Федеральное агентство по образованию РФ
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ГОУВПО «АмГУ»)

УТВЕРЖДАЮ
Зав. кафедрой ИУС
_____А.В.Бушманов
«_____»_____

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС

по дисциплине «Микропроцессорные средства»
для студентов специальности 230102 «Автоматизированные системы обработки информации и управления»

Составитель: ассистент кафедры ИУС Дрюков А.А.

Факультет математики и информатики
Кафедра информационных и управляющих систем

2008

Печатается по решению
редакционно-издательского совета
факультета математики и информатики
Амурского государственного
университета

А.А. Дрюков

Учебно-методический комплекс по дисциплине «Микропроцессорные средства». 230102 «Автоматизированные системы обработки информации и управления» очной формы обучения.- Благовещенск: Амурский гос. ун-т, 2008.

Пособие содержит рабочую программу, курс лекций, методические рекомендации по проведению и выполнению лабораторных работ. Составлено в соответствии с требования государственного образовательного стандарта.

I. РАБОЧАЯ ПРОГРАММА

Курс 4 Семестр 7

Лекции 30 (час.) Зачет 7 семестр

Лабораторные работы 30 (час.)

Самостоятельная работа 25 (час.)

Всего часов 85 час.

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

1.1. Цели и задачи дисциплины

Цель курса - ознакомить студентов с возможностями и областями применения микропроцессорных средств; архитектуры микропроцессорных систем (МПС); организации подсистем обработки, управления, памяти и ввода-вывода; основных задач проектирования, тестирования и отладки МПС.

В результате изучения дисциплины студенты должны:

1) знать области применения МПС (включая однокристальные микроЭВМ, контроллеры и мультимикропроцессорные системы) и современные тенденции развития микропроцессорной техники, варианты построения архитектуры, подсистем обработки, управления, памяти и ввода-вывода МПС;

2) уметь разрабатывать программы на языке ассемблера микроЭВМ и пользоваться методами и современными средствами оценки, анализа и выбора состава и конфигурации микропроцессорных средств;

3) иметь представление о задачах системного, алгоритмического, структурного и логического проектирования МПС, о методах обеспечения надежности программных и аппаратных средств, включая методы тестирования и отладки МПС.

При изучении дисциплины "Микропроцессорные средства" используются знания, полученные в дисциплинах "Электротехника", "Схемотехника ЭВМ", "Организация ЭВМ и систем", "Системное программное обеспечение" и "Моделирование".

СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

2.1 Наименование тем, их содержание, объем в лекционных часах

Тема 1 – Программируемые логические контроллеры – 16 часов.

Понятие программируемого логического контроллера. Модульная архитектура ПЛК. Технические характеристики контроллеров. Критерии выбора контроллера для конкретного объекта. Промышленные сети. Типы архитектур АСУ. Примеры программируемых контроллеров. Контроллеры для малой автоматизации. Типичные примеры автоматизации промышленных объектов с помощью ПЛК. Семейство контроллеров SIMATIC S7-200. Аппаратура SIMATIC S7-200. Центральные процессоры S7-200. Модули SIMATIC S7-200. Модули человеко-машинного интерфейса SIMATIC HMI. Программное обеспечение для SIMATIC S7-200 STEP 7 Micro/WIN.

Тема 2 – Схемотехника блоков питания – 2 часа.

Схема блока питания с трансформаторным входом. Схема блока питания с бестрансформаторным входом.

Тема 3 – Цифро-аналоговые преобразователи и аналого-цифровые преобразователи – 4 часа.

ЦАП. Основные параметры ЦАП. Схема простейшего ЦАП. ЦАП с резистивной матрицей R-2R. АЦП. АЦП прямого преобразования. АЦП последовательного счета.

Тема 4 – Структура МПС – 4 часа.

Общая структура МПС. Понятие интерфейса в МПС. Примеры. Аппаратный интерфейс. Программный интерфейс. Мультиплексирование многоразрядной шины. Дешифратор адреса в МПС. Основные функции. Пример (схема).

Тема 2 – Передача данных в промышленных сетях – 4 часа.

Способы передачи информации в МПС. Асинхронная передача данных. Синхронная передача данных. Интерфейс RS485. Основные характеристики. Преимущества и недостатки. Интерфейс RS232. Основные характеристики. Преимущества и недостатки.

2.2 Лабораторные занятия, их наименование и объем в часах

Среда разработки приложений реального времени

1. Основы программирования контроллера «Мега» – 2 часа.
2. Программирование контроллера «Мега» – 2 часа.
3. Программа управления семисегментным индикатором – 2 часа.
4. Программа для системы сигнализации – 2 часа.
5. Программа управления шаговым двигателем – 2 часа.
6. Основы работы с контроллером SIMATIC S7-200.
7. Работа с памятью в контроллере SIMATIC S7-200.

2.3 Вопросы для самостоятельного изучения

1. Применение МПС в управлении синхронными двигателями.
2. Применение МПС в управлении асинхронными двигателями.
3. Цифровые датчики.
4. Цифровые регуляторы.
5. Частотное управление с помощью цифровых частотных преобразователей.

2.4 Вопросы к зачету

1. Понятие программируемого логического контроллера.
2. Модульная архитектура ПЛК.
3. Технические характеристики контроллеров.
4. Критерии выбора контроллера для конкретного объекта.
5. Промышленные сети.
6. Типы архитектур АСУ.
7. Примеры программируемых контроллеров.
8. Контроллеры для малой автоматизации.
9. Типичные примеры автоматизации промышленных объектов с помощью ПЛК.
10. Семейство контроллеров SIMATIC S7-200.
11. Аппаратура SIMATIC S7-200.
12. Центральные процессоры S7-200.
13. Модули SIMATIC S7-200.
14. Модули человеко-машинного интерфейса SIMATIC HMI.
15. Программное обеспечение для SIMATIC S7-200 STEP 7 Micro/WIN.
16. Схема блока питания с трансформаторным входом.
17. Схема блока питания с бестрансформаторным входом.
18. ЦАП.
19. Основные параметры ЦАП.
20. Схема простейшегоЦАП.
21. ЦАП с резистивной матрицей R-2R.
22. АЦП.
23. АЦП прямого преобразования.
24. АЦП последовательного счета.
25. Общая структура МПС.
26. Понятие интерфейса в МПС.
27. Аппаратный интерфейс.
28. Программный интерфейс.
29. Мультиплексирование многоразрядной шины.
30. Дешифратор адреса в МПС.
31. Основные функции МПС.
32. Способы передачи информации в МПС.
33. Асинхронная передача данных.
34. Синхронная передача данных.

35. Интерфейс RS232. Основные характеристики.
36. Интерфейс RS485. Основные характеристики.
37. Цифровые датчики.
38. Цифровые регуляторы.
39. Частотное управление с помощью цифровых частотных преобразователей.
40. Применение МПС в управлении синхронными двигателями.

3. ЛИТЕРАТУРА

Основная:

1. Анкудинов И.Г. Микропроцессорные системы. Архитектура и поектрование: Учеб. Пособие. – СПб.: СЗТУ, 2003. – 109 с.

Дополнительная:

1. Хвощ С.Т., Варлинский Н.Н., Попов Е.А. Микропроцессоры и микроЭВМ в системах автоматического управления: Справочник. – Л.: Машиностроение, 1987. – 640 с.
2. Угрюмов Е.П. Проектирование элементов и узлов ЭВМ. – М.: Высш. школа, 1987. – 318 с.
3. Майоров С.А., Кириллов В.В., Приблуда А.А. Введение в микроЭВМ. – Л.: Машиностроение, 1988. – 304 с.

II. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПРОВЕДЕНИЮ И ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

Лабораторные работы проводятся по подгруппам в компьютерном классе. Задания к лабораторным работам выполняются парой студентов на одном компьютере соответствии с вариантом.

Выполняя задание, студенты пользуются материалом, изложенным в тексте лабораторной работы; готовят письменный отчет, включающий краткое изложение проделанных действий, выводы.

Преподаватель, принимая лабораторную работу, проверяет навыки, полученные студентами при выполнении задания, отчет, задает дополнительные вопросы по теоретическому материалу.

III. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ И ЗАДАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ

Указание: сначала прочесть весь текст, иметь представление о содержимом, а затем выполнить задание. На каждом занятии студенты должны выполнить по одному заданию из приведенных в конце раздела. На последней паре проводится защита лабораторных работ.

