драйвер запрашивает 2 или 4 байта из указанной области, преобразует их в формат Trace Mode и передает в систему в виде HEX или FLOAT. При этом 4

байта, прочитанные из памяти контроллера как FLOAT, могут рассматриваться далее в системе как число с плавающей точкой (FLOAT) или как целое (HEX). Во втором случае дробная часть будет отброшена. Аналогично, прочитанные как WORD 2 байта могут далее рассматриваться в Trace Mode как целое (HEX) или как число с плавающей точкой (FLOAT), однако дробная часть его всегда будет равна 0. Во избежание ошибок следует указывать вид представления HEX для данных типа WORD и тип FLOAT для данных типа FLOAT.

Настройка Offset (Смещение) указывает порядковый номер байта внутри указанной области, который является первым в представлении переменной. При этом могут иметь место пересечения данных (рис. 6).

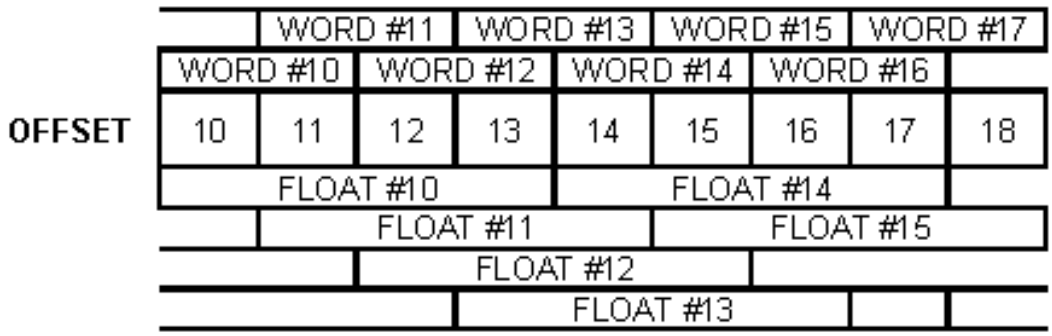


Рис. 6. Пересечения данных в S7 200

Как видно из рисунка, изменение значения слова со смещением 10 может повлечь за собой непредвиденное изменение слова со смещением 11. Аналогично и для данных с плавающей точкой, но в этом случае затрагиваются 4 последующих значения. Во избежание ошибок рекомендуется обращаться к целочисленным данным с четным смещением, а к данным с плавающей точкой – со смещением, кратным 4.

Области Timer, Counter и High Speed Counter являются структурированными. Смещение в этом случае определяет порядковый номер элемента. Элементы не пересекаются. Кроме значения, элемент содержит бит статуса. Для доступа к этому биту используются настройки Timer(Status), Counter(Status) и High Speed Counter(Status). Для High Speed Counters он всегда равен 0.

Смещение задается в HEX-формате.

4) Area – область памяти контроллера. Драйвер позволяет получить доступ ко всем существующим областям памяти контроллера: Stage, System Memory, Analog Input, Analog Output, Counter, Timer, High Speed Counter, Discrete Input, Discrete Output, Marker, Variable Memory (рис. 7). Кроме того, любая область памяти, кроме Timer, Counter и High Speed Counter, может рассматриваться как массив целых 2-байтных чисел (WORD) или 4-х байтных чисел с плавающей точкой (FLOAT). Это определяется выбранным параметром Area и не зависит от типа канала (H или F). Данные в областях Timer, Counter хранятся только в целочисленном формате. Данные в области High Speed Counter хранятся как 4-байтные двойные слова, поэтому для получения максимального количества значащих цифр необходимо использовать представление канала F. Кроме того, перечисленные области имеют бит статуса, но в области High Speed Counter он

не используется, поэтому всегда равен 0. Некоторые области доступны только для чтения, другие – только для записи.

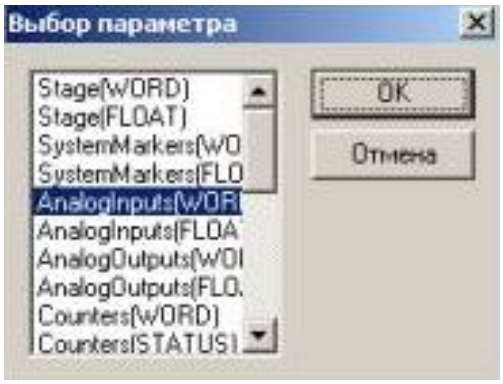


Рис. 7. Выбор параметра

Таблица 2. Область данных

AREA	WORD		FLOAT		STATUS	
	READ	WRITE	READ	WRITE	READ	WRITE
Stage	+	+	+	+	X	X
System Memory	+	+ *	+	+ *	X	X
Analog Input	+	-	+	-	X	X
Analog Output	-	+	- **	- **	X	X
Counter	+	+	- **	- **	+	-
Timer	+	+	- **	- **	+	-
High Speed Counter	+	-	- **	- **	+ ***	-
Discrete Input	+	+	+	+	X	X
Discrete Output	+	+	+	+	X	X
Markers	+	+	+	+	X	X
Variable Memory	+	+	+	+	X	X

* - SM0 .. SM29 доступны только для чтения
** - эти области рассматриваются только как целые
*** - бит статуса HighSpeedCounter всегда равен 0

Чтение данных происходит по запросу сервера матобработки в соответствии с фазой и периодом работы канала. Запись – при изменении выходного значения канала типа O. При каждой посылке ответ ожидается в течение заданного таймаута. Если ответа за это время нет, то каналу выставляется признак достоверности, равный 1. Вне зависимости от того, была ли попытка успешной или нет, драйвер возвращает управление серверу матобработки. Необходимо обратить внимание на то, что при неуспешной попытке записи значения канала Trace Mode будет пытаться повторить запись до тех пор, пока она не пройдет успешно.

В случае возникновения ошибок рекомендуется включить в конфигураторе опцию Вести журнал событий и проанализировать протокол обмена.

Задание

1. Разработать и собрать принципиальную электрическую схему соединений для реализации системы регулирования.

2. Перевести преобразователь частоты в режим управления «Мотор-потенциометр» и произвести установку параметров программирования ПЧ.

3. Разработать и ввести в контроллер программу управления.

4. Разработать монитор реального времени Trace Mode.

5. Апробировать систему и продемонстрировать ее работу преподавателю.

Содержание отчета

1. Принципиальная электрическая схема соединений цепей управления контроллера, преобразователя частоты и персонального компьютера. На схеме должны быть приведены только используемые входы и выходы.

2. Список установленных параметров преобразователя частоты с их значениями.

3. Программы контроллера в представлениях LAD, STL и FBD.

4. База каналов и экранные формы монитора реального времени Trace Mode с их настройками.

5. Результаты экспериментов в виде заполненной таблицы 3 (3 – 4 строки):

Таблица 3. Результаты экспериментов

№	Заданная скорость (рад/сек)	Зона нечувствительности (рад/сек)	Действительная скорость (рад/сек)

Контрольные вопросы

1. Чем определяется реакция преобразователя частоты на сигналы на его дискретных входах?

2. Опишите назначение дискретных входов преобразователя частоты в режиме «Мотор-потенциометр».

3. Опишите алгоритм выбора ускорения при уменьшении/увеличении частоты напряжения питания электропривода.

4. Какие факторы определяют точность регулирования частоты вращение и быстродействие системы?

5. Опишите каналы Trace Mode, используемые в системе.

6. Разработка системы регулирования угла поворота электропривода

Цель работы: построение системы автоматического регулирования угла поворота частотно-управляемого асинхронного электропривода.

Теоретические сведения

Система регулирования угла поворота, разрабатываемая в данной лабораторной работе, будет использовать следующие элементы:

1) подсистему измерения угла поворота;

2) подсистему воздействия на выходную частоту преобразователя частоты. Контроллер будет выдавать на дискретные входы преобразователя частоты сигналы «пуск», «реверс», «переход на пониженную скорость»;

3) подсистему задания и отображения угла поворота, построенную на базе SCADA-системы Trace Mode.

1. Подсистема измерения угла поворота

Подсистема строится на базе скоростного счетчика Siemens S7 200. В отличие от предыдущих лабораторных работ при управлении счетчиком будет использоваться механизм изменения направления счета. Направление счета скоростного счетчика будет изменяться под воздействием внешнего сигнала, поступающего на один из дискретных входов контроллера. Этот сигнал будет формировать преобразователь частоты с помощью одного из дискретных (релейных) выходов при изменении направления вращения двигателя.

Для реализации подсистемы необходимо:

1) осуществить подсоединение релейного выхода преобразователя частоты к дискретному входу контроллера (см. лаб. работу №3). Номер дискретного входа контроллера определяется номером счетчика (см. лаб. работу №4). Для счетчика HSC0 используется вход I0.1;

2) запрограммировать релейный выход ПЧ на срабатывание при изменении направления вращения двигателя. Для этого требуется установить параметр 1.RELAY (для первого выхода) или 2.RELAY (для второго выхода) страницы PAGE2 равным 6 («Двигатель вращается вперед», реле обесточивается, если двигатель вращается вперед);

3) запрограммировать скоростной счетчик на работу в режиме 3 («Однофазный счетчик с внешним управлением направлением»);

2. Подсистема воздействия на выходную частоту преобразователя частоты реализует алгоритм регулирования, представленный на рис. 1.

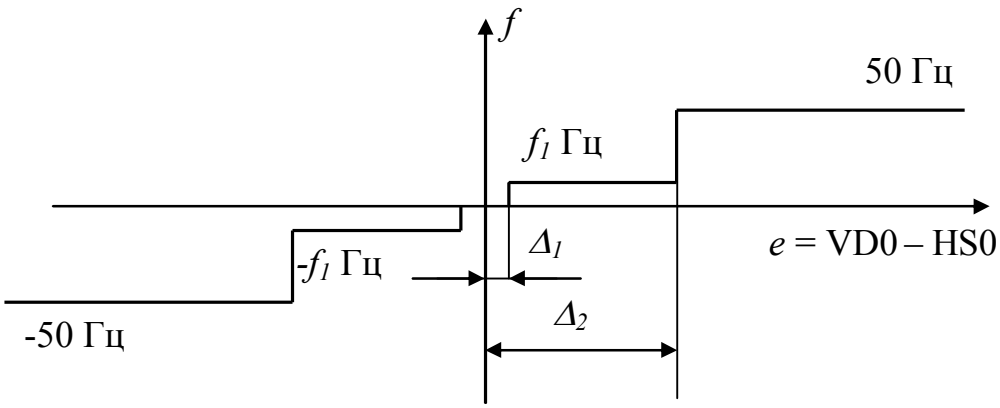


Рис. 1. Алгоритм регулирования угла поворота

Ошибка регулирования определяется разностью между переменной VD0, в которую SCADA-система заносит значение заданного угла поворота, и переменной HC0, в которой находится текущее значение скоростного счетчика HSC0 (если используется HSC0). Если модуль разности превосходит величину Δ_2, привод включается на максимальную скорость в ту ли иную сторону вращения в зависимости от знака ошибки. При $|e| < \Delta_2$ производится переход на пониженную скорость для уменьшения величины выбега привода при остановке. При $|e| < \Delta_1$ привод отключается. Воздействие на преобразователь частоты производится с помощью трех дискретных команд: «пуск», «реверс», «переход на пониженную (фиксированную) скорость», как в лаб. работе №3. С помощью переключателя, расположенного под панелью управления преобразователя, последний должен быть переведен в «Стандартный» режим (см. лаб. работу №3).

Для точной остановки привода следует:

1) принять минимально возможное значение зоны нечувствительности алгоритма Δ_1;

2) использовать максимально возможное ускорение при торможении двигателя с максимальной частоты до пониженной (см. лаб. работу №5);

3) использовать торможение двигателя постоянным током при остановке. Для этого следует установить параметр STOP страницы параметров PAGE 2 равным «DC Brake»;

4) использовать механизм прерываний для ускорения реакции контроллера в момент перехода ошибки через ноль. Необходимо установить предустановленное значение скоростного счетчика равным заданному значению и разрешить прерывания при равенстве текущего значения предустановленному. В подпрограмме обработки прерывания предусмотреть отключение привода с помощью команды «Reset». Основная программа должна производить перепрограммирование скоростного счетчика при каждом изменении заданного угла поворота со стороны SCADA-системы.

3. 2. Подсистема задания и отображения угла поворота строится аналогично подобной подсистеме в лаб. работе №5. SCADA-система считывает те-

кущее значение угла поворота из переменной HC0 (при использовании скоростного счетчика HCS0) и записывает заданное значение в переменную VD0.

Задание

1. Разработать и собрать принципиальную электрическую схему соединений для реализации системы регулирования.

2. Перевести преобразователь частоты в режим управления «Стандартный» и произвести установку параметров программирования ПЧ.

3. Разработать и ввести в контроллер программу управления.

4. Разработать монитор реального времени Trace Mode.

5. Апробировать систему и экспериментально определить минимальное значение Δ_2 при заданной фиксированной частоте f_1. В качестве ориентира в первом приближении можно использовать следующие значения: f_1 = 1 Гц, Δ_2 = 150 (число импульсов в течение 1 сек. при движении на скорости, соответствующей данной частоте без учета скольжения).

6. Продемонстрировать работу системы преподавателю.

Содержание отчета

1. Принципиальная электрическая схема соединений цепей управления контроллера, преобразователя частоты и персонального компьютера. На схеме должны быть приведены только используемые входы и выходы.

2. Список установленных параметров преобразователя частоты с их значениями.

3. Программы контроллера в представлениях LAD, STL и FBD.

4. База каналов и экранные формы монитора реального времени Trace Mode с их настройками.

5. Результаты экспериментов в виде заполненной таблицы 1 (3– 4 строки).

Таблица 1. Результаты экспериментов

№	Δ_2	f_1, Гц

Контрольные вопросы

1. Опишите принципиальную схему соединений контроллера.