Руководство для выполнения лабораторных работ на учебном стенде «Мега»

Руководство программиста

1. Устройства

Возможно применение следующих устройств:

- вход (X) – дискретный вход, обозначающийся X00-X0F (по номеру входа)
- выход (Y) – дискретный выход, обозначающийся Y00-Y0F (по номеру выхода)
- дискретный вход, всегда равный 0 – I0
- дискретный вход, всегда равный 1 – I1
- таймер (T) – дискретный таймер (T00-T0F, T10-T1F)
- регистр (R) – временная дискретная переменная (R00-R0F)
- аналоговый вход (A) – аналоговый вход (A00-A0F)
- специальные флаги – специальные флаги (например, флаги сравнения) (F00-F02)

2. Команды

2.1. Load, Load Inverse

LD (Load) – загрузить дискретный флаг в аккумулятор

LDI (Load Inverse) – загрузить инверсный дискретный флаг в аккумулятор

Применяется к: X, I, Y, R, F, T.

Пример программы:

```
0     LD  X00
1     OUT Y00
2     LDI X01
3     OUT R01
4     OUT T00
5     LD  T00
6     OUT Y01
7     LD    I0
8     OUT  Y02
```

2.2. Out

OUT (Out) – выдать на дискретный выход значение аккумулятора
Применяется к: Y, R, T
Примечание: возможно применение нескольких инструкций OUT параллельно.
Пример программы: см. пример к команде LD

2.3. And, And Inverse

AND (AND) – выполнить логическое И с аккумулятором и сохранить результат в аккумулятор
ANI (AND Inverse) – выполнить логическое И инвертированного дискретного флага с аккумулятором и сохранить результат в аккумулятор
Применяется к: X, I, Y, R, F, T
Пример программы:

```
0    LD   X02
1    AND  X00
2    OUT  Y03
3    LD   Y03
4    ANI  X03
5    OUT  R01
6    AND  T01
7    OUT  Y04
```

2.4. Or, Or Inverse

OR (OR) – выполнить логическое ИЛИ с аккумулятором и сохранить результат в аккумуляторе
ORI (OR Inverse) – выполнить логическое ИЛИ инвертированного флага с аккумулятором и сохранить результат в аккумуляторе
Применяется к: X, I, Y, R, F, T
Пример программы:

```
0    LD   X04
1    OR   X06
2    ORI  R02
3    OUT  Y05
4    LDI  Y05
5    AND  X07
6    OR   R03
7    ANI  X10
8    OR   R04
9    OUT  R03
```

2.5. XOR, XOR Inverse

XOR (XOR) – выполнить логическое ИСКЛЮЧАЮЩЕЕ ИЛИ с аккумулятором и сохранить результат в аккумуляторе
XRI (XOR Inverse) – выполнить логическое ИСКЛЮЧАЮЩЕЕ ИЛИ инвертированного флага с аккумулятором и сохранить результат в аккумуляторе
Применяется к: X, I, Y, R, F, T

Пример программы:

```
0     LD   X05
1     XOR  Y05
2     OUT  Y05
3     LD   Y05
4     XRI  Y05
5     OUT  Y06
```

2.6. Inverse

INV (Inverse) – инвертировать значение аккумулятора

Применяется к: -

Пример программы:

```
0     LD   X06
1     OR   Y06
2     INV
3     OUT  Y06
```

2.7. Set, Reset

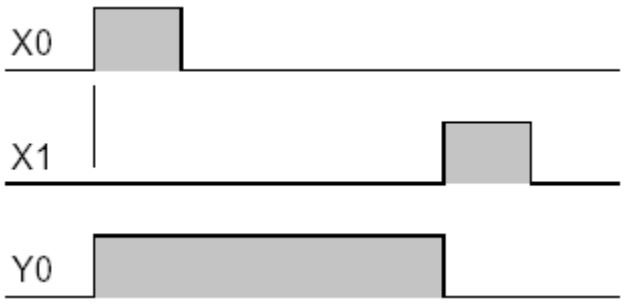
SET (SET) – установить значение дискретного флага в 1, если результат в аккумуляторе равен 1 (TRUE)

RST (ReSeT) – сбросить значение дискретного флага в 1, если результат в аккумуляторе равен 1 (TRUE)

Применяется к: Y, R, T (только для RST)

Пример программы:

```
0     LD   X00
1     SET  Y00
2     LD   X01
3     RST  Y00
4     LD   X02
5     SET  R00
6     LD   X03
7     RST  R00
```



2.8. CMP

CMP (CoMPare) – сравнить аналоговый вход с константой или аналоговым входом. Сохраняет результат в специальных флагах F00, F01, F02.

Например: CMP A00, 100

После выполнения F00 = 1, если A00 = 100

F01 = 1, если A00 < 100

F02 = 1, если A00 > 100

Применяется к: A

Пример программы:

```
0     CMP  A00, 100
1     LD   F00
2     OR   F01
```

```
3       OUT  Y00
```

2.9. MOV

MOV (MOVe) – записать значение в настроечную ячейку памяти, если значение аккумулятора равно 1 (TRUE)

Применяется к: T

Пример программы:

```
0       LD   X00
1       MOV T00,  1000
2       LD   X01
3       OUT  T00
4       LD   T00
5       OUT  Y00
```

2.10. END

END (END) – начать с начала итерацию программного цикла

3. Таймеры

3.1. Общие сведения

Таймер имеет вход, разрешающий счёт и выход, показывающий, что внутренний счётчик достиг нужного значения.

Таймеры бывают двух видов – с ручным и автоматическим сбросом. В таймере с автоматическим сбросом, при пропадании сигнала разрешения счёта, внутренний счётчик таймера сбрасывается; в ручном – сброс должен производится вручную.

Минимальный период таймера – 1мс.

После того, как внутренний счётчик таймера будет больше или равен периоду таймера, на его выходе выдаётся логическая 1. До этого момента на выходе таймера стоит логический 0.

3.2. Инициализация

Для задания периода таймера используется команда MOV. Здесь первый операнд представляет собой номер таймера, второй – число фреймов, через которых включенный таймер должен сработать. При переинициализации внутренний счётчик фреймов не сбрасывается.

3.3. Таймер с ручным сбросом

Таймеры с ручным сбросом имеют номера T00 – T0F. Для сброса его внутреннего счётчика должна использоваться команда RST

3.4. Таймер с автоматическим сбросом

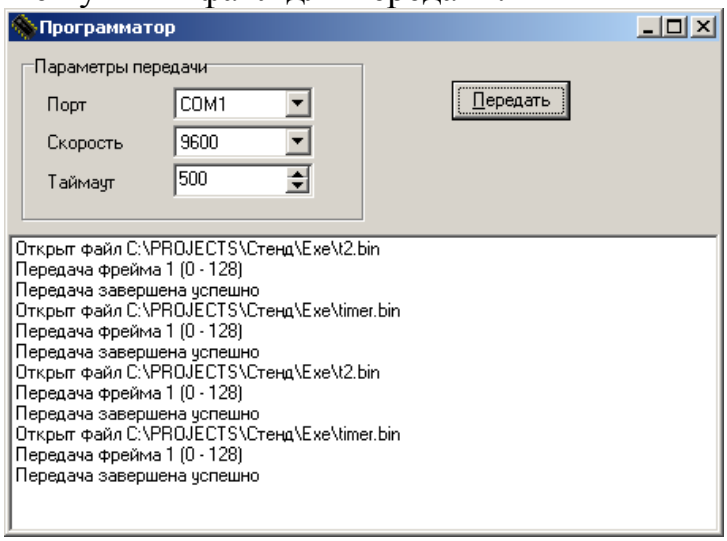
Таймеры с автоматическим сбросом имеют номера T10 – T1F. При исчезновении сигнала разрешения счёта его внутренний счётчик автоматически сбрасывается.