2. Опишите схему взаимодействия контроллера и преобразователя частоты.

3. Опишите алгоритм регулирования угла поворота.

4. Опишите режим №3 работы скоростного счетчика Siemens S7-200.

5. Какие параметры закона регулирования определяют точность и быстродействие системы, и каким образом?

7. Разработка и реализация программы управления роботом-манипулятором для контроллера Siemens S7-200

Цель работы: получение навыков реализации систем программно-логического управления.

Теоретические сведения

Робот-манипулятор

Кинематическая структура робота-манипулятора имеет вид, показанный на рис. 1.

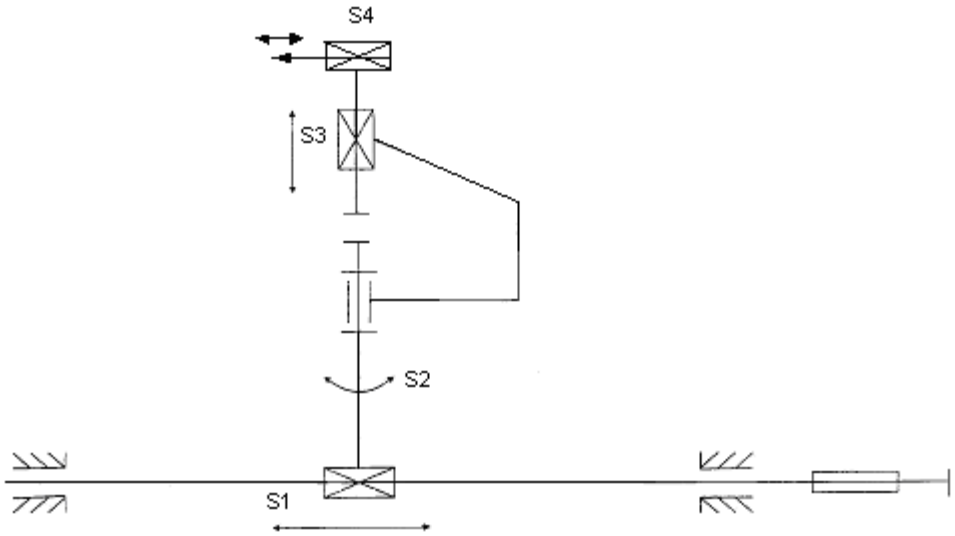


Рис. 1. Кинематическая схема

Манипулятор состоит из трех кинематических пар: вращательной S2 и поступательных S1 и S3, а также механизма изменения положения рабочего органа (пишущей насадки) S4. Последний представляет собой соленоид, благодаря которому рабочий орган может занимать два положения: «втянут» и «выдвинут».

Внешний вид установки представлен на рис. 2.

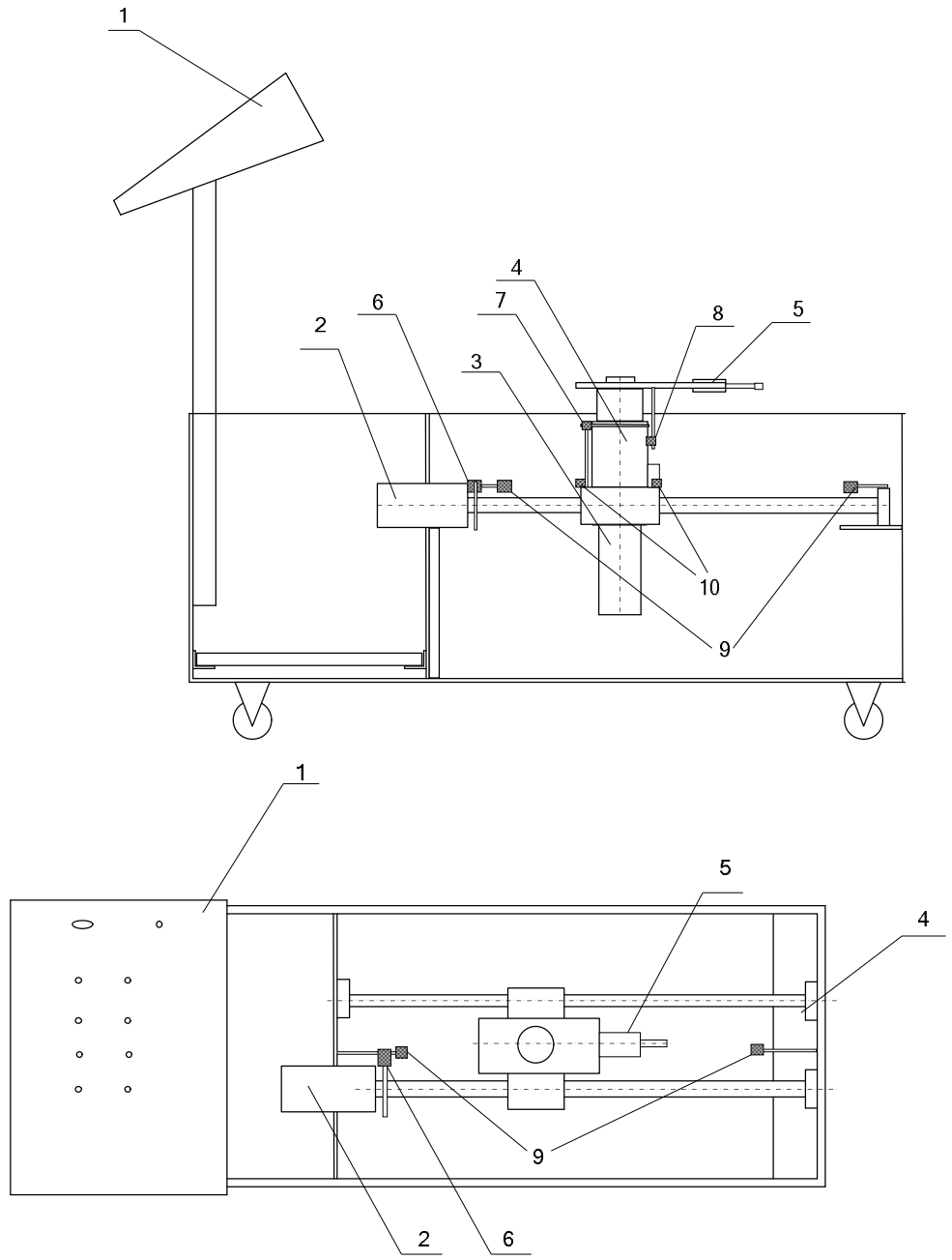


Рис. 2. Внешний вид робота-манипулятора

Обозначения на рис. 2:

1 – пульт ручного управления;

2 – электропривод перемещения по горизонтальной оси (координата X);

3 – электропривод поворота вокруг вертикальной оси (координата Y);

4 – электропривод перемещения по вертикальной оси (координата Z);

5 – соленоид;

6 – датчик перемещения по координате X;

7 – датчик перемещения по координате Y;

8 – датчик перемещения по координате Z;

9 – концевые выключатели движения по координате X;

10 – концевые выключатели движения по координате Y.

Приводной электромеханический модуль для вращательной кинематической пары разработан на базе серийного электромеханизма МЗК-2, для поступательных пар – на базе электромеханизмов МП-100 и МЗК-3.

Используемые в электромеханизмах двигатели являются двигателями постоянного тока с последовательным возбуждением. Двигатели рассчитаны на питание напряжением не более 27 В. Они имеют на своем валу тормозную муфту, затормаживающие вал двигателя при снятии напряжения. На двигателе МЗК-3 в силовой цепи установлены концевые выключатели.

Рабочий орган робота-манипулятора приводится в движение электромагнитным соленоидом фирмы Toyota. Соленоид рассчитан на работу от источника постоянного тока напряжением 12 В. Механическая часть соленоида выдвигается при подаче на него напряжения определенной полярности и задвигается при изменении полярности.

Для питания механизмов и вспомогательных устройств робота-манипулятора используется блок питания. Он смонтирован отдельным блоком, располагается в нижней части управляющей стойки.

Блок питания состоит из трансформатора, двух диодных мостов, расположенных на одной плате и емкостных фильтров к ним.

Диодные мосты (построенные на диодах Д204) служат для выпрямления напряжения, подаваемого с трансформатора. Для выравнивания уровня выдаваемого напряжения используются ёмкостные фильтры (конденсаторы 50В×2000мкФ).

Принципиальная электрическая схема питания изображена на рис. 3.

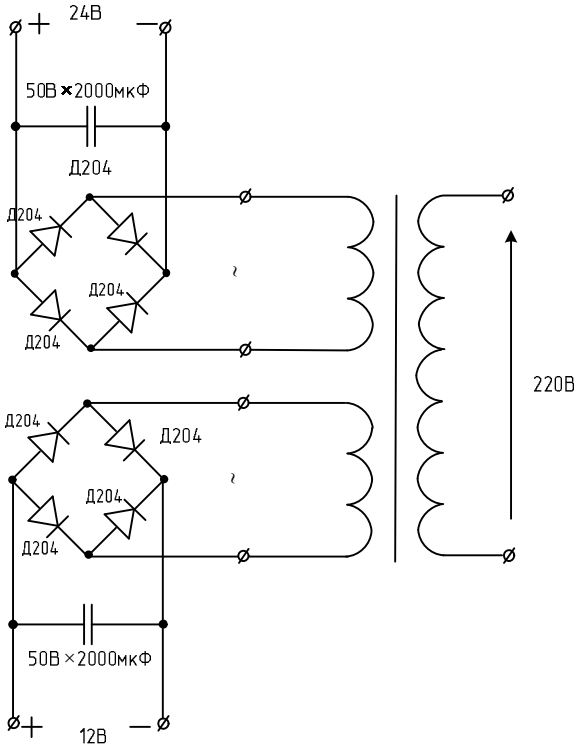


Рис. 3. Принципиальная электрическая схема питания

Диодные мосты и ёмкостные фильтры располагаются на одной плате и крепятся к плате питания при помощи двух шурупов.

Все питающие напряжения снимаются с общего трансформатора и подаются на соответствующие диодные мосты, затем фильтруются и через разъемы попадают на все функциональные блоки робота.

Система коммутации электроприводов построена на реле РЭС22, концевых выключателях двигателей и кнопках КП-3.

Система исключает включение одного привода в двух направлениях, предотвращая появления короткого замыкания на обмотках двигателя, защищая привод от поломки.

При построении пульта ручного управления используются кнопки КП-3 и переключатель ПК-1.

Каждая из кнопок отвечает за включение каждого из приводов в одном из направлений. Переключатель КП-1 служит для переключения манипулятора из автоматического (управление от контроллера) в ручной режим. Реле РЭС-22 необходимы для коммутации приводов, а также для организации системы взаимной блокировки.

Для предотвращения продолжения работы двигателей при достижении ими крайних положений в систему коммутации электроприводов включены концевые выключатели.

Нужно отметить, что на электроприводе МЗК-3 концевые выключатели установлены в силовую цепь. Остальные четыре концевых выключателя (по два на горизонтальной оси и по окружности) установлены на неподвижных платформах и включены в управляющую цепь.

Схема силовых цепей робота-манипулятора показана на рис. 4, общая схема цепей коммутации на рис. 5.

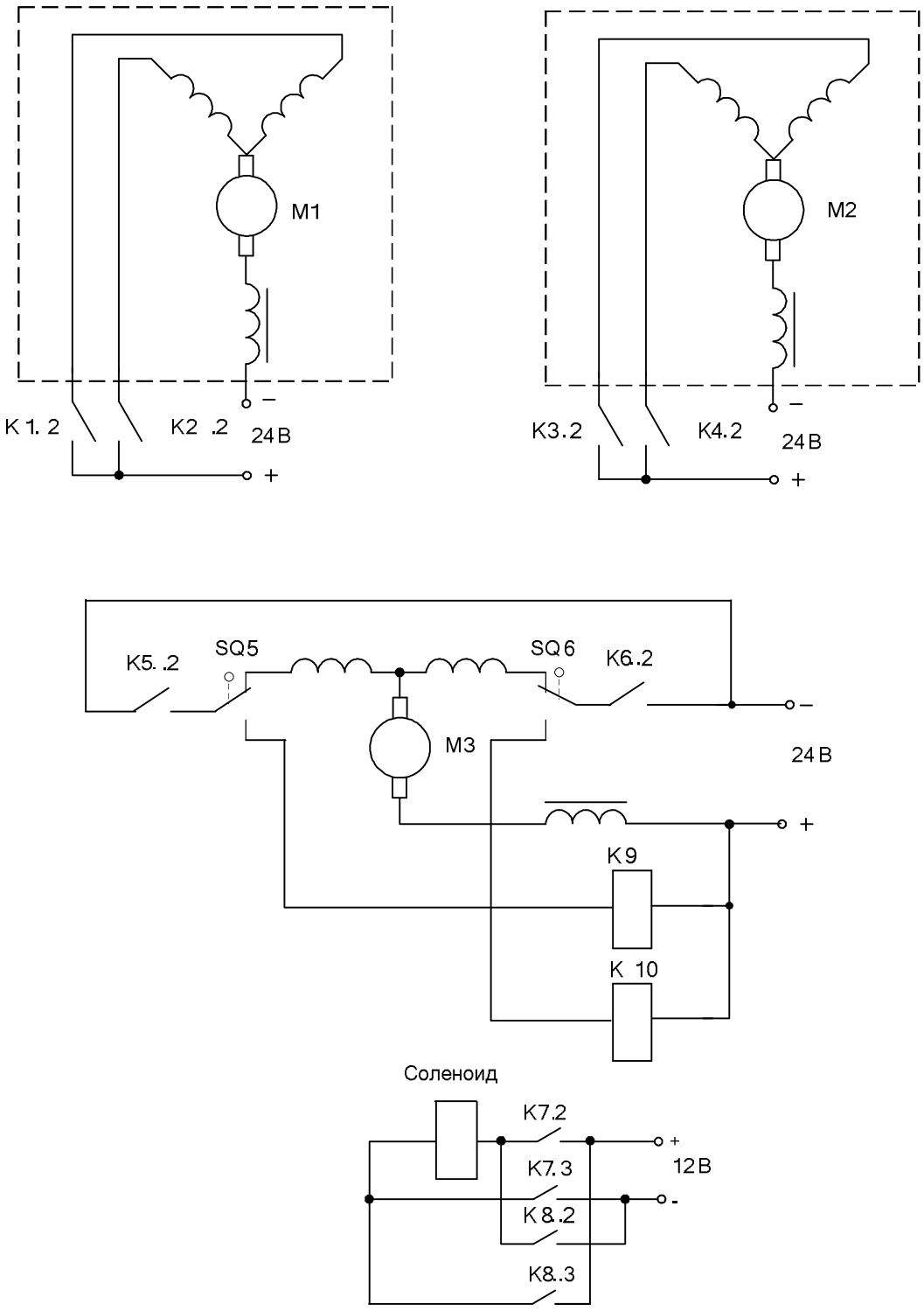


Рис. 4. Схема силовых цепей

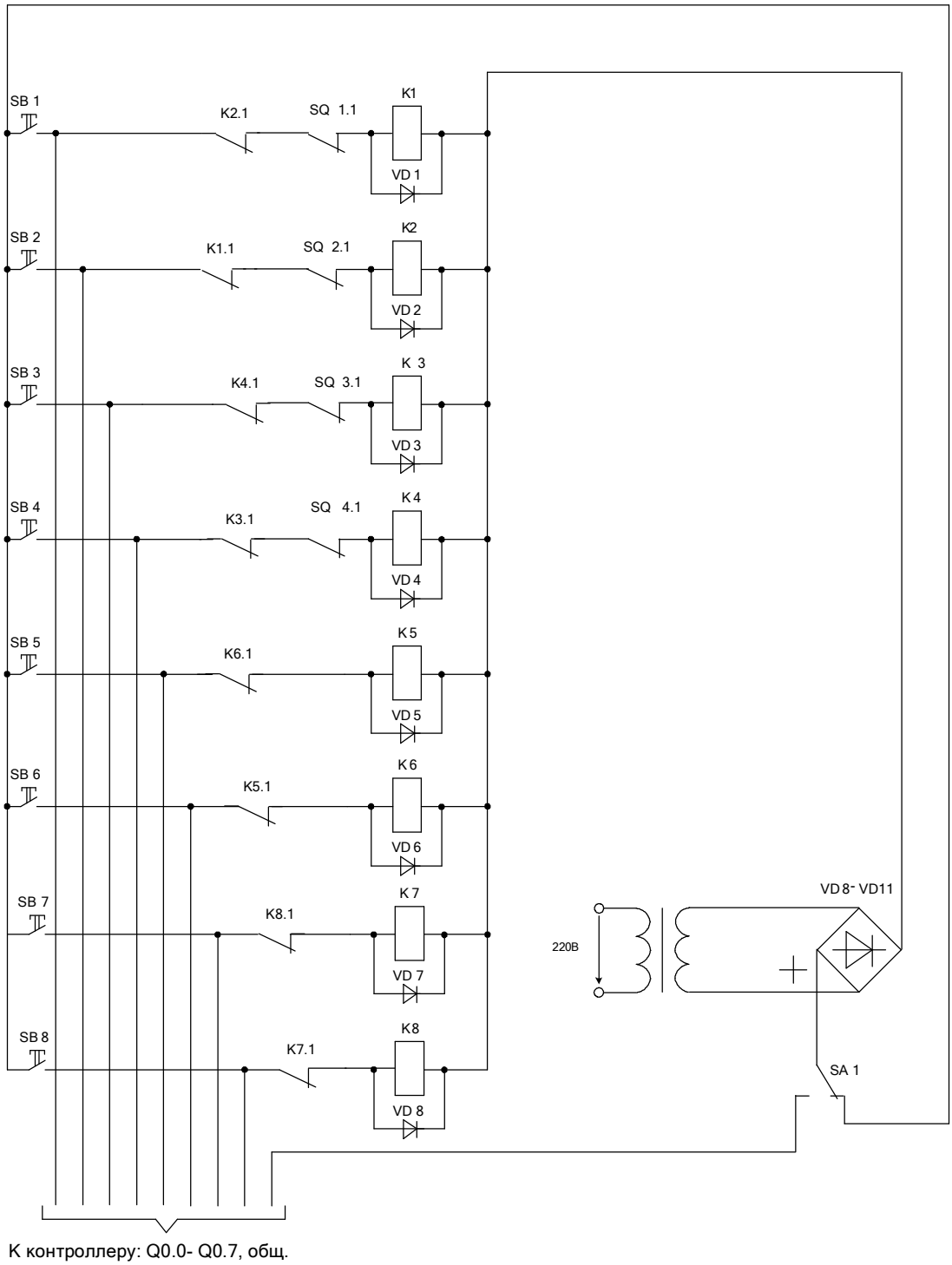


Рис.5. Схема цепей коммутации (ручное управление)

В качестве датчиков положения выбраны оптические пары светодиод – фотодиод из манипулятора «мышь».

Светодиоды, подключённые к цепи постоянного тока, излучают ИК-лучи; фотодиоды, принимая сигнал, меняют своё сопротивление, и далее схема обработки сигнала создаёт необходимое напряжение на входе контроллера.

В качестве прерывающих ИК-сигнал решёток используются различные для каждого из датчиков приспособления.

Вследствие того, что используемая нами модификация контроллера Siemens S7-200 имеет лишь дискретные входы, для обработки сигналов датчиков

применена схема транзисторного ключа. Таким образом, датчики будут выдавать, а контроллер принимать и подсчитывать импульсы. В схему обработки сигналов датчиков включены добавочные переменные сопротивления в цепи светодиодов. В связи с этим появляется возможность изменения силы тока, а, следовательно, и уровня излучения светодиодов в процессе наладки узлов робота-манипулятора.

В манипуляторе измеряется три координаты: по горизонтали, вертикали и угловое перемещение. Всего используется три датчика и три транзисторных ключа. Общая схема системы измерения положения рабочего органа изображена на рис. 6.

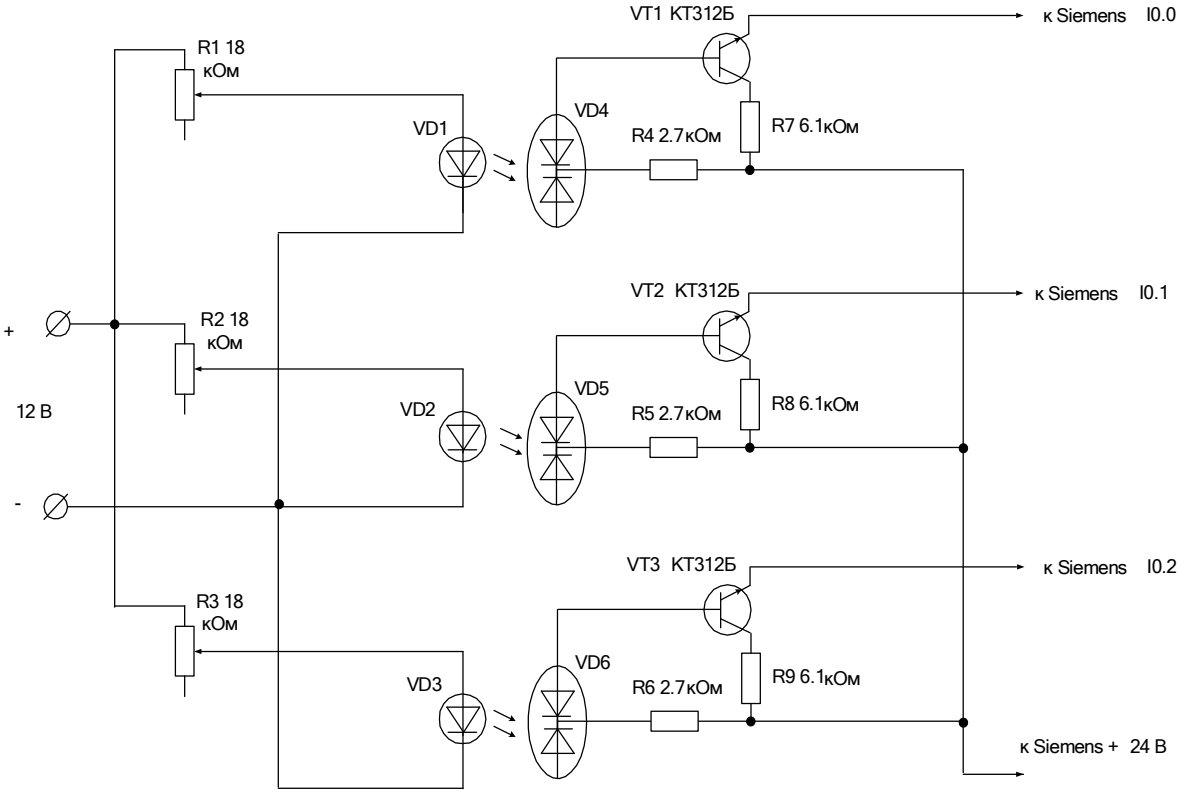


Рис. 6. Схема системы измерения положения рабочего органа

Сопряжение робота-манипулятора с лабораторным стендом

К стенду Siemens S7-200 робот манипулятор присоединяется при помощи многожильного кабеля, на одной стороне которого располагаются штекеры для подключения к гнездам лабораторного стенда. Нумерация штекеров приведена в табл.1.

Таблица 1. Нумерация штекеров

№	Тип	Назначение
1	Вход	Концевой выключатель по часовой
2	Вход	Концевой выключатель справа
3	Вход	Датчик поворота по окружности
4	Вход	Датчик движения по горизонтали
5	Вход	Датчик движения по высоте

№	Тип	Назначение
6	Вход	Концевой выключатель верхний
7	Вход	Концевой выключатель нижний
8	Вход	Концевой выключатель слева
9	Вход	Концевой выключатель против часовой
1А	Вход	Включен привод по X вперед
2А	Вход	Включен привод по Y почасовой
3А	Вход	Включен привод по Z вверх
4А	Вход	Шток «втягивается» (сол. под напряжением)
5А	Вход	Шток «выдвигается» (сол. под напряжением)
П	Вход	Кнопка «Память»
10	Вход	Общий входов
11	Выход	Реле поворота по часовой (Y)
12	Выход	Реле поворота против часовой (Y)
13	Выход	Реле пуска влево (X)
14	Выход	Реле пуска вправо (X)
15	Выход	Общий (GND)
16	Выход	Реле пуска вниз (Z)
17	Выход	Реле пуска вверх (Z)
18	Выход	Реле выдвижения штока соленоида
19	Выход	Реле втягивания штока соленоида

Входы 1А – 5А, П предназначены для специфических приложений, связанных с самообучением системы путем ввода программы перемещений непосредственно с пульта ручного управления роботом. В данной работе они не используются.

На рис. 7. показана рекомендуемая схема подключения робота манипулятора к стенду.

Входы и выходы контроллера используются следующим образом.

I0.0 – сигнал с датчика X;
I0.1 – сигнал с датчика Y;
I0.2 – сигнал с датчика Z;
I0.3 – сигнал КВ X вперед;
I0.4 – сигнал КВ X назад;
I0.5 – сигнал КВ Y по часовой;
I0.6 – сигнал КВ Y против часовой;
I0.7 – сигнал КВ Z вверх;
I1.0 – сигнал КВ Z вниз;
I2.0 – сигнал о срабатывании реле привода по оси X;
I2.1 – сигнал о срабатывании реле привода по оси Y;
I2.2 – сигнал о срабатывании реле привода по оси Z;
I2.3 – сигнал реле штока (шток втянут);
I2.4 – сигнал реле штока (шток вытянут);
I2.5 – кнопка «память»;

Q0.0 – выход X пуск назад;
Q0.1 – выход X пуск вперед;
Q0.2 – выход Y пуск против часовой;
Q0.3 – выход Y пуск по часовой;
Q0.4 – выход Z пуск вниз;
Q0.5 – выход Z пуск вверх;
Q0.6 – втянуть шток соленоида;
Q0.7 – выдвинуть шток соленоида.