Руководство пользователя

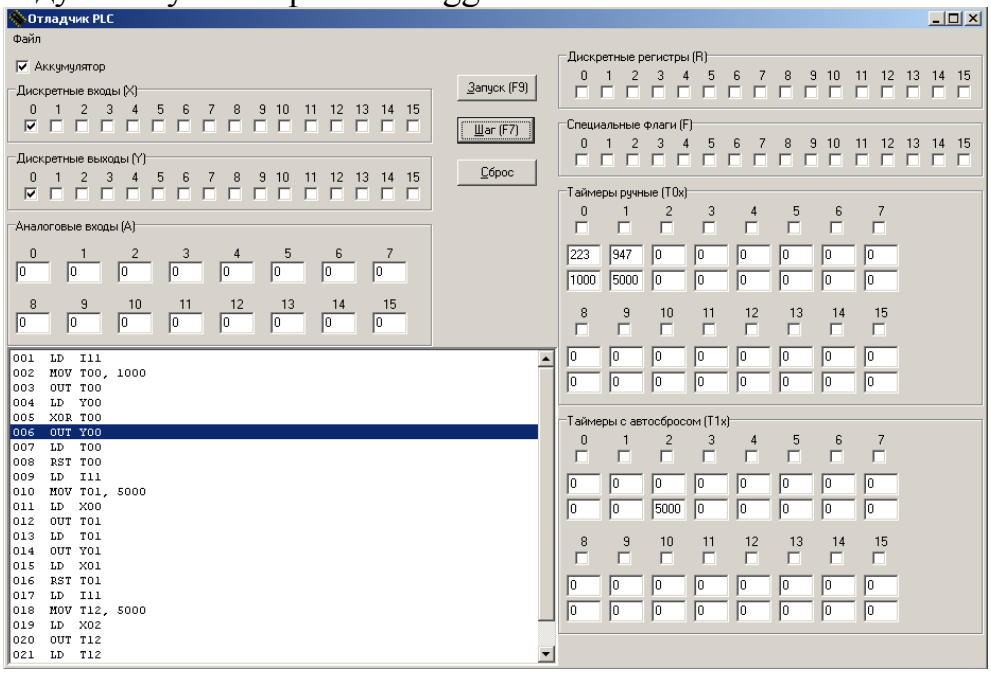
Написание программы производится в любом текстовом редакторе, например, в Far-е.

Для её дальнейшей компиляции следует запустить файл compile.exe с параметрами, куда передаются имена входного и выходного файлов. Например, *compile.exe logic.pla logic.bin*

Передача программы на стенд осуществляется посредством программы Sender. Запустите её, установите параметры соединения и нажмите на кнопку передать, после чего выберите нужный файл для передачи.



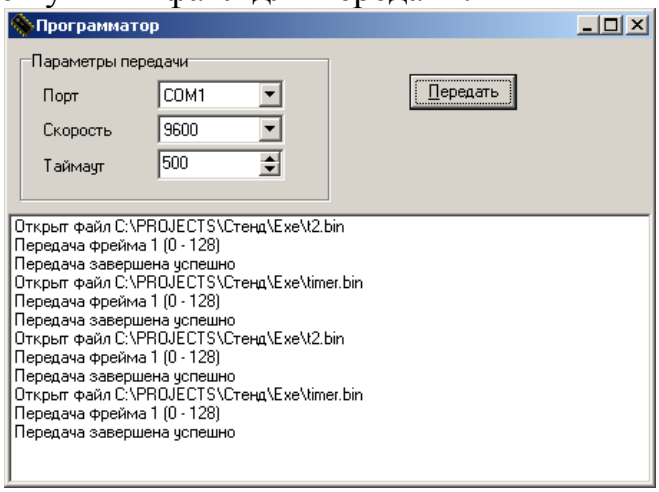
Для отладки программы можно использовать специальный отладчик. Для его запуска следует запустить файл debugger.exe



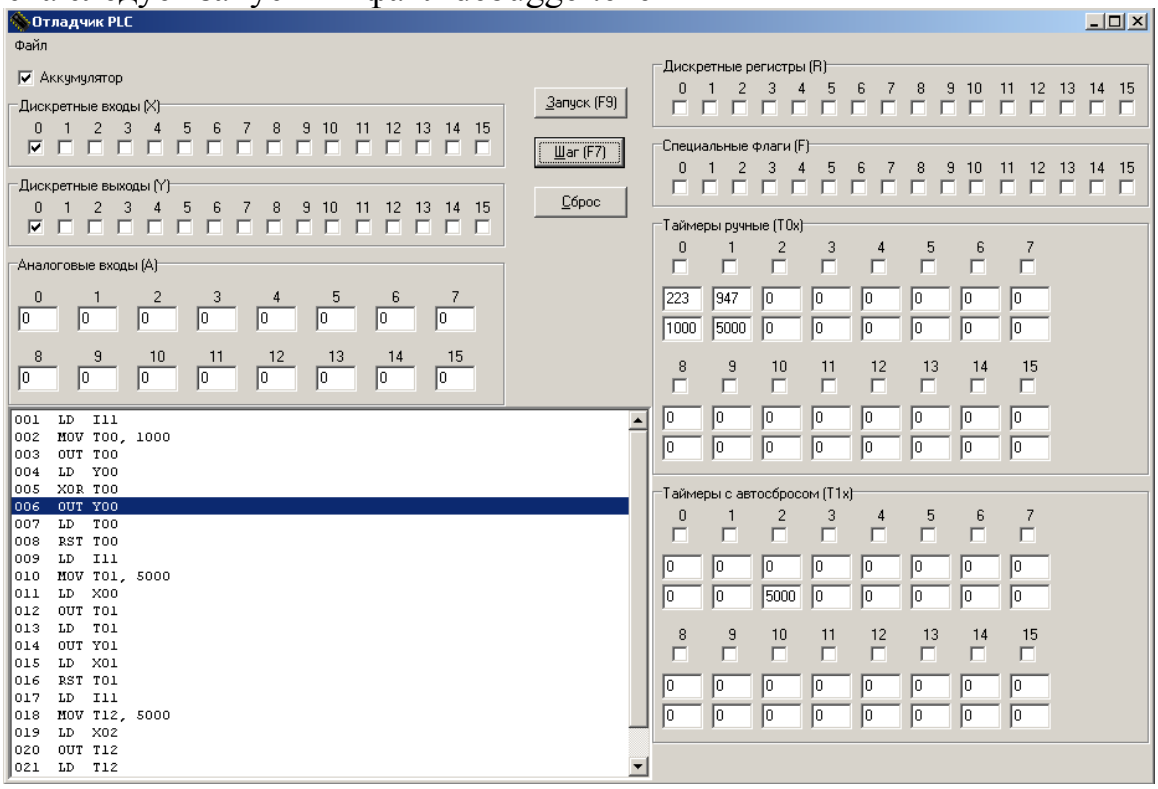
Написание программы производится в любом текстовом редакторе, например, в Far-e.

Для её дальнейшей компиляции следует запустить файл compile.exe с параметрами, куда передаются имена входного и выходного файлов. Например, *compile.exe logic.pla logic.bin*

Передача программы на стенд осуществляется посредством программы Sender. Запустите её, установите параметры соединения и нажмите на кнопку передать, после чего выберите нужный файл для передачи.



Для отладки программы можно использовать специальный отладчик. Для его запуска следует запустить файл debugger.exe



Руководство для выполнения лабораторных работ на учебном стенде «Siemens S7-200»

С помощью STEP 7-Micro/WIN вы можете легко программировать свой S7-200. Всего занесколько коротких шагов на простом примере вы узнаете, как подключать, программировать иэксплуатировать свой S7□200.

Для этого примера вам потребуется кабель PC/PPI, CPU S7-200 и устройство программирования, на котором установлено программное обеспечение STEP 7-Micro/WIN.

Подключение CPU S7.200

Подключить ваш S7.200 совсем несложно. В этом примере вам нужно только присоединить к CPU S7.200 источник питания, а затем присоединить кабель связи к устройству программирования и CPU S7.200.

Подключение источника питания к CPU S7.200

Сначала подключите S7.200 к источнику питания. На рис. 2.1 показано присоединение проводов для исполнений CPU S7.200 постоянного и переменного тока. Перед монтажом или демонтажем любого электрического устройства вы должны обеспечить, чтобы питание этого устройства было выключено. Выполните все необходимые предписания по технике безопасности и убедитесь, что перед монтажом или демонтажем S7.200 его питание выключено.