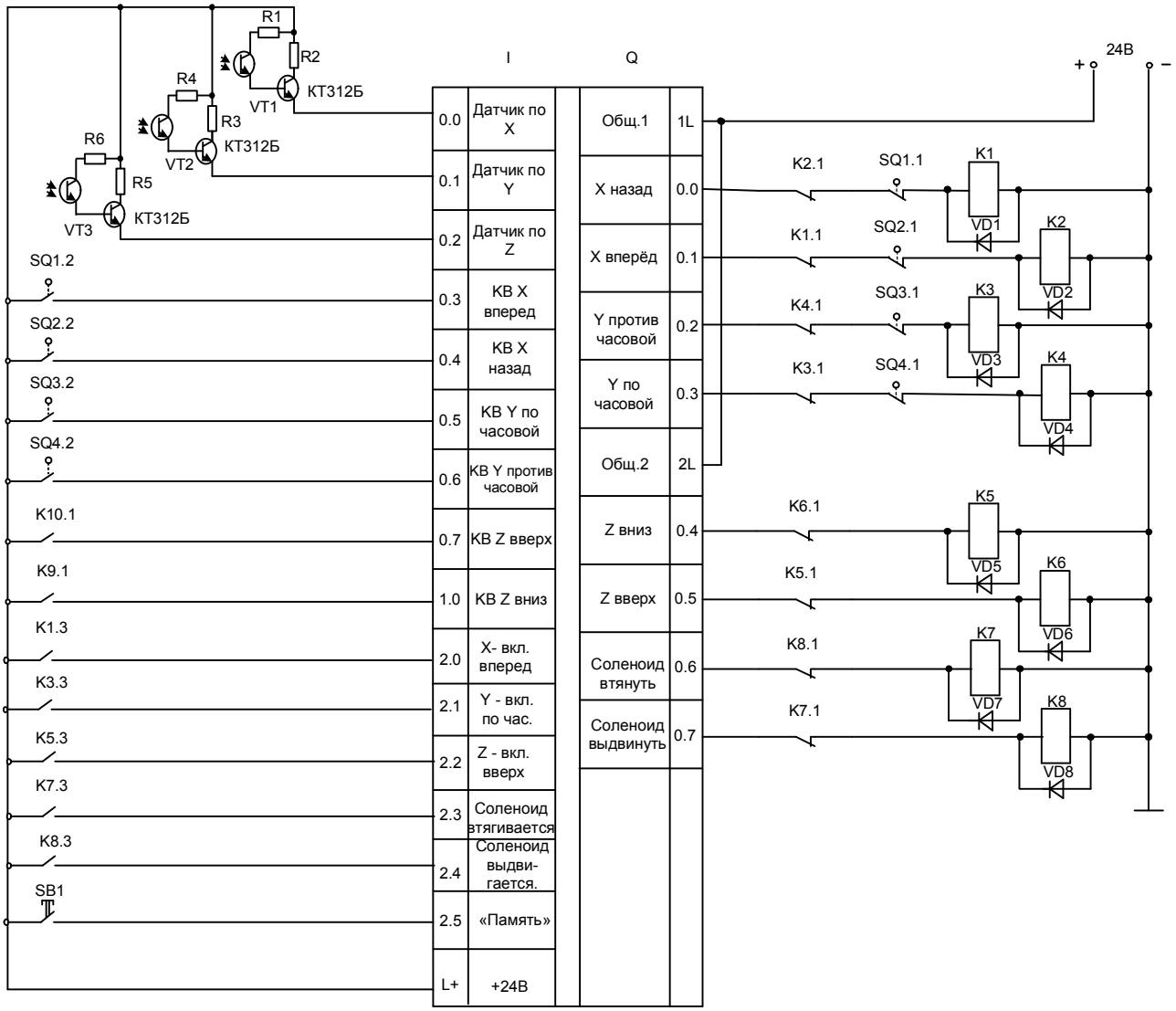


Рис. 7. Рекомендуемая схема подключения робота манипулятора к стенду

Описание используемых команд контроллера

В STEP 7 находится множество команд, с помощью которых можно создавать довольно сложные системы автоматического управления. Ниже рассмотрены только те, которые будут использоваться в программе управления.

Битовые логические операции представлены на рис.8.

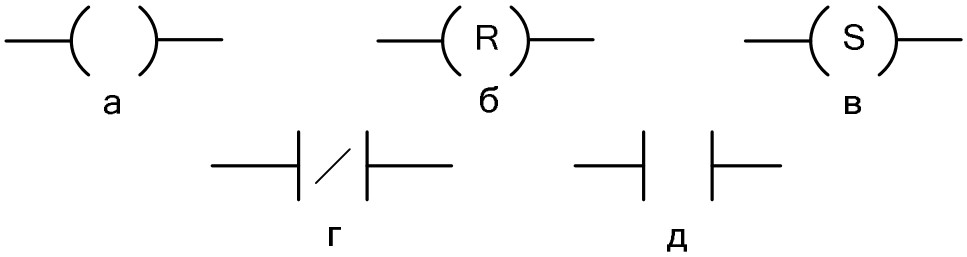


Рис. 8. Основные битовые логические операции

Команды «Нормально открытый контакт» (рис. 8, г), (LD, A и O) и «Нормально замкнутый контакт» (рис.8, д), (LDN, AN, ON) получают исходное значение из памяти или из регистра образа процесса (здесь и далее в скобках указывается названия команд на языке STL).

Нормально открытый контакт замкнут (включен), когда бит равен 1, а нормально замкнутый контакт замкнут (включен), когда бит равен 0. Команды, представляющие нормально открытый контакт, загружают значение адресного бита в вершину стека или выполняют логическое сопряжение значения адресного бита со значением в вершине стека в соответствии с таблицей истинности логического И или ИЛИ, а команды, представляющие нормально замкнутый контакт, 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 присваивания (рис. 8,а, (=) записывает новое значение для бита памяти или выходного бита в регистре образа процесса.

Команды установки (рис. 8, б), (S) и сброса (рис. 8, в), (R) устанавливают (включают) или сбрасывают (выключают) указанное количество битов, начиная с указанного адреса (бита). Можно установить или сбросить от 1 до 255 входов и выходов.

Если команда сброса указывает на бит таймера (T) или счетчика (C), то команда сбрасывает бит таймера или счетчика и стирает текущее значение таймера или счетчика.

Команды сравнения используются для сравнения двух величин:

IN1 = IN2, IN1 >= IN2, IN1 <= IN2,

IN1 > IN2, IN1 < IN2, IN1 <> IN2.

Операции сравнения байтов не учитывают знака.

Операции сравнения целых учитывают знак.

Если сравнение истинно, то команда сравнения загружает «1» в вершину стека или выполняет логическое сопряжение значения «1» со значением в вершине стека в соответствии с таблицей истинности для И или ИЛИ .

Счетчики

S7–200 имеет в своем распоряжении три вида счетчиков, которые подсчитывают нарастающие фронты на счетных входах счетчика: один вид счетчиков ведет прямой счет, другой считает только в обратном направлении, а третий вид считает в обоих направлениях. Со счетчиком связаны две переменные:

текущее значение: это 16-битовое целое со знаком хранит счетное значение, накопленное счетчиком;

бит счетчика: этот бит устанавливается или сбрасывается, когда текущее значение становится равным предустановленному значению. Предустановленное значение вводится как часть команды счетчика.

Обращение к обоим этим элементам данных производится через адрес счетчика (C + номер счетчика). Происходит ли обращение к биту счетчика или к текущему значению, зависит от используемой команды: команды с операндами в битовом формате обращаются к биту счетчика, тогда как команды с операндами в формате слова обращаются к текущему значению.

Реверсивный счетчик (рис.9), (CTUD) ведет счет вверх при положительном фронте на входе прямого счета (CU) или ведет счет вниз при положительном фронте на входе обратного счета (CD). PV – предварительно установленное значение. Счетчик сбрасывается, когда включается сброс (R). Счетчик прекращает счет, когда он достигает предварительно установленного значения или нуля.

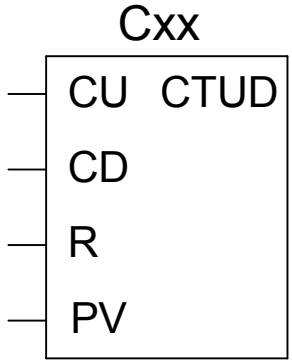


Рис. 9. Команда реверсивный счетчик

Таймеры

S7–200 имеет в своем распоряжении таймеры, которые отсчитывают приращения времени с разрешениями (шагами базы времени) 1 мс, 10 мс или 100 мс. С таймером связаны две переменные:

текущее значение: это 16-битовое целое со знаком хранит количество времени, отсчитанное таймером;

бит таймера: этот бит устанавливается или сбрасывается, когда текущее значение становится равным предустановленному значению. Предустановленное значение вводится как часть таймерной команды.

Обращение происходит к обоим этим элементам данных через адрес таймера (T + номер таймера). Происходит ли обращение к биту таймера или к текущему значению, зависит от используемой команды: команды с операндами в битовом формате обращаются к биту таймера, тогда как команды с операндами в формате слова обращаются к текущему значению.

Команды «Таймер с задержкой включения» (TON) и «Таймер с задержкой включения с запоминанием» (TONR) отсчитывают время, когда включен разрешающий вход. Номер таймера (Txx) определяет его разрешающую способность, и эта разрешающая способность теперь отображается в блоке команды.

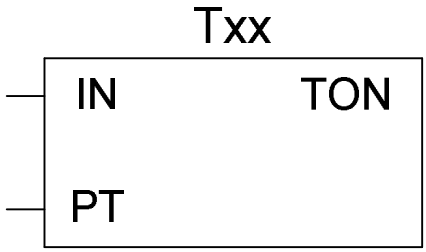


Рис. 10. Таймерная команда

Таймер с задержкой выключения (TOF) используется для задержки выключения выхода на фиксированный интервал времени после выключения входа.

Три вида таймеров выполняют различные задачи измерения времени:

таймер с задержкой включения TON может использоваться для отсчета отдельного интервала.

таймер с задержкой включения с запоминанием TONR может использоваться для накапливания нескольких отсчитанных интервалов времени.

таймер с задержкой выключения TOF может использоваться для увеличения интервала времени после выключения (или сбоя), например, для охлаждения двигателя после его отключения.

Команды TON и TONR отсчитывают время, когда включен разрешающий вход. Когда текущее значение становится больше или равно предустановленному времени, бит таймера устанавливается.

Текущее значение таймера TON сбрасывается, когда выключается разрешающий вход, тогда как текущее значение таймера TONR сохраняется, когда этот вход выключается.

Так же можно использовать таймер TONR для накопления времени, когда этот вход включается и выключается. Для стирания текущего значения TONR используется команда Сброс (R).

Таймеры TON и TONR продолжают счет после достижения предустановленного значения, они останавливают счет при достижении максимального значения, равного 32767.

Команда TOF 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 вход выключается, таймер ведет отсчет времени, пока истекшее время не достигнет предустановленного значения. Когда предустановленное время достигнуто, бит таймера сбрасывается, а отсчет текущего значения прекращается; однако, если вход включается снова, прежде чем TOF достигнет предустановленного значения, то бит таймера остается установленным.

Чтобы таймер TOF начал отсчет времени, к его разрешающему входу должен быть приложен падающий фронт.

Таймеры отсчитывают интервалы времени. Разрешающая способность (или база времени) таймера определяет промежуток времени на один интервал. Например, TON с разрешающей способностью 10 мс отсчитывает количество

10-миллисекундных интервалов, прошедших после активизации TON: отсчет 50 на 10-миллисекундном таймере представляет 500 мс.

Команды пересылки байта (MOVB), слова (MOVW), двойного слова (MOVD) (рис. 11) и вещественного числа (MOVR) пересылают значение из адреса IN в адрес OUT, не изменяя исходной величины.

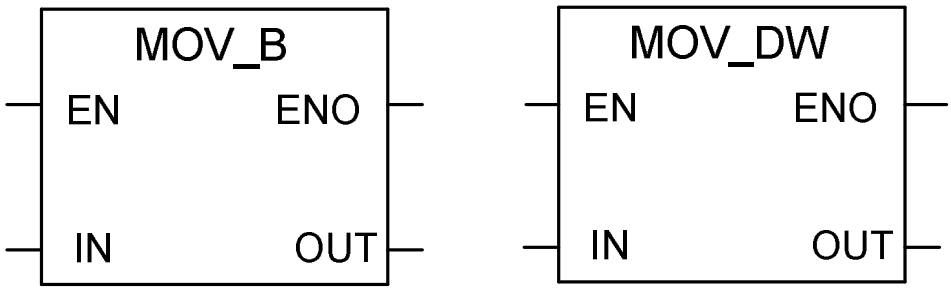


Рис. 11. Команда пересылки слова и двойного слова

Косвенная адресация использует указатель для доступа к данным в памяти. Указатели - это ячейки памяти, имеющие размер двойного слова, которые содержат адрес другой ячейки памяти. В качестве указателей можно использовать только ячейки памяти переменных и локальных данных.

Для создания указателя необходимо использовать команду «Переместить двойное слово». Эта команда передает адрес косвенно адресованной ячейки памяти в ячейку указателя. Правила работы с адресами и указателями аналогичны таковым в языке С: для получения адреса переменной используется оператор «&», для обращения по адресу, записанному в указателе – оператор «*».

Обращение к данным в областях памяти

В контроллере Siemens S7 200 имеется множество областей памяти. Ниже приведено краткое описание только тех областей, которые используются в программе.

1) Регистр входов образа процесса I.

В начале каждого цикла S7–200 опрашивает физические входы и записывает полученные значения в регистр входов образа процесса. К образу процесса можно обратиться в формате бита, байта, слова и двойного слова:

Бит: I [адрес байта] [адрес бита], например, I0.1.

Байт, слово или двойное слово: I [длина] [начальный адрес байта], например, IB4.

2) Регистр выходов образа процесса Q.

В конце цикла S7–200 копирует значения, хранящиеся в регистре выходов образа процесса, в физические выходы. К образу процесса можно обратиться в формате бита, байта, слова и двойного слова:

Бит: Q [адрес байта] [адрес бита], например Q1.1.

Байт, слово или двойное слово: Q [длина] [начальный адрес байта], например, QB5.

3) Область памяти переменных V

Память переменных можно использовать для хранения промежуточных результатов операций, выполняемых в программе. В памяти переменных, воз-

можно, хранить также другие данные, имеющие отношение к процессу или к решению вашей задачи автоматизации. К памяти переменных можно обратиться в формате бита, байта, слова и двойного слова:

Бит: V [адрес байта] [адрес бита], например, V10.2.

Байт, слово или двойное слово: V [длина] [начальный адрес байта] , например, VW100.

4) Область битовой памяти M.

Биты памяти (маркеры) можно использовать как управляющие реле для хранения промежуточных результатов операций или другой управляющей информации. К битам памяти можно обратиться в формате бита, байта, слова и двойного слова:

Бит: M [адрес байта] [адрес бита], например, M26.7.

Байт, слово или двойное слово: M [длина] [начальный адрес байта], например, MD20.

5) Специальные биты памяти SM.