Присоединение кабеля PC/PPI

На рис. 2.2 показан кабель PC/PP, соединяющий S7.200 с устройством программирования. Для подключения кабеля PC/PPI:

1. Присоедините штекер RS.232 (обозначенный «PC») кабеля PC/PPI к коммуникационному порту устройства программирования. (В этом примере используйте COM 1.)
2. Присоедините штекер RS.485 (обозначенный «PPI») кабеля PC/PPI к порту 0 или 1 S7.200.
3. Обратите внимание, чтобы DIP-переключатели на кабеле PC/PPI были установлены, как показано на рис. 2.2

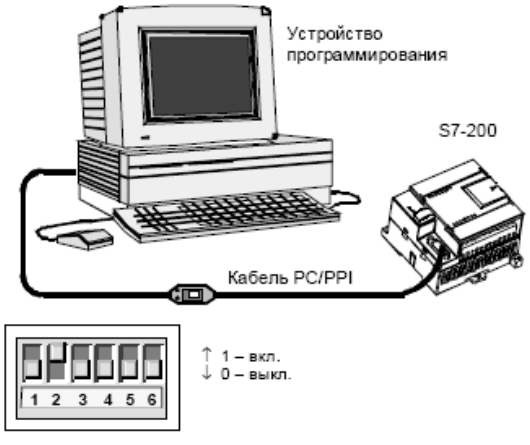


Рис. 2–2. Присоединение кабеля PC/PPI

Вызов STEP 7-Micro/WIN

Щелкните на символе STEP 7-Micro/WIN, чтобы открыть новый проект. На рис. 2.3 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.

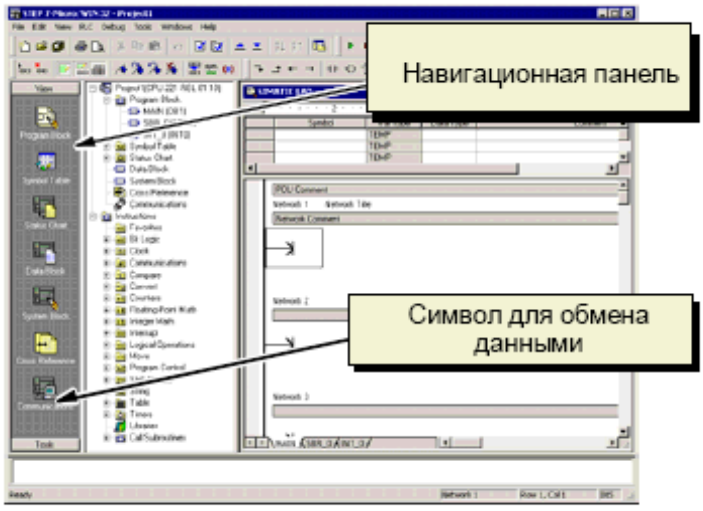


Рис. 2–3. Новый проект STEP 7-Micro/WIN

Проверка параметров обмена данными для STEP 7-Micro/WIN

Проект-пример использует для STEP 7-Micro/WIN и кабеля PC/PPI настройки по умолчанию. Проверьте эти настройки:

1. Проверьте, чтобы адрес кабеля PC/PPI в диалоговом окне Communications был установлен на 0.
2. Проверьте, чтобы в качестве интерфейса для сетевых параметров был установлен кабель PC/PPI (COM1).
3. Проверьте, чтобы для скорости передачи (transmission rate) было установлено значение 9.6 Кбит/с. Если вам необходимо изменить настройки параметров для обмена данными, прочитайте главу 7.

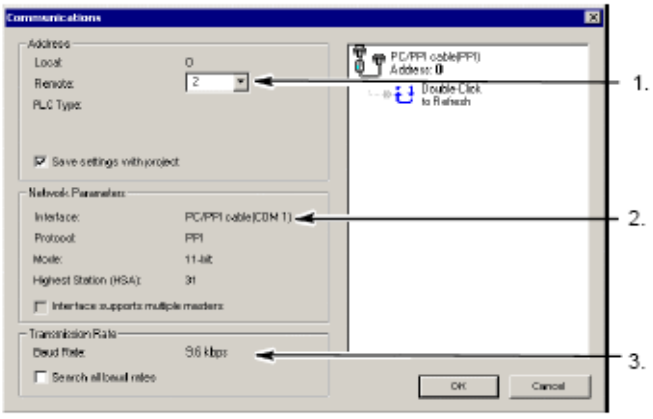


Рис. 2.4. Проверка параметров обмена данными

Установка связи с S7.200

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

1. 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.
2. 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.

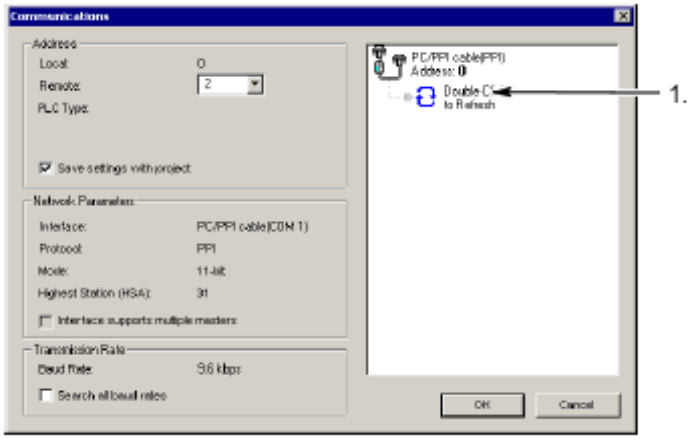
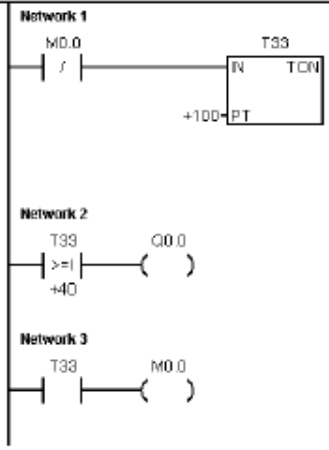


Рис. 2.5. Установка связи с S7.200

Создание программы-примера

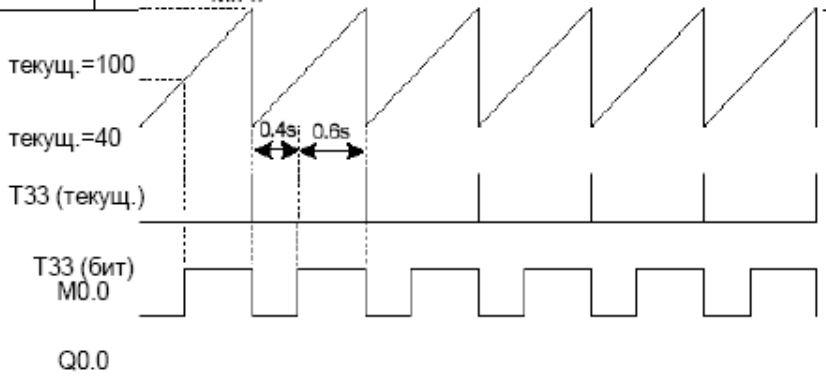
Ввод этого примера программы управления поможет вам понять, как просто работать со STEP 7-Micro/WIN. Эта программа содержит шесть команд в тех сегментах (Network) и образует из них очень простой таймер, сам запускается и сам себя сбрасывает. Команды для этого примера введите в редакторе LAD (КОР). Следующий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

Пример: Образец программы для начала работы со STEP 7-Micro/WIN




```
Network 1       //10-миллисекундный таймер T33 включается по
                //истечении (100 x 10 мс = 1 с)
                //Импульс M0.0 слишком быстр, чтобы его можно
                //было увидеть в отображении состояния
                //программы.
LDN     M0.0
TON     T33, +100
Network 2       //Сравнение становится истинным со скоростью,
                //позволяющей видеть его результат в отображении
                //состояния программы. Включить Q0.0 по
                //истечении (40 x 10 мс = 0,4 с), чтобы получить
                //сигнал с соотношением времен
                //выключения/включения 40% /60%.
LDW>=   T33, +40
=       Q0.0
Network 3       //Импульс T33 (бит) слишком быстр, чтобы его
                //можно было увидеть в отображении состояния
                //программы.
                //Сбросить таймер через M0.0 по истечении
                //(100 x 10 мс  = 1 с).
LD      T33
=       M0.0
```

Импульсная диаграмма



Вызов редактора программ

Чтобы открыть редактор программ, щелкните на символе Program Block [Программный блок]. См. рис. 2.6. 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.

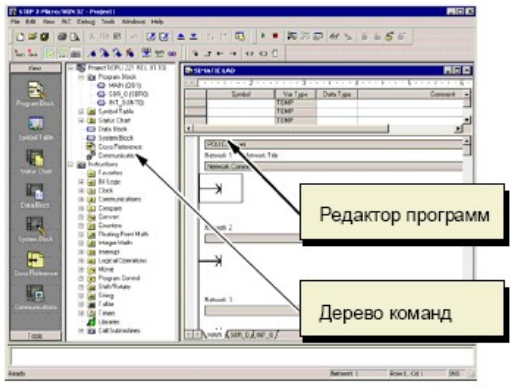


Рис. 2.6. Окно STEP 7-Micro/WIN

Ввод сегмента (Network) 1: Запуск таймера

Если M0.0 выключен (0), этот контакт включается и передает поток сигнала для запуска таймера. Для ввода контакта для M0.0:

1. Дважды щелкните на символе Bit Logic [Битовая логика] или один раз щелкните на знаке плюс (+) для отображения битовых логических операций.
2. Выберите нормально замкнутый 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. Чтобы ввести таймерную команду для T33:
1. Дважды щелкните на символе Timers [Таймеры], чтобы отобразить таймерные команды.

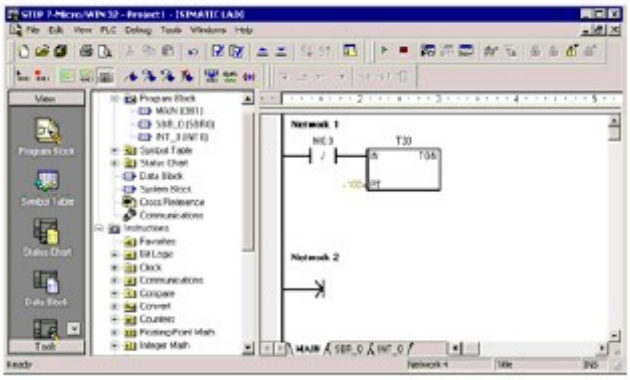


Рис. 2.7. Сегмент 1

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 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. Удеривая в нажатом состоянии левую кнопку мыши, перетащите эту команду сравнения во второйсегмент.
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.

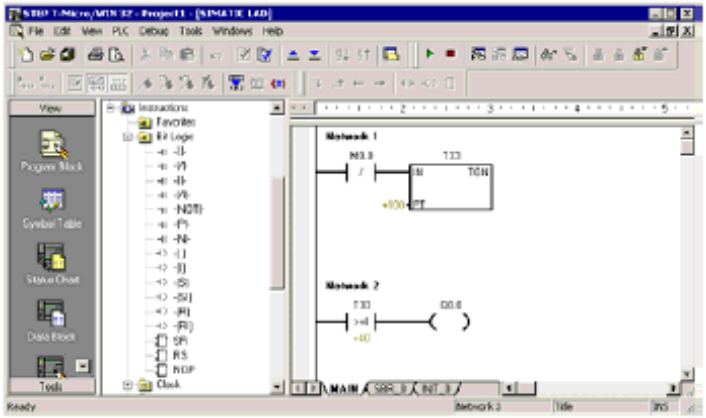


Рис. 2.8. Сегмент 2

5. Введите следующую величину для сравнения со значением таймера: 40
6. Нажмите клавишу Return, чтобы ввести это значение. 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 нормально открытый 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.

Чтобы ввести катушку для включения M0.0:

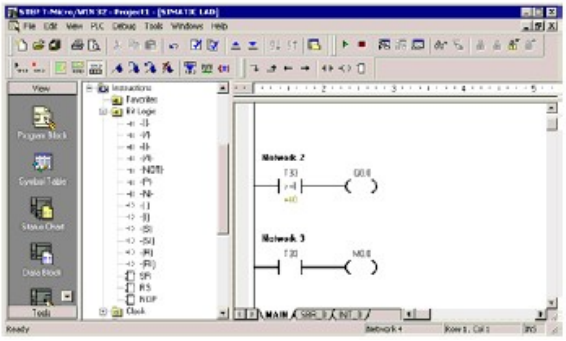


Рис. 2.9. Сегмент 3

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.

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

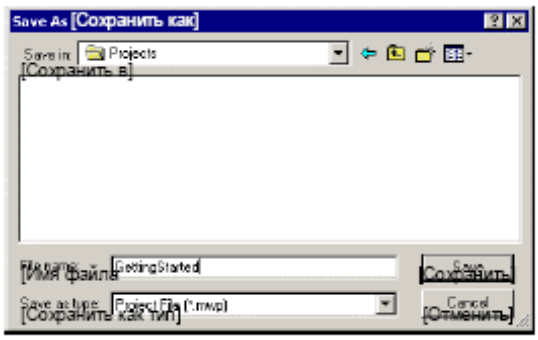


Рис. 2.10. Сохранение примера проекта

Загрузка программы-примера

символе Download [Загрузить], находящемся на панели инструментов, или выберите команду меню **File > Download [Файл > Загрузить]**. См. рис. 2.11.
2. Для загрузки элементов программы в S7.200 щелкните на OK.

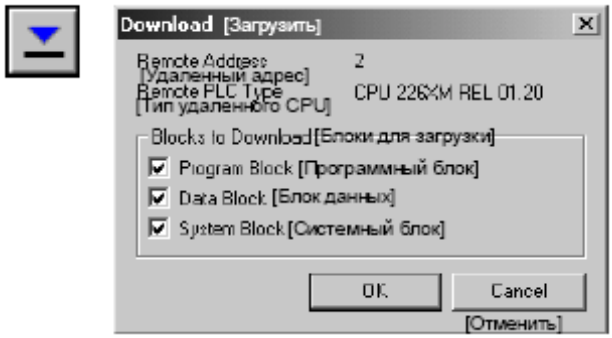


Рис. 2.11. Загрузка программы

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].

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 RUNS7.200 исполняет программу:

1. Щелкните на символе RUN, находящемся на панели инструментов, или выберите команду меню **PLC > RUN [ПЛК > RUN]**.
2. 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.

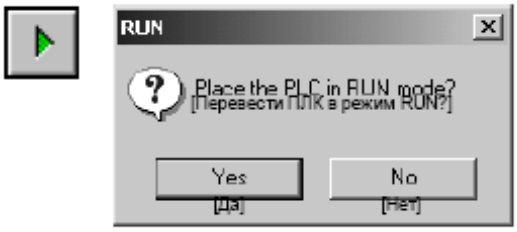


Рис. 2.12. перевод S7.200 в режим RUN

Поздравляем! Вы только что создали свою первую программу для S7.200.
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]**.

ОСНОВЫ ПЛК

Основной функцией S7.200 является контроль полевых входов и, на основе логики управления, включение и выключение полевых выходных устройств. В этой главе объясняются основы выполнения программы, различные виды используемой памяти и способы сохранения.

Выполнение логики управления с помощью S7.200

S7.200 обрабатывает логику управления в вашей программе циклически, считывая и записывая данные.

S7.200 ставит вашу программу в соответствие физическим входам и выходам

Основной принцип действия S7.200 очень прост:
- S7.200 считывает состояние входов.
- Программа, хранящаяся в S7.200, использует эти входы для анализа логики управления. Во время обработки программы S7.200 обновляет данные.
- S7.200 записывает данные на выходы.

На рис. 4.1 показана связь между простой коммутационной схемой и S7.200. В этом примере состояние выключателя для запуска двигателя логически связано с состояниями других входов. Оценки этих состояний определяют затем сигнальное состояние выхода для исполнительного устройства, которое запускает двигатель.

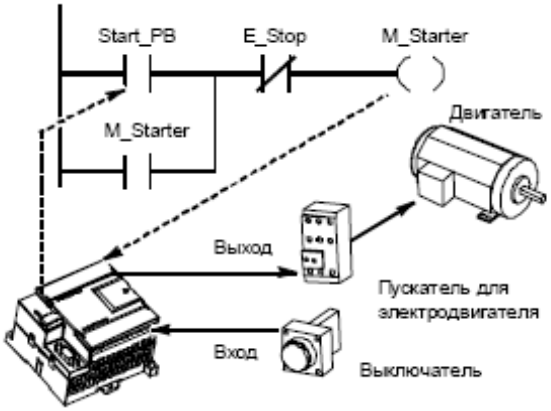


Рис. 4.1. Управление входами и выходами

S7.200 выполняет все задачи в цикле

S7.200 выполняет последовательность задач неоднократно. Эта регулярная обработка задач называется циклом. Как показано на рис. 4.2, S7.200 выполняет в цикле большинство или все из следующих задач:

- Чтение входов: S7.200 копирует состояние физических входов в регистр входов образа процесса.
- Выполнение логики управления в программе:

S7.200 выполняет команды программы и сохраняет значения в различных областях памяти.