Специальные биты памяти (SM) предоставляют средство для обмена данными между CPU и программой. Можно использовать эти биты для выбора и управления некоторыми специальными функциями CPU S7–200, например: бит, который устанавливается только в первом цикле; бит, который устанавливается и сбрасывается с фиксированной частотой, или бит, который указывает на состояние арифметической или иной команды. К SM-битам можно обращаться в формате бита, слова или двойного слова:

Бит: SM [адрес байта] [адрес бита], например, SM0.1.

Байт, слово или двойное слово: SM [длина] [начальный адрес байта], например, SMB86.

Программа работы.

Разработка программы управления

Сформулируем требования к проектируемой программе и наметим пути решения, которые удовлетворят этим требованиям.

Программа должна управлять приводами робота-манипулятора с целью отработки *пользовательской программы* перемещения рабочего органа робота.

Пользовательская программа будет состоять из набора шагов, каждый из которых задает положение рабочего органа по трем координатам, положение штока соленоида (выдвинут или втянут), а также выдержку времени после отработки всех приводов на данном шаге. Эта программа может выполняться однократно или циклически.

В памяти контроллера пользовательская программа может быть представлена набором фреймов. Определим размер фрейма.

Заданные координаты по осям должны будут сравниваться с текущими координатами, отсчитываемыми счетчиками. Выходное слово счетчика имеет размер 2 байта. Выдержка времени производится с помощью таймера, его выходное слово также имеет размер 2 байта. Для управления соленоидом потребуется всего 2 бита: один для фиксации команды втянуть шток, другой для фиксации команды выдвинуть шток. Кроме того, программа управления долж-

на каким-либо образом распознавать последний фрейм пользовательской программы. Для этого потребуется еще один бит. Объединим биты управления соленоидом и фиксации последнего фрейма в понятие «управляющее слово». Для того, чтобы сделать общий размер фрейма кратным двум и упростить тем самым «навигацию» по фреймам, определим размер управляющего слова равным двум байтам. Таким образом, получим следующую структуру фрейма:

Задание по координате X:	2 байта;
Задание по координате Y:	2 байта;
Задание по координате Z	2 байта;
Выдержка времени:	2 байта;
Управляющее слово:	2 байта;
Всего:	10 байт.

Пользовательская программа будет размещаться в области переменных V.

Проектируемая программа, очевидно, будет содержать регуляторы положения по трем координатам. Заданные значения координат для каждого шага хранятся во фреймах. Таким образом, адреса заданий будут постоянно меняться, что создает определенные трудности при разработке программы. Избежать их можно, отведя требуемый фиксированный объем памяти для текущих заданий. Назовем эту область памяти «областью текущих заданий». Ее размер будет равен размеру фрейма, т.е. 10 байт. Перед выполнением очередного шага программа управления будет копировать содержимое очередного фрейма в эту область. Поэтому регуляторы положения и механизм выдержки времени будут всегда обращаться по одним и тем же адресам.

Для перемещения по фреймам нам потребуется указатель. Указатель занимает в памяти 4 байта.

Разместим указатель и область текущих заданий в самом начале области переменных V. На это потребуется 14 байт. Следовательно, пользовательская программа будет начинаться с 14 байта (учитывая, что нумерация байтов начинается с нуля). Общая структура использования памяти области V приведена на рис. 12.

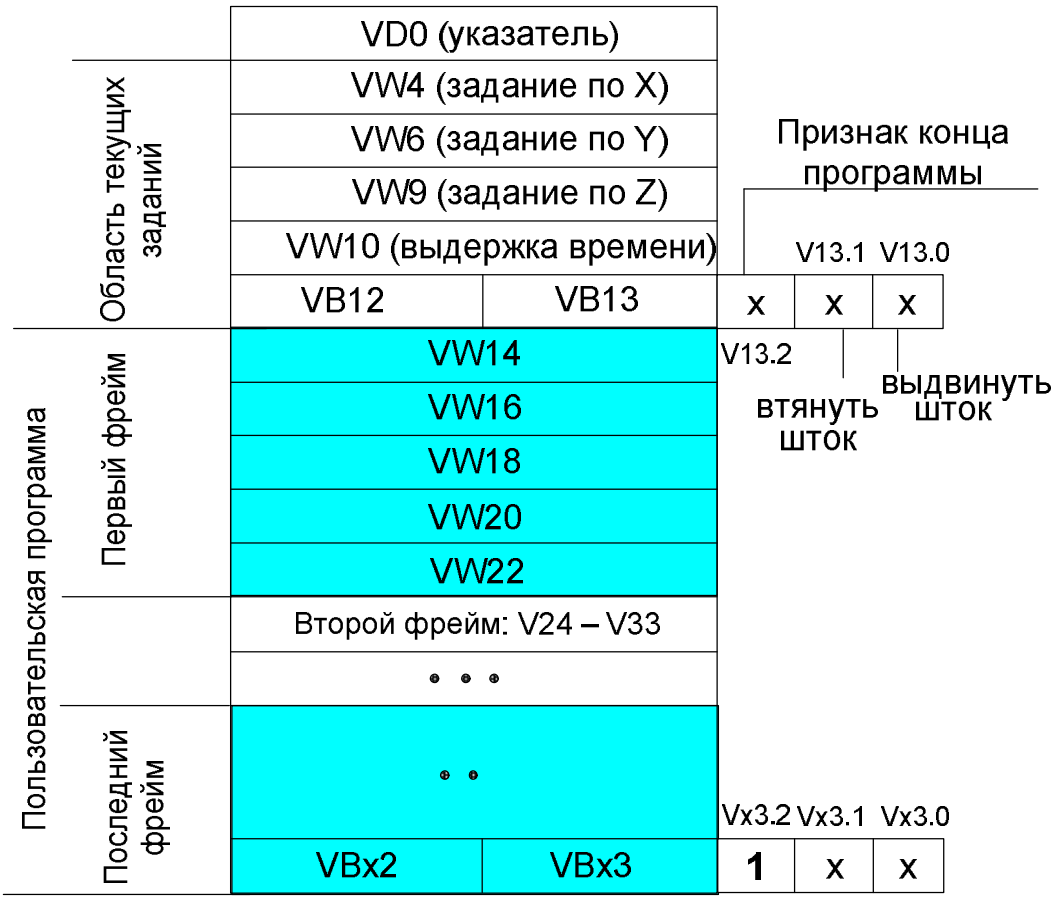


Рис. 12. Использование памяти области V

Далее определимся с органами управления и структурой программы.

Поскольку в роботе-манипуляторе по трем координатам используются датчики не абсолютного, а относительного перемещения, необходимо перед отработкой программы перевести робот-манипулятор в некоторое начальное (нулевое) положение, относительно которого и будет в дальнейшем осуществляться отчет перемещений. Пусть это будет следующее положение:

привод горизонтального перемещения – крайнее положение в направлении «назад»;

привод поворота – крайнее положение в направлении «против часовой стрелки»;

привод вертикального перемещения – крайнее положение в направлении «вниз».

Перевод всех приводов в начальное положение оформим в виде подпрограммы Init (инициализация). Вызов этой подпрограммы будет производиться пользователем путем включения выключателя, связанного с входом контроллера I1.2 и размещенного на стенде.

Другая подпрограмма – Work (работа) будет заниматься основной операцией – отработкой пользовательской программы. Ее вызов будет осуществляться путем включения выключателя, связанного с входом I1.3.

Если оба выключателя (I1.2 и I1.3) не включены, все приводы робота должны быть остановлены. За это будет отвечать подпрограмма Stop.

Пользователь должен иметь возможность переводить систему на циклическую отработку программы перемещений. Для этого будем использовать выключатель, связанный с входом I1.4.

Далее определим, какие дополнительные переменные нам потребуются, анализируя те состояния, которые нужно зафиксировать.

1. Привод вертикального перемещения оснащен встроенными в силовую цепь концевыми выключателями (см. рис. 4). Из схемы видно, что при достижении крайнего положения, например, в направлении «вниз», привод будет отключен непосредственно концевым выключателем SQ5. Одновременно получает питание катушка реле K9. Реле замыкает контакт во входной цепи контроллера и активирует вход I0.7 (см. рис. 5). Следуя своей логике, контроллер должен отключить привод. Для этого он обнуляет выход Q0.4 и катушка реле K5 теряет питание. Это приводит к тому, что обесточивается цепь концевого выключателя SQ5 и катушки реле K9. Следовательно, на вход контроллера I0.7 вновь подается сигнал логического нуля. Контроллер, «забывая» о том, что достигнуто конечное положение, вновь подает питание на реле K5. В результате будет наблюдаться так называемый «дребезг» контактов реле K5 и K9, что неминуемо выведет их из строя. Аналогичная ситуация имеет место при достижении верхнего конечного положения привода.

Очевидно, проблема состоит в том, что достижение крайнего положения по координате Z не запоминается «аппаратно», следовательно, его необходимо запомнить «программно». С этой целью будем использовать биты памяти маркеров M0.0 и M0.1.

Бит M0.0 будет устанавливаться при достижении приводом крайнего нижнего положения (срабатывании концевого выключателя SQ5 и реле K9) и сбрасываться при включении привода в направлении «вверх».

Бит M0.1 будет устанавливаться при достижении приводом крайнего верхнего положения (срабатывании концевого выключателя SQ6 и реле K10) и сбрасываться при включении привода в направлении «вниз».

2. Для выдвижения или втягивания штока соленоида требуется кратковременно (не более 1 сек.) подать на него напряжение. Длительное нахождение соленоида под напряжением недопустимо. Поэтому при выполнении тех шагов программы пользователя, на которых предусмотрена работа соленоида, программа управления должна знать, выполнена ли операция управления штоком или еще нет. Для фиксации факта отработки штоком задания будем использовать бит M0.2. Этот бит будет сбрасываться при переходе на очередной шаг и устанавливаться, если операция управления штоком на текущем шаге завершилась или таковой не было предусмотрено.

3. Для организации загрузки очередного фрейма программы пользователя в область текущих заданий будем использовать бит M0.3. Этот бит будет устанавливаться в единицу, если закончена выдержка времени на данном шаге и выполнены все необходимые подготовительные действия, связанные с перенастройкой указателя.

Программирование

Структура программы

Запустите Step 7 MicroWin. Создайте новый проект. Сохраните его под осмысленным именем.

Как определено выше, программа управления будет состоять из 4 частей:

1) основная программа;

2) подпрограмма инициализации Init;

3) подпрограмма работы Work;

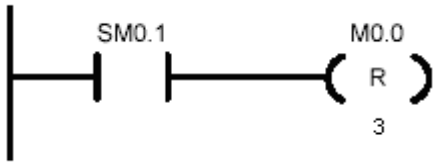
4) подпрограмма остановки Stop.

Переименуйте подпрограмму «SBR_0» в «Init». Удалите ненужную подпрограмму обработки прерываний «INT_0». Добавьте подпрограммы Work и Stop

Основная программа

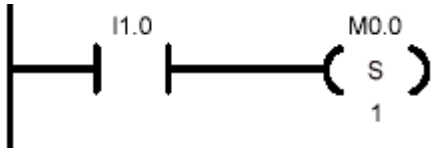
В основной программе будут осуществляться вызовы подпрограмм, а также выполняться действия общие для всех режимов работы.

На первом цикле работы контроллера сбрасываем биты M0.0, M0.1 и M0.2. Для определения того, является ли выполняемый цикл первым, анализируется бит специальной памяти SM0.1, который установлен только во время первого цикла:

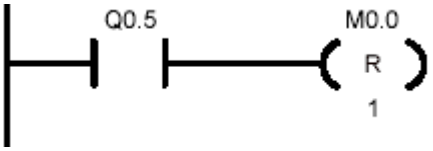


Далее осуществляется работа с битами M0.0 и M0.1, фиксирующими крайние положения привода вертикального перемещения. Запоминание крайних положений требуется осуществлять во всех режимах работы программы, поэтому и будем выполнять его здесь, в основной программе.

Если сработал концевой выключатель в крайнем нижнем положении (замкнут контакт реле K9 цепи входа I1.0), устанавливаем бит M0.0:

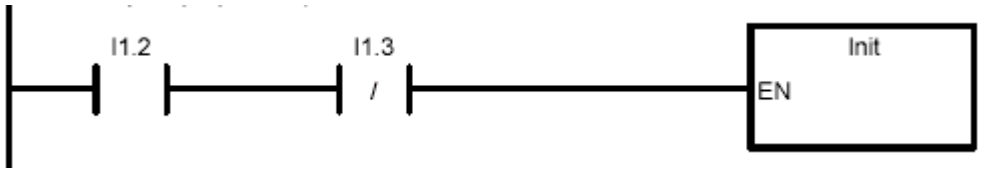


Если происходит движение вверх (активирован выход Q0.5), сбрасываем бит M0.0:



«Цепи» управления битом M0.1 постройте самостоятельно.

Далее в зависимости от положения выключателей, связанных с входами I1.2 и I1.3 вызываем подпрограмму Init, Work и Stop, не допуская при этом их одновременного вызова при одновременном включении выключателей. Вызов подпрограммы Init:



Вызов подпрограмм Work и Stop организуйте самостоятельно.

Основная программа закончена.

Подпрограмма Init

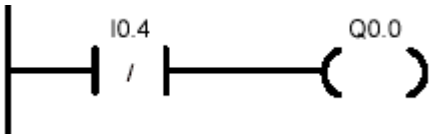
В подпрограмме Init будут выполняться следующие действия:

1) перевод всех приводов в начальное положение;

2) загрузка первого фрейма пользовательской программы в область текущих заданий;

3) сброс счетчиков, отсчитывающих текущие координаты приводов.