- Обработка запросов на обмен данными: S7.200 выполняет все задачи, необходимые для обмена данными.
- Самодиагностика CPU: S7.200 проверяет, чтобы встроенное программное обеспечение, программная память и все модули расширения работали надлежащим образом.
- Запись в выходы: Значения, хранящиеся в регистре выходов образа процесса, записываются в физические выходы.

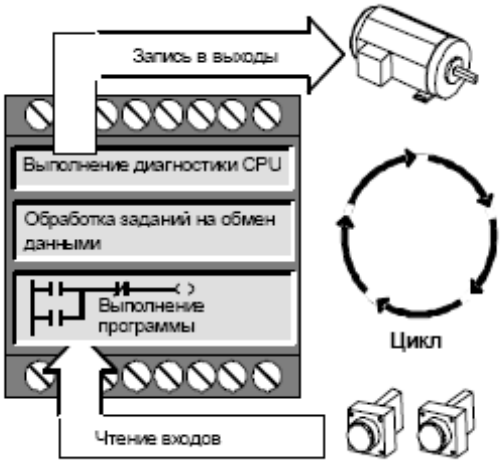


Рис. 4.2. Цикл S7.200

Выполнение цикла зависит от того, находится ли S7.200 в состоянии STOP или в состоянии RUN. В состоянии RUN ваша программа выполняется; в состоянии STOP ваша программа не выполняется.

Чтение входов

Цифровые входы: В начале цикла текущие значения цифровых входов считываются, а затем записываются в регистр входов образа процесса.
Аналоговые входы: S7.200 не обновляет аналоговые входы автоматически как часть цикла, если вы не активизировали фильтрацию аналоговых входов. Аналоговый фильтр обеспечивает стабильность сигналов. Вы можете активизировать аналоговый фильтр для каждого входа.
Если фильтр для аналогового входа активизирован, то S7.200 обновляет этот аналоговый вход один раз за цикл, выполняет функцию фильтрации и сохраняет отфильтрованное значение внутри. Это отфильтрованное значение затем предоставляется в распоряжение всякий раз, когда ваша программа обращается к этому аналоговому входу.
Если фильтр аналогового входа выключен, то S7.200 считывает значение этого аналогового входа из физического модуля всякий раз, когда ваша программа обращается к аналоговому входу.

Обработка программы

На этом участке цикла S7.200 обрабатывает программу с первой команды до последней. Вы можете непосредственно управлять входами и выходами и получать, таким образом, доступ к ним во время исполнения основной программы или программы обработки прерываний. Если вы используете в своей программе прерывания, то программы обработки прерываний, которые ставятся в соответствие прерывающим событиям, хранятся как часть основной программы. Однако программы обработки прерываний исполняются не как составная часть нормального цикла, а только тогда, когда происходит прерывающее событие (оно возможно в любом месте цикла).

Обработка запросов на обмен данными

На участке цикла, выделенном для обработки коммуникаций, S7.200 обрабатывает все сообщения, полученные из коммуникационного порта или от интеллектуальных модулей ввода/вывода.

Самодиагностика CPU

На этом участке цикла S7.200 проверяет надлежащую работу CPU, области памяти и состояние модулей расширения.

Запись в цифровые выходы

В конце каждого цикла S7.200 записывает значения, хранящиеся в регистре выходов образа процесса, в цифровые выходы. (Аналоговые выходы обновляются немедленно, независимо от цикла.)

Доступ к данным S7.200

S7.200 хранит информацию в различных местах памяти, которые имеют однозначные адреса. Вы можете явно указать адрес в памяти, к которому вы хотите обратиться. Благодаря этому ваша программа имеет прямой доступ к информации. Таблица 4.1 показывает диапазон целых значений, которые могут быть представлены с помощью данных различной длины.

Таблица 4–1. Десятичные и шестнадцатеричные диапазоны для данных различной длины

Представление	Байт (B)	Слово (W)	Двойное слово (D)
Целое без знака	от 0 до 255 от 0 до FF	от 0 до 65 535 от 0 до FFFF	от 0 до 4 294 967 295 от 0 до FFFF FFFF
Целое со знаком	от -128 до +127 от 80 до 7F	от –32 768 до +32 767 от 8000 до 7FFF	от –2 147 483 648 до +2 147 483 647 от 8000 0000 до 7FFF FFFF
Вещественное IEEE 32–битовое с плавающей точкой	*Неприменимо*	*Неприменимо*	от +1.175495E-38 до +3.402823E+38 (положительное) от –1.175495E-38 до –3.402823E+38 (отрицательное)

Для обращения к биту в некоторой области памяти вы должны указать адрес бита. Этот адрес состоит из идентификатора области памяти, адреса байта и номера бита. На рис. 4.3 показан пример обращения к биту (адресация в формате «байт.бит»). В этом примере за областью памяти и адресом байта (I = input [вход], 3 = байт 3) следует точка («.»), чтобы отделить адрес бита (бит 4).

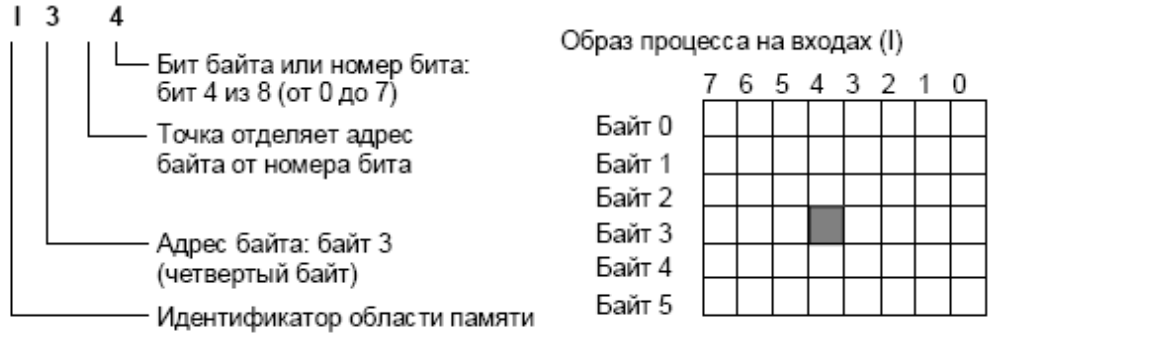


Рис. 4–3. Адресация байт.бит

Когда вы применяете формат байт.бит, вы можете обратиться к данным в большинстве областей памяти (V, I, Q, M, S, L и SM) как к байтам, словам или двойным словам. Если вы хотите обратиться к байту, слову или двойному слову данных в памяти, то вы должны указать эти адреса подобно адресу бита. Вы указываете идентификатор области, обозначение длины данных и начальный адрес байта, слова или двойного слова, как показано на рис. 4.4. К данным в других областях памяти (напр., T, C, HC и аккумуляторы) вы обращаетесь, указывая в качестве адреса идентификатор области и номер элемента.

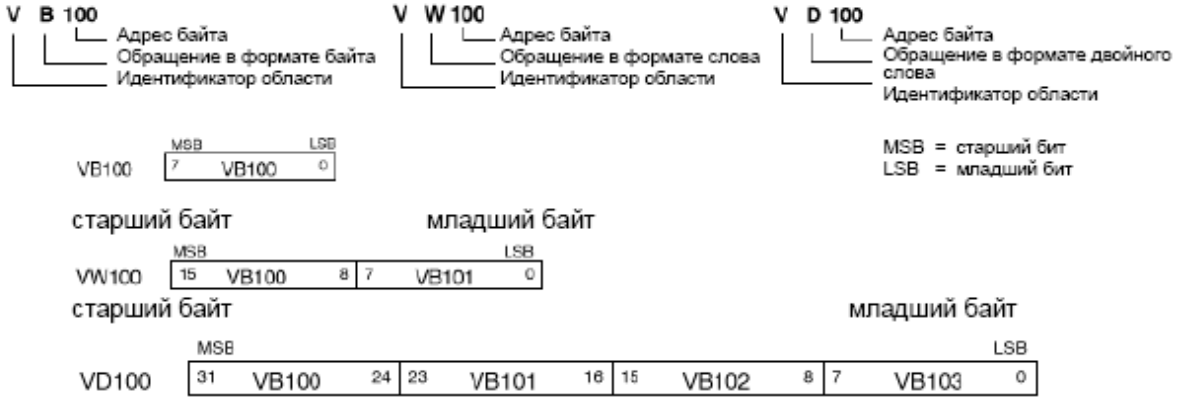


Рис. 4–4. Обращение к одному и тому же адресу в формате байта, слова и двойного слова

Обращение к данным в областях памяти

Регистр входов образа процесса: I

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

Бит: I*[адрес байта].[адрес бита]* I0.1

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

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]* Q1.1

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

Область памяти переменных: V

Память переменных можно использовать для хранения промежуточных результатов операций, выполняемых в вашей программе. В памяти переменных вы можете 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]* V10.2

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

Область битовой памяти: M

Биты памяти (меркеры) 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]* M26.7

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

Таймеры: T

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

Вы обращаетесь 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. Как показано на рис. 4.5, команда "Нормально открытый контакт" обращается к биту таймера, а команда "Передать слово" обращается к текущему значению таймера.