Рассмотрим перевод привода координаты X в начальное положение. Если не сработал концевой выключатель в направлении «назад», необходимо включить привод в направлении «назад»:

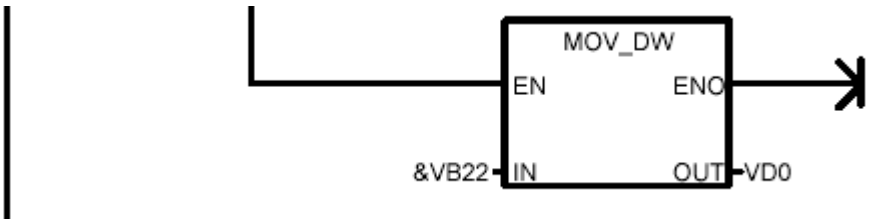


Перевод приводов координат Y и Z в начальное положение организуйте самостоятельно. При этом контролировать достижение приводом координаты Z крайнего нижнего положения следует, анализируя бит M0.0, а не вход I1.0.

Процедура загрузки первого фрейма в область текущих заданий (используется операция копирования слов):

– запись задания по координате X.

Продолжите здесь Network (запись заданий по Y,Z, выдержки времени, управляющего слова)



– закончим Network, записав в указатель VD0 адрес последнего слова первого фрейма. Передвижение указателя на второй фрейм будет осуществляться в режиме «работа»

В программе управления будут использоваться три счетчика C0, C1, C2 для подсчета импульсов, поступающих с датчиков на входы I0.0, I0.1, I02. Для сброса счетчика достаточно записать нуль в соответствующую переменную:

– сброс счетчика C0 (по координате X).

Закончите здесь Network сбросом счетчиков C1 и C2

Подпрограмма Init закончена. Ее можно протестировать отдельно. Для этого загрузите программу в контроллер, переведите его в режим Run и включите выключатель, связанный со входом контроллера I1.2.

Подпрограмма Work

В подпрограмме Work будут выполняться следующие действия:

1) контроль текущего положения приводов;

2) регулирование положения по координатам;

3) управление соленоидом с выдержкой времени;

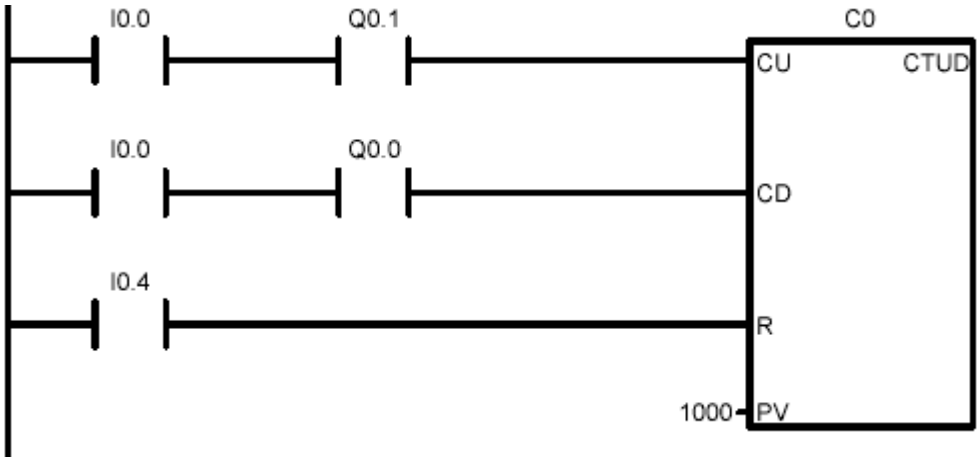
4) выдержка времени после отработки всех перемещений и работы соленоида;

5) загрузка очередного фрейма пользовательской программы в область текущих заданий и обнуление таймеров.

1. Контроль текущего положения приводов.

Для контроля положения по горизонтальной оси используется реверсивный счетчик CTUD C0. Значение счета увеличивается на единицу, если поступил импульс на вход I0.0 и привод двигался в направлении «вперед» (активирован выход Q0.1). Значение счета уменьшается на единицу, если поступил импульс на вход I0.0 и привод двигался в направлении «назад» (активирован вы-

ход Q0.0). Счетчик сбрасывается в нуль, если достигнуто крайнее положение в направлении «назад» (активирован вход I0.4):



Предустановленное значение PV устанавливаем достаточно большим, по крайней мере, большим максимального числа импульсов датчика (на полное перемещение).

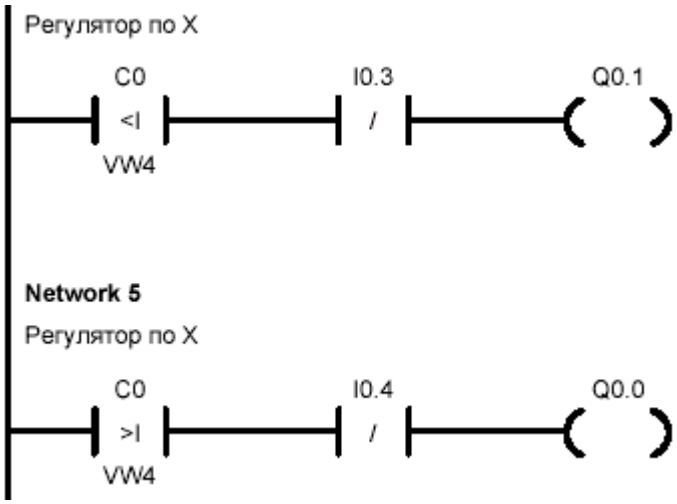
Счет импульсов по координатам Y и Z организуйте самостоятельно.

2. Регулирование координат.

В нашем случае применяется простейший релейный регулятор положения.

Если текущее значение координаты X, отсчитываемое счетчиком C0, меньше заданного, сохраненного в области текущих заданий в переменной VW4 и, кроме того, не достигнуто крайнее положение в направлении «вперед», включить привод в направлении «вперед».

Если текущее значение координаты X больше заданного и, кроме того, не достигнуто крайнее положение в направлении «назад», включить привод в направлении «назад».

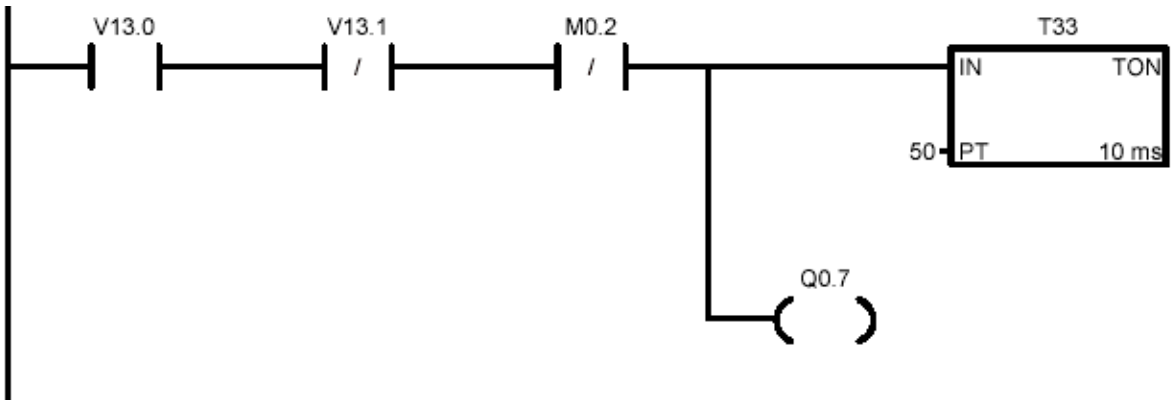


Контроль достижения крайних положений защищает программу от ввода недопустимых заданий.

Регулирование координат Y и Z организуйте самостоятельно.

3. Управление соленоидом с выдержкой времени.

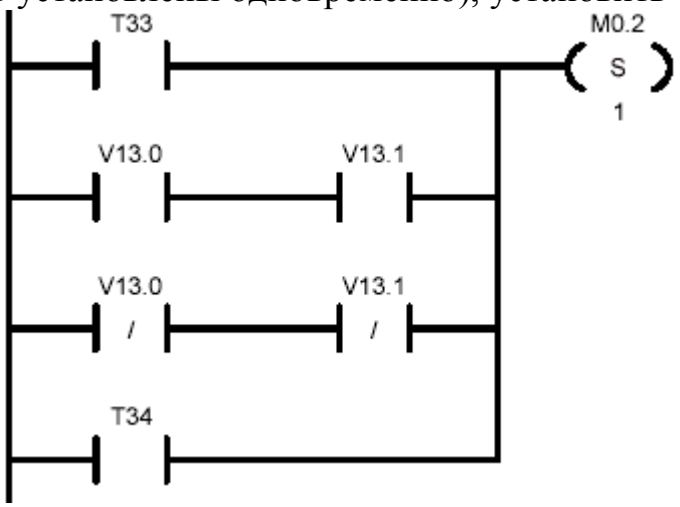
Если на данном шаге требуется выдвижение штока (V13.0) и не требуется его втягивание (не установлен бит V13.1, например, по ошибке) и, кроме того, соленоид еще не отработал (бит M0.2 не установлен), запустить выдержку времени 0,5 сек. и подать напряжение на соленоид для выдвижения штока (активировать выход Q0.7):



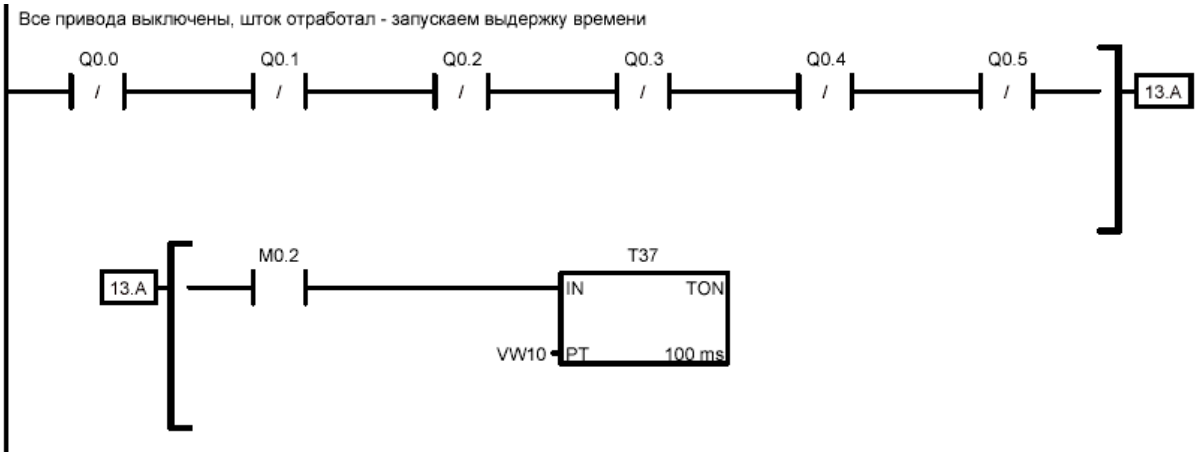
Network для управления втягиванием штока организуйте самостоятельно. При этом используйте таймер T34.

Управление битом M0.2, запоминающим факт отработки задания соленоидом, приведено ниже.

Если закончена выдержка времени на подачу напряжения на соленоид для выдвижения (T33) **ИЛИ** для втягивания его штока (T34) **ИЛИ** соленоид не работает на данном шаге (биты V13.0 и V13.1 обнулены) **ИЛИ** по ошибке дано задание на одновременное выдвижение и втягивание штока (биты V13.0 и V13.1 установлены одновременно), установить бит M0.2 (соленоид отработал):



4. Выдержка времени после отработки всех перемещений и работы соленоида. Заданная выдержка времени хранится в области текущих зажаний в переменной VW10:



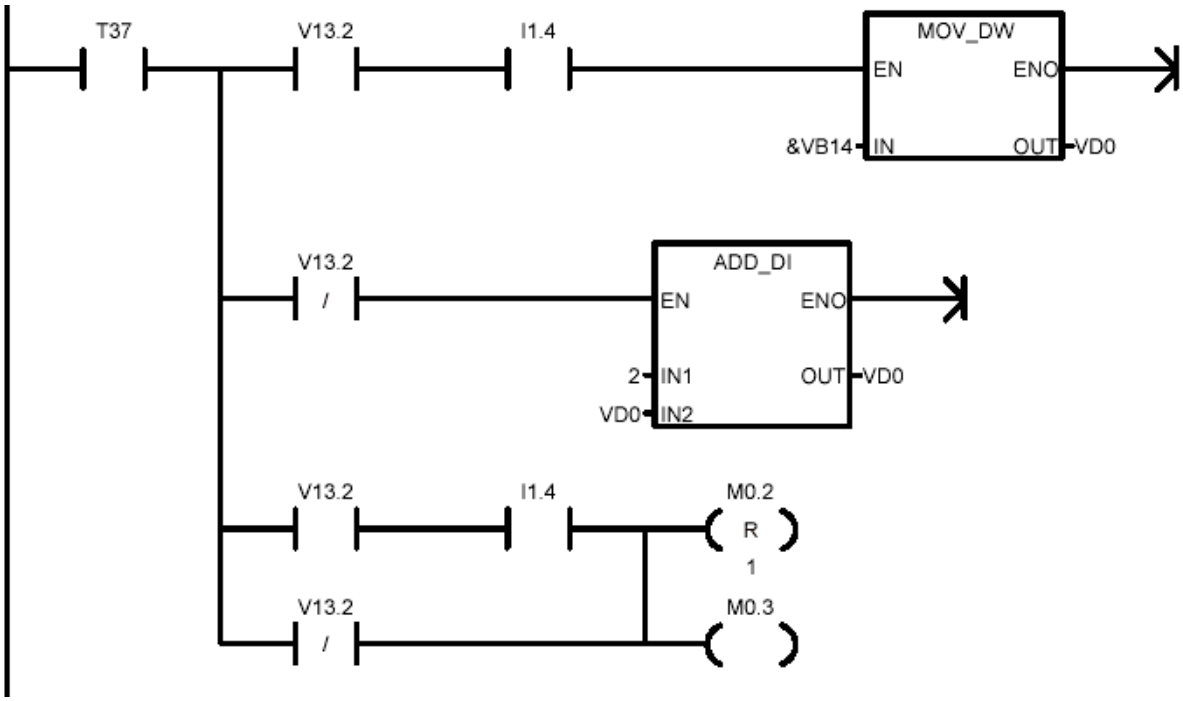
5. Загрузка очередного фрейма пользовательской программы в область текущих заданий и обнуление таймеров.

Если закончилась выдержка времени, то

если отрабатываемый фрейм – последний (V13.2 установлен) **И** программа работает в циклическом режиме (выключатель I1.4 включен), – записать в указатель VD0 адрес первого слова первого фрейма пользовательской программы, подготавливая его выполнение;

если отрабатываемый фрейм не последний (V13.2 не установлен), – увеличить указатель VD0 на два, перейдя на следующий фрейм;

если отрабатываемый фрейм последний (V13.2 установлен) **И** программа работает в циклическом режиме (выключатель I1.4 включен)} **ИЛИ** отрабатываемый фрейм не последний (V13.2 не установлен), – сбросить бит M0.2 (соленоид <u>не отработал</u>) **И** установить бит M0.3, разрешающий загрузку нового фрейма (если не установить этот бит, новый фрейм не загрузится и программа остановится):



Загрузка очередного фрейма:

– то, на что указывает указатель, занести в переменную VW4 (задание по X),

– увеличить указатель на 2.

Продолжите здесь Network (запись заданий по Y,Z, выдержки времени, управляющего слова)

Далее – обнуление таймеров:

– обнуление таймера T33.

Закончите здесь Network, обнулив таймеры T34 и T37 и сбросив бит M0.3.

Подпрограмма Work готова.

Подпрограмма Stop

Подпрограмма Stop должна выключать все электроприводы робота-манипулятора. Напишите ее самостоятельно.

Запуск и тестирование программы

Теперь требуется ввести пользовательскую программу и протестировать систему.

Для этого зайдите в раздел «Блок данных» и введите следующее:

```
VW14 20, 20, 5, 30, 0
VW24 20, 20, 5, 30, 1
```

```
VW34 20, 20, 5, 30, 2
VW44 0, 0, 0, 30, 4
```

Здесь записана простейшая пользовательская программа, состоящая из четырех фреймов:

1) перевод робота в положение X = Y = 20, Z = 5, выдержка времени 3 сек, соленоид не работает;

2) срабатывание соленоида (выдвижение штока) в той же позиции, выдержка времени 3 сек;

3) срабатывание соленоида (втягивание штока) в той же позиции, выдержка времени 3 сек;

4) перевод в начало координат, остановка программы, соленоид не работает.

Загрузите программу в контроллер.

Переведите контроллер в режим «Run».

Запустите подпрограмму инициализации. Дождитесь остановки приводов.

Запустите подпрограмму Work в режиме однократного исполнения.

Переведите систему в режим циклического исполнения (включите выключатель, связанный с входом I1.4 контроллера). Наблюдайте циклическое исполнение пользовательской программы.

Содержание отчета

1. Принципиальная электрическая схема соединений цепей управления контроллера, робота-манипулятора и органов управления. На схеме должны быть приведены только используемые входы и выходы контроллера и робота.

2. Программа контроллера в представлениях LAD, STL и FBD.

Контрольные вопросы

1. Опишите кинематическую схему робота-манипулятора.

2. Укажите расположение концевых выключателей приводов робота и оптических датчиков по координатам.

3. Опишите схему силовых цепей робота.

4. Опишите схему цепей коммутации электроприводов робота.

5. Опишите схему подключения внешних цепей контроллера.

6. Опишите команды «Реверсивный счетчик» и «Таймер», а также команды пересылки, используемые в программе управления.

7. Какие области памяти контроллера использовались в программе?

8. Опишите принципы обращения к данным в областях памяти.

8. Опишите структуру «пользовательской» программы перемещения рабочего органа робота.

9. Какова особенность управления приводом вертикального перемещения робота?

10. Какова особенность управления соленоидом?

11. Опишите структуру программы управления и назначения органов управления.

12. Опишите основную программу.
13. Опишите подпрограмму Init.
14. Опишите подпрограмму Work.

8. Разработка системы обучения робота манипулятора

Цель работы: разработка и реализация программной системы обучения робота-манипулятора.

Теоретические сведения

В предыдущей лабораторной работе разработана система воспроизведения программы движения робота-манипулятора. Программа задается набором фреймов, каждый из которых описывает желаемое положение по трем осям, выдержку времени по окончанию движения и информацию о положении штока соленоида (см. лаб. работу №7, рис. 12). Фреймы занимают по 10 байт и размещаются в области переменных V, начиная с адреса V14. При отработке программы содержимое очередного фрейма загружается в «Область текущих значений» (V4 – V13), откуда текущие задания считываются фрагментами подпрограммы «Work», отвечающими за регулирование положений, выдержку времени, приведение в движение штока. Загрузка фреймов в «Область текущих заданий» осуществляется с помощью указателя VD0 (четыре байта). Последний фрейм имеет специальную «метку» – установленный в единицу второй бит последнего байта Vx3.2. При ее обнаружении в зависимости от режима (единичный/циклический) программа либо прекращает свое выполнение, либо начинает выполняться снова, начиная с первого фрейма. Выбор режима осуществляется с помощью переключателя, связанного с входом контроллера I1.4.

Основной недостаток разработанной системы состоит в сложности ее программирования: пользователь вручную должен сформировать блок данных программы контроллера, записав численные данные о положении приводов (число импульсов датчиков), выдержке времени и т.д. по каждому фрейму.

Более простой и естественной представляется следующий подход к программированию движения робота.

Пользователь переводит робота в начальное положение вызовом подпрограммы «Init». Далее с помощью органов ручного управления робот переводится в первое заданное положение и нажимает кнопку «Память» («первое нажатие»). Программа запускает специальный таймер и записывает текущие значения координат в переменные VW14, VW16, VW18, а также биты V23.0 и V23.1 (информация о движении штока). По окончанию требуемой выдержки времени в данной позиции пользователь вновь нажимает кнопку «Память» («второе нажатие»). Программа считывает значение таймера и заносит его в переменную VW20. Далее аналогично заполняется второй, третий и т.д. фреймы.

Определенную проблему представляет определение последнего фрейма программы. Для ее решения предлагается следующее. После второго нажатия кнопки «Память» программа контроллера в течение заданного короткого промежутка времени, например, 0,5 сек., ожидает «третье нажатие». Если оно происходит, программа воспринимает это как команду «Последний фрейм» и устанавливает в единицу соответствующий бит (Vx3.2).

После записи последнего фрейма пользователь вновь переводит робота в начальное положение и запускает программу «Work».

Для реализации описанного подхода предлагается модифицировать программу контроллера, созданную при выполнении лаб. работы №7. Для записи пользовательской программы необходимо разработать специальную подпрограмму, пусть она будет называться «Record». В основной программе следует предусмотреть ее вызов, например, при активации входа I1.5 с помощью соответствующего выключателя на стенде.

Структура подпрограммы «Record» в основном будет повторять структуру подпрограммы «Work». Однако при ее разработке следует обратить внимание на ряд обстоятельств:

1) поскольку управление приводами будет осуществляться в ручном режиме пользователем, контроллер для определения направлений счета реверсивных счетчиков по координатам должен иметь средства распознавания направлений включений приводов. Кроме того, контроллер должен «уметь» фиксировать факты втягивания и выдвижения штока соленоида, а также факт нажатия кнопки «Память». Для этих целей программой будут анализироваться входы I2.4 – I2.5, к которым должны быть подключены штекеры 1А – 5А, П, связанные с дополнительными контактами реле и кнопкой, см. табл. 1

Таблица 1. Дополнительные входы контроллера, используемые подпрограммой «Record»

Вход контроллера	Штекер	Назначение входа
I2.0	1А	Включен привод по X вперед
I2.1	2А	Включен привод по Y почасовой
I2.2	3А	Включен привод по Z вверх
I2.3	4А	Шток «втягивается» (сол. под напряжением)
I2.4	5А	Шток «выдвигается» (сол. под напряжением)
I2.5	П	Кнопка «Память»

2) для запоминания фактов срабатывания соленоида можно использовать биты памяти области M, а для подсчета числа нажатий кнопки «Память» – специальный счетчик. Биты и счетчик необходимо сбрасывать при переходе к следующему фрейму.

Задание

1. Разработать и собрать принципиальную электрическую схему соединений для реализации подсистемы обучения робота-манипулятора.

2. Модифицировать программу управления контроллера, разработав подпрограмму «Record», а также добавив средства ее вызова в главную программу.

3. Апробировать систему и продемонстрировать ее работу преподавателю.

Содержание отчета

1. Дополнение принципиальной электрической схемы соединений цепей управления контроллера, робота-манипулятора и органов управления, разработанной при выполнении лаб. работы №7.

2. Программа контроллера в представлениях LAD, STL и FBD (добавленные и/или измененные компоненты).

Контрольные вопросы

1. Опишите процедуру обучения робота.

2. Опишите дополнительные цепи контроллера, используемые в работе.

3. Каким образом контроллер определяет направление движения приводов по осям в процедуре обучения?

4. Опишите логику обработки программой нажатия кнопки «Память» и реализацию этой логики.

9. Разработка монитора реального времени для управления роботом манипулятором

Цель работы: разработка монитора реального времени для управления роботом-манипулятором.

Теоретические сведения

Необходимость разработки системы дистанционного компьютерного управления на практике может быть связана, например, с вредными и опасными условиями работы робота-манипулятора (радиоактивное излучение, загазованность, опасность взрыва). В этом случае для минимизации длины проводных соединений управляющий контроллер устанавливается непосредственно на роботе, а компьютер выносится из опасной зоны и соединяется с контроллером по сетевому интерфейсу, например, RS-485. В случае необходимости робот может быть дополнительно оснащен видеокамерой, изображение с которой будет транслироваться на монитор компьютера.

Программное обеспечение компьютера, в нашем случае монитор реального времени (МРВ) Trace Mode, должно решать следующие задачи:

1) контроль положения рабочего органа робота-манипулятора (значения счетчиков по координатам, состояние концевых выключателей, положение штока соленоида) во всех режимах работы;

2) прямое управление приводами робота манипулятора в режиме «Дистанционное управление»;

3) запуск подпрограмм инициализации и работы.

Дополнительно могут быть реализованы следующие возможности:

1) графическая визуализация работы робота на «координатной сетке»;

2) просмотр и редактирование фреймов программы перемещения;

3) обучение робота командами монитора реального времени, и др.

Теоретические сведения о взаимодействии Trace Mode с контроллером Siemens S7-200 приведены в лабораторной работе №5. Описаны специальный драйвер и его настройка, а также способы обращения к областям памяти контроллера.

В данной работе информационный обмен между монитором реального времени и контроллером будет включать в себя передачу как «аналоговых» сигналов (например, значений счетчиков), так и дискретных (включение приводов, опрос конечных выключателей). Ниже описана технология организации дискретных каналов на примере каналов опроса области входов I контроллера.

Пусть, например, требуется опрашивать входы I0.0 – I1.7. Создадим объект DInput, в нем – канал DI1_In с настройками представленными, на рис. 1 (смещение – 0000, тип – DiscreteInputs(WORD)).

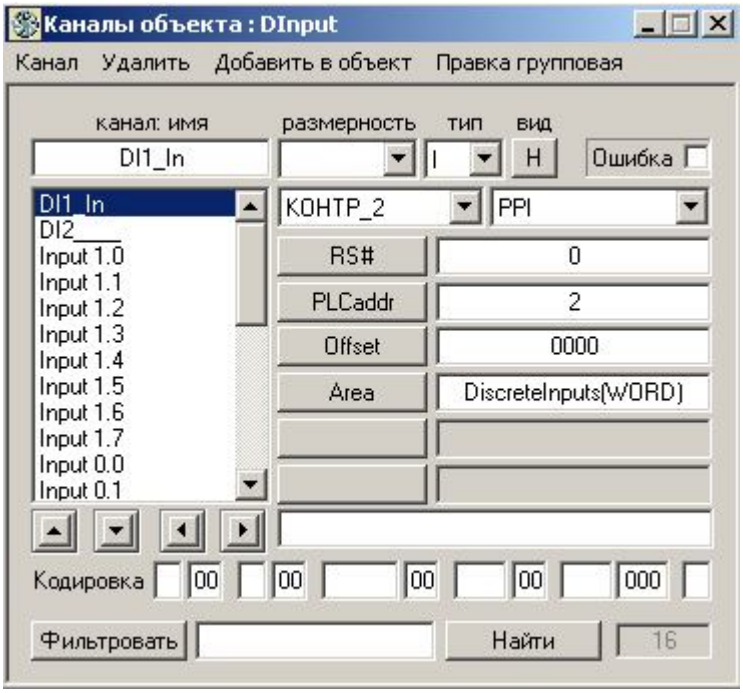


Рис. 1. Канал DI1_In

Данный канал вмещает в себя 16 простых битовых каналов. Для того, чтобы разбить его на простые, необходимо в реквизитах канала поставить число бит – 16 (рис. 2) и разбить его по битам («Канал» → «Разбить побитно»).