Рис. 4–5. Обращение к биту или к текущему значению таймера

Счетчики: C

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. Вы обращаетесь к обоим этим элементам данных 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. Как показано на рис. 4.6, команда "Нормально открытый контакт" обращается к биту счетчика, а команда "Передать слово" обращается к текущему значению счетчика.

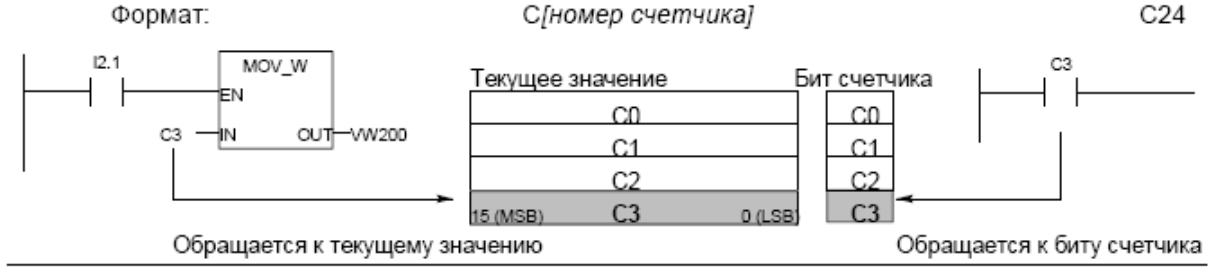


Рис. 4–6. Обращение к биту или к текущему значению счетчика

Скоростные счетчики: HC

Скоростные счетчики подсчитывают быстрые события независимо от цикла CPU. Скоростные счетчики имеют в своем распоряжении 32.битовое целое счетное значение (текущее значение). Для обращения к счетному значению скоростного счетчика введите его адрес, указав область памяти (HC) и номер счетчика (напр., HC0). Текущее значение скоростного счетчика защищено от записи и может быть адресовано только в формате двойного слова (32 бита).

Формат: HC*[номер скоростного счетчика]* HC1

Аккумуляторы: AC

Аккумуляторы . это элементы чтения/записи, которые могут использоваться как память. Например, вы можете использовать аккумуляторы для передачи параметров в подпрограммы и из них или для хранения промежуточных результатов расчетов. S7.200 имеет в своем распоряжении четыре 32.битовых аккумуля-

тора (AC0, AC1, AC2 и AC3). К данным в аккумуляторах можно обратиться в формате бита, слова или двойного слова. Длина данных, к которым производится обращение, зависит от команды, которая используется для обращения к аккумулятору. Как показано на рис. 4.7, при обращении к аккумулятору в формате бита или слова используются младшие 8 или 16 битов значения, хранящегося в аккумуляторе. При обращении к аккумулятору в формате двойного слова используются все 32 бита.

Информацию об использовании аккумуляторов в программах обработки прерываний вы найдете в главе 6.

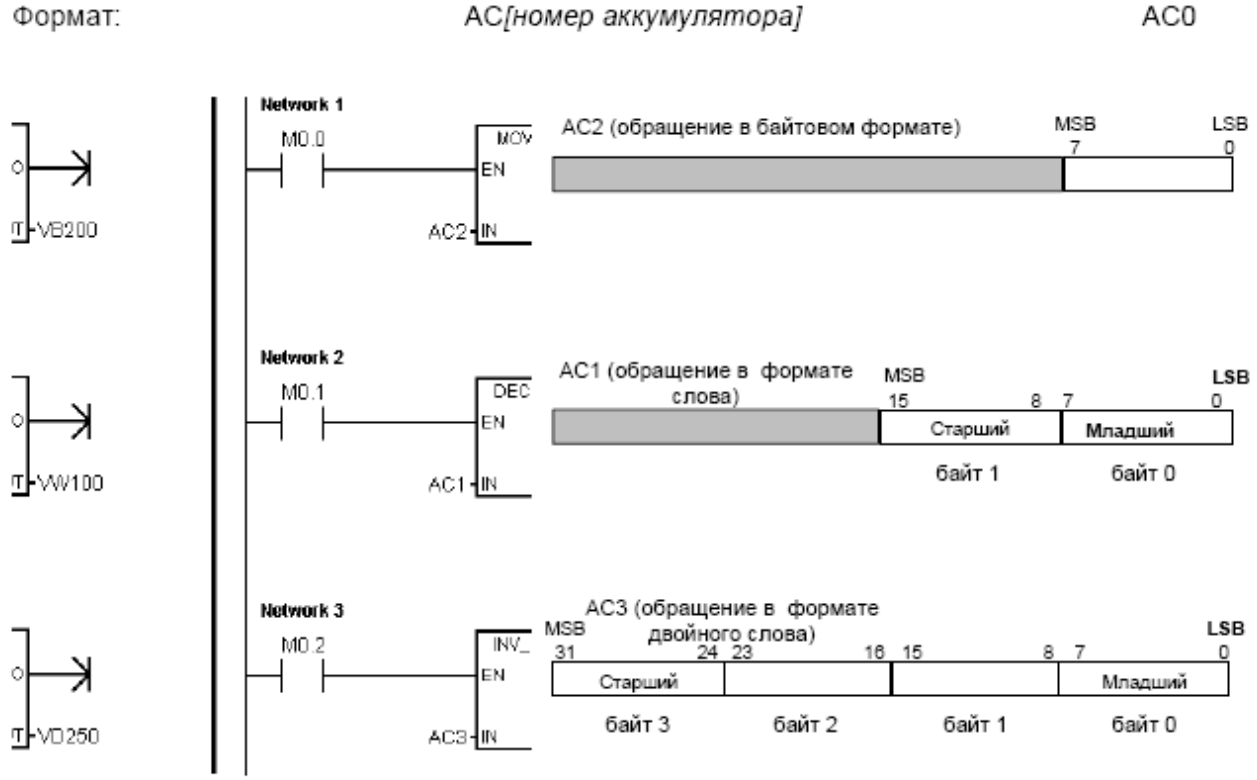


Рис. 4–7. Обращение к аккумуляторам

Специальные биты памяти: SM

Специальные биты памяти (SM) предоставляют средство для обмена данными между CPU и вашей программой. Вы можете 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. (Подробную информацию о специальных битах памяти вы найдете в Приложении D.) К SM-битам можно обращаться в формате бита, слова или двойного слова:

Бит: SM*[адрес байта].[адрес бита]* SM0.1

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

Память локальных данных: L

S7.200 имеет в своем распоряжении 64 байта локальной памяти, из которых 60 могут быть использованы в качестве промежуточной памяти или для передачи формальных параметров в подпрограммы. Память локальных данных похожа на память переменных с одним существенным отличием. Память переменных

доступна глобально, тогда как память локальных данных доступна локально. Глобальная доступность означает, что к адресу в этой области памяти можно обратиться из любой организационной единицы программы (из основной программы, подпрограммы или подпрограмм обработки прерываний). Локальная доступность означает, что эта область памяти ставится в соответствие определенной организационной единице программы. S7.200 выделяет 64 байта локальной памяти для главной программы, 64 байта для каждого уровня вложенности подпрограмм и 64 байта для программ обработки прерываний.