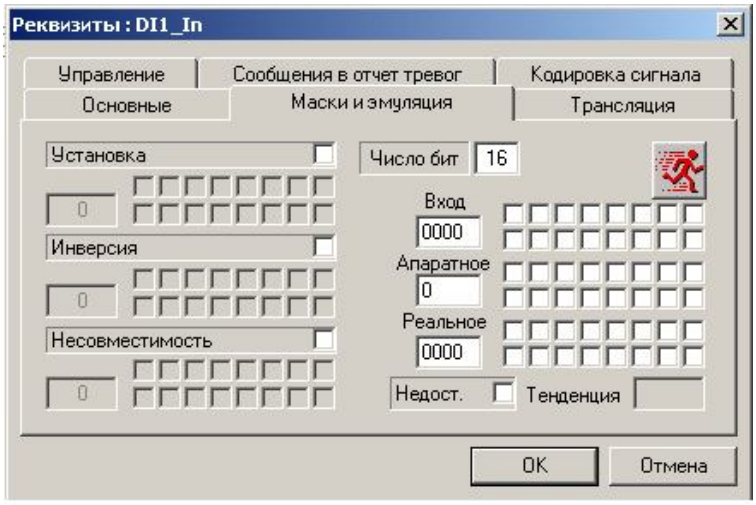


Рис. 2. Реквизиты канала DI1_In

В итоге получаем список из 16 простых каналов. Далее переименовываем их соответствующим образом (Input 0.0 - Input 1.7). Следует обратить внимание на порядок следования каналов: канал Input 0.0 (рис. 3) будет соответствует девятому биту канала DI1_In (ATTR = Bit 9), а канал Input 1.0 (рис. 4) – первому биту (ATTR = Bit 1).

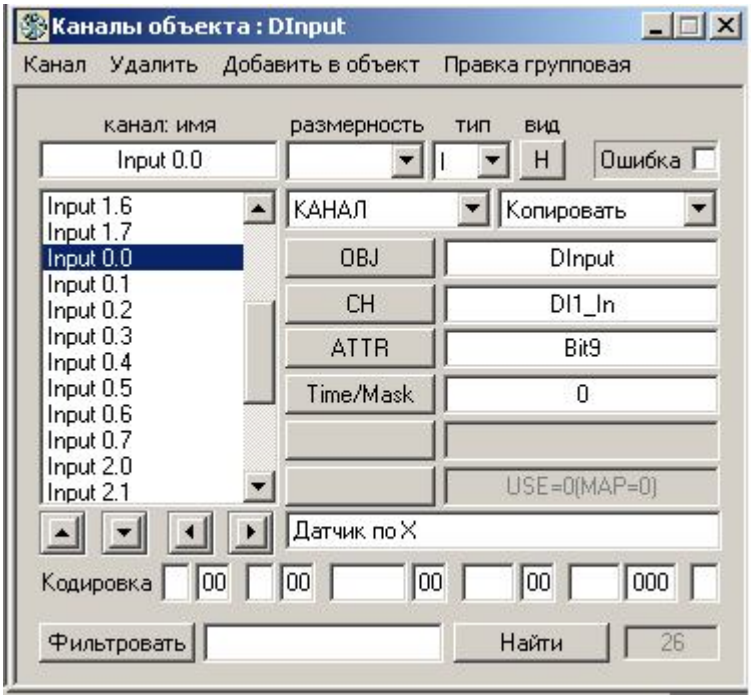


Рис. 3. Канал Input 0.0

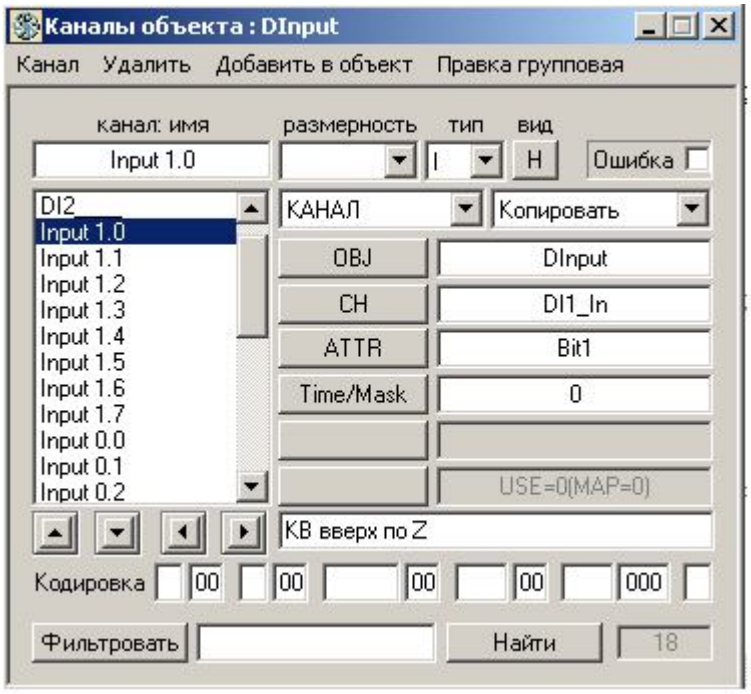


Рис. 4. Канал Input 1.0

Аналогично организуются каналы обмена с областями памяти контроллера Q и M. Обмен с областью Q необходим для включения приводов. Область M рекомендуется использовать для передачи команд, в частности запуска подпрограмм. Кроме того, биты этой области будут опрашиваться для получения информации о состоянии программы (например, о режиме управления) или самого робота (например, о положении штока соленоида).

Задание

1. Разработать монитор реального времени Trace Mode для управления роботом-манипулятором.

2. Модифицировать программу управления контроллера, обеспечив возможность взаимодействия с MPB.

3. Апробировать систему и продемонстрировать ее работу преподавателю.

Содержание отчета

1. База каналов и экранные формы монитора реального времени.

2. Программа контроллера в представлениях LAD, STL и FBD (добавленные и/или измененные компоненты).

Контрольные вопросы

1. Какие преимущества дает использование компьютерного управления роботом-манипулятором перед ранее разработанной системой?

2. Опишите каналы MPB, отвечающие за контроль текущего положение робота и состояния его рабочего органа.

3. Опишите каналы MPB, отвечающие за прямое управление приводами робота;

4. Опишите каналы MPB, отвечающие за запуск подпрограмм инициализации и работы.

5. Опишите изменения, сделанные в программе управления роботом для организации взаимодействия со SCADA-системой.

ПРИЛОЖЕНИЕ. Краткое техническое описание приборов и устройств лабораторных стендов

1. CPU Siemens S7-200

Серия S7-200 – это ряд микропрограммируемых логических контроллеров (микроконтроллеров), которые могут управлять разнообразными прикладными системами автоматизации. Компактная конструкция, низкая стоимость и мощная система команд делают контроллеры S7–200 идеальным средством решения для управления малыми приложениями. Большое разнообразие моделей S7–200 и инструментальные средства программирования на основе Windows обеспечивают необходимую гибкость при решении задач автоматизации.

Семейство программируемых логических микроконтроллеров (микро-ПЛК) S7-200 может управлять широким спектром устройств для решения задач автоматизации. S7-200 контролирует входы и изменяет выходы под управлением программы пользователя, которая может содержать булевы логические операции, функции счета и времени, сложные математические операции и операции по обмену данными с другими интеллектуальными устройствами. Благодаря компактной конструкции, гибкой конфигурации и мощному набору команд S7-200 в высшей степени пригоден для решения широкого спектра прикладных задач управления.

В табл. 1 представлены технические данные CPU 226 с релейными выходами, который входит в состав стенда.

Таблица 1. Технические данные CPU 226

Параметр	Значение
Память	
Размер программы пользователя с редактированием в режиме RUN без редактирования в режиме RUN	 16384 байт 24576 байт
Данные пользователя	10240 байт
Буферизация (мощный конденсатор) (возможна батарейка)	Тип. 100 час. (мин. 70 час. при 40°C) Тип. 200 дней
Входы/выходы	
Дискретные входы и выходы	24 входа/16 выходов
Аналоговые входы и выходы	нет
Цифровые входы и выходы (образ процесса)	256 (128 входов/128 выходов)
Аналоговые входы и выходы (образ процесса)	64 (32 входа и 32 выхода)
Макс. количество модулей расширения	7 модулей
Макс. количество интеллектуальных модулей	7 модулей
Входы для регистрации импульсов	24
Скоростные счетчики	Всего 6 счетчиков

Параметр	Значение
однофазные двухфазные	6 при 30 кГц 4 при 20 кГц
Импульсные выходы	нет
Общие данные	
Таймеры	Всего таймеров 256; 4 таймера (1 мс); 16 таймеров (10 мс); 236 таймеров (100 мс)
Счетчики	256 (с буферизацией от конденсатора большой емкости или батарейки)
Специальная память	
Биты внутренней памяти. Сохраняются при потере питания	256 (с буферизацией от конденсатора большой емкости или батарейки) 112 (сохраняются в ЭСППЗУ)
Прерывания, управляемые временем	2 с разрешением 1 мс
Прерывания по фронту сигнала	4 по нарастающему и/или 4 по убывающему фронту
Аналоговый потенциометр	2 с разрешением 8 битов
Скорость выполнения булевых операций	0,22 мкс на команду
Часы реального времени	Встроенные
Дополнительные съемные модули	Память и батарейка
Встроенные средства для обмена данными	
Порты (с ограничением мощности)	2 порта RS-485
Скорости передачи PPI, DP/T	9,6; 19,2; 187,5 кБод
Скорости передачи для свободно программируемого обмена данными	от 1,2 кБод 115,2 кБод
Макс. длина кабеля на сегмент	С гальванически развязанным повторителем: 1000 м до 187,5 кБод, 1200 м до 38,4 кБод. Без повторителя без гальванической развязкой: 50 м
Макс. количество станций	32 на сегмент, 126 на сеть
Макс. количество master-устройств	32
Двухточечное соединение (режим master-устройства PPI)	Да (NETR/NETW)
Соединения MPI	Всего 4, 2 зарезервированы (1 для устройства программирования и 1 для панели оператора)

В таблице 2 представлены технические данные дискретных входов CPU

Таблица 2. Технические данные дискретных входов CPU

Параметр	Значение
Тип	Обычная полярность/обратная полярность (1EC тип 1 -обычная полярность)
Номинальное напряжение	Тип. 24 В пост, тока при 4 мА
Макс. допустимое длительное напряжение	30 В пост, тока
Бросок напряжения	35 В пост, тока в течение 0,5 с
Логика 1 (мин.)	15 В пост, тока при 2,5 мА
Логика 0 (макс.)	5 В пост, тока при 1 мА
Входная задержка	Настраивается (от 0,2 до 12,8 мс)
Присоединение 2-проводного датчика близости (Bero) Допустимый ток утечки (макс.)	 1мА
Электрическая развязка (полевых устройств с логикой) Оптическая (гальваническая) Потенциально развязанные группы	Да 500 В перем. тока в течение 1 минуты См. схему соединений
Одновременно включенные входы	Все
Длина кабеля (макс.) экранированный не экранированный	 500 м для обычных входов, 50 м для входов скоростных счетчиков 300 м для обычных входов

В табл. 3 представлены технические данные дискретных выходов CPU.

Таблица 3. Технические данные дискретных выходов CPU

Параметр	Значение
Тип	Слаботочный контакт
Номинальное напряжение	24 В пост. тока или 250 В переменного тока
Диапазон напряжений	от 5 до 30 В пост. тока или от 5 до 250 В перем. тока
Бросок тока (макс.)	5 А в течение 4 с при относительной длительности 10%
Номинальный ток на один выход (макс.)	2 А

Параметр	Значение
Номинальный ток на провод (макс.)	10 А
Частота следования импульсов (макс.)	1Гц
Длина кабеля (макс.) экранированный не экранированный	 500м 150м

CPU S7-200 имеют в своем распоряжении источник питания датчиков, который поставляет 24 В постоянного тока для входов, для питания катушек реле на модулях расширения и других потребителей. Если потребности в мощности превышают возможности источника питания датчиков, то необходимо подключить к системе внешний источник питания 24 В постоянного тока.

2. Приводные механизмы робота-манипулятора

Приводной электромеханический модуль для вращательной кинематической пары робота разработан на базе серийного электромеханизма МЗК-2, для поступательных пар – на базе электромеханизмов МП-100 и МЗК-3.

Основные технические параметры электромеханизмов (скорость вращения выходного вала, развиваемый момент, весогабаритные характеристики) наиболее полно отвечают требованиям, предъявляемым к приводным устройствам манипуляторов.

Во многих случаях при выполнении технологических операций не требуется значительных скоростей, а предъявляются повышенные требования к точности манипулирования и надежности конструкции. Условия электробезопасности также играют не последнюю роль, что было учтено при выборе для всех электрических и питающих напряжений не выше 24 В.

Параметры используемых электроприводов помещены в таблицах 1, 2 и 3.

Таблица 1. Параметры МЗК-2

Параметр	Значение параметра
Напряжение питания, В	27
Ток, А	3
Мощность, Вт	10
Количество оборотов, об/мин	12500
Вес, кг	2,5
Выходной момент, Нм	27
Угол поворота исполнительной оси, град	90

Таблица 2. Параметры МЗК-3

Параметр	Значение параметра
Напряжение питания, В	27
Ток, А	3
Мощность, Вт	10
Количество оборотов, об/мин	10000
Вес, кг	3

Таблица 3. Параметры МП-100

Параметр	Значение параметра
Напряжение питания, В	27
Ток, А	2,7
Мощность, Вт	8
Количество оборотов, об/мин	274,6
Вес, кг	2

Используемые двигатели являются двигателями постоянного тока с последовательным возбуждением.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ASC 300. Руководство пользователя. Преобразователи частоты ACS 300 для регулирования скорости вращения асинхронных двигателей с короткозамкнутым ротором 0,55…11 кВт. – ABB, 1996 – 62 с.

2. SIEMENS. SIMATIC. Программируемый контроллер S7-200. Системное руководство. – SIEMENS, Издание 06/2004 – 514 с.

3. TRACE MODE v. 5.12. Справочная система. – AdAstra Research Group, 2003.

ДЛЯ ЗАМЕТОК

Андрей Николаевич Рыбалев
зав. кафедрой АПП и Э АмГУ, зав. кафедрой

Программируемые логические контроллеры и аппаратура управления: лабораторный практикум. Часть 2. Siemens S7-200.

Учебное пособие

Изд-во АмГУ. Подписано к печати ??.??.2009. Формат 60х84/16. Усл. печ. ?,?? Тираж 100. Заказ ???.