К области локальных данных, поставленной в соответствие основной программе, не имеют доступа подпрограмм и программы обработки прерываний. Подпрограмма не может обращаться к области локальных данных основной программы, программы обработки прерываний или другой подпрограммы. Аналогично, программа обработки прерываний не имеет доступа к области локальных данных основной программы или подпрограммы. S7.200 выделяет область локальных данных по мере необходимости. Это значит, что при выполнении основной программы области локальных данных для подпрограмм и программ обработки прерываний не существуют. Если возникает прерывание или вызывается подпрограмма, то по потребности выделяется локальная память. Вновь выделенная локальная память может снова использовать те же адреса, которые использовались другой подпрограммой или программой обработки прерываний. S7.200 не инициализирует область локальных данных к моменту ее назначения, поэтому она может содержать любые значения. Если при вызове подпрограммы передаются формальные параметры, то S7.200 сохраняет значения передаваемых параметров в соответствующих адресах области локальных данных, выделенной этой подпрограмме. Адреса в области локальных данных, которые не получили значений при передаче формальных параметров, не инициализируются и при выделении могут содержать произвольные значения.

Бит: L*[адрес*8.56 592.16 *байта].[адрес бита]* L0.0

Байт, слово или двойное слово: L*[длина] [начальный адрес байта]* LB33

Аналоговые входы: AI

S7.200 преобразует аналоговые величины (например, температуру или напряжение) в цифровые величины, имеющие длину слова (16 битов). Вы обращаетесь к этим значениям через идентификатор области (AI), длину данных (W) и начальный адрес байта. Так как в случае аналоговых входов речь идет о словах, которые всегда начинаются на байтах с четными номерами (например, 0, 2, 4 и т.д.), то вы обращаетесь к этим значениям с помощью адресов четных байтов (например, AIW0, AIW2, AIW4). Аналоговые входы можно только считывать.

Формат: AIW*[начальный адрес байта]* AIW4

Аналоговые выходы: AQ

S7.200 преобразует цифровые величины, имеющие длину слова (16 битов), в ток или напряжение пропорционально цифровой величине. Вы обращаетесь к этим значениям через идентификатор области (AQ), длину данных (W) и начальный адрес байта. Так как в случае аналоговых выходов речь идет о словах, которые всегда начинаются на байтах с четными номерами (например, 0, 2, 4 и

т.д.), то вы записываете эти значения с адресами четных байтов (например, AQW0, AQW2, AQW4). Аналоговые выходы можно только записывать.
Формат: AQW*[начальный адрес байта]* AQW4

Реле управления очередностью (SCR): S

SCR или S-биты разделяют функционирование установки на отдельные шаги или эквивалентные части программы. С помощью реле управления очередностью программа управления представляется в виде структуры, состоящей из логических сегментов. К S-битам можно обращаться в формате бита, слова или двойного слова.
Бит: S*[адрес байта].[адрес бита]* S3.1
Байт, слово или двойное слово: S*[длина][начальный адрес байта]* SB4

Формат вещественных чисел

Вещественные числа (или числа с плавающей точкой) представляются как 32.битовые числа однократной точности, формат которых описан в стандарте NSI/IEEE 754-1985. См. рис. 4.8.
Обращение к вещественным числам производится в формате двойного слова.

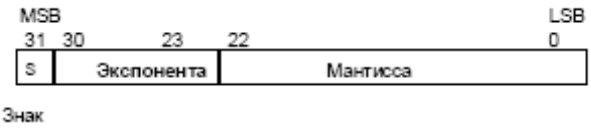


Рис. 4–8. Формат вещественного числа

У S7.200 числа с плавающей точкой имеют точность до 6 десятичных разрядов. Поэтому при вводе константы с плавающей точкой можно указывать до 6 десятичных разрядов.

Точность при вычислениях с вещественными числами

Расчеты, включающие в себя длинные последовательности значений, содержащие очень большие и очень малые числа, могут привести к неточным результатам. Это может произойти, если числа отличаются друг от друга в 10 в степени x раз, где $x > 6$.
Например: 100 000 000 + 1 = 100 000 000

Формат для строк

Строка . это последовательность символов, причем каждый символ хранится как байт. Первый байт строки определяет ее длину, т.е. количество содержащихся в ней символов. На рис. 4.9 показан формат строки. Строка может включать в себя от 0 до 254 символов, плюс байт, содержащий информацию о длине, таким образом, максимальная длина строки равна 255 байтам.

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

Рис. 4–9. Формат строк

Задание констант для команд S7.200

Во многих командах для S7.200 можно использовать константы. Константы могут быть байтами, словами или двойными словами. S7.200 хранит все константы в виде двоичных чисел, которые могут быть представлены в десятичном, шестнадцатеричном формате, в формате ASCII или в формате вещественных чисел (чисел с плавающей точкой). См. таблицу

4.2.

Таблица 4–2. Представление постоянных величин

Представление	Формат	Пример
Десятичное	[десятичное значение]	20047
Шестнадцатеричное	16#[шестнадцатеричное значение]	16#4E4F
Двоичное	2#[двоичное число]	2#1010_0101_1010_0101
ASCII	'[текст ASCII]'	Текст в одинарных кавычках.'
Вещественное	ANSI/IEEE 754-1985	+1.175495E-38 (положительное) –1.175495E-38 (отрицательное)

Адресация встроенных входов/выходов и входов/выходов модулей расширения

Встроенные входы и выходы центрального устройства (CPU) имеют фиксированные адреса. Вы можете добавить входы и выходы к CPU S7.200, подключив с правой стороны CPU модули расширения. Адреса входов и выходов на модуле расширения определяются видом входов и выходов, а у нескольких модулей одного типа также их расположением. Например, модуль вывода не влияет на адреса модуля ввода и наоборот. Адреса входов и выходов аналоговых и цифровых модулей также не зависят друг от друга.

На рис. 4.10 показан пример нумерации входов и выходов для конкретной конфигурации аппаратуры. Пропуски в адресации (показаны серым курсивом) не могут использоваться вашей программой.

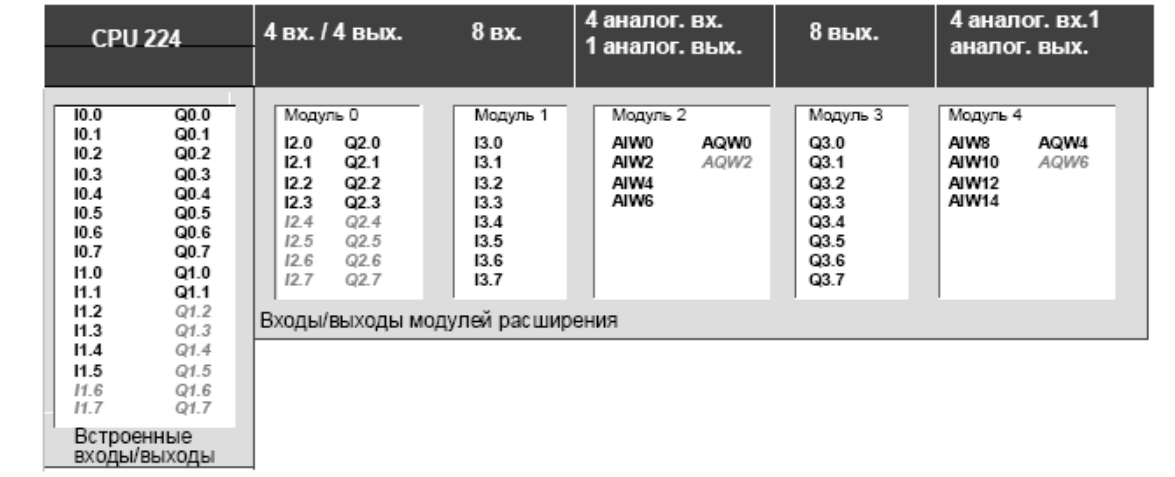


Рис. 4–10. Пример адресов встроенных входов/выходов и входов/выходов модулей расширения (CPU 224)

Косвенная адресация областей памяти S7.200 с помощью указателей

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 или аккумуляторные регистры (AC1, AC2 или AC3). Для создания указателя необходимо использовать команду "Переместить двойное слово". Эта команда передает адрес косвенно адресованной ячейки памяти в ячейку указателя. Указатели могут также передаваться в подпрограмму в качестве параметров.

S7.200 дает возможность использования указателей для косвенной адресации следующих областей памяти: I, Q, V, M, S, T (только текущее значение) и C

(только текущее значение). Косвенную адресацию нельзя использовать для обращения к отдельному биту или к областям памяти AI, AQ, HC, SM или L. Если вы хотите косвенно обратиться к данным, расположенным по некоторому адресу в памяти, вы можете создать указатель на этот адрес, введя амперсанд (&) и соответствующий адрес. Входному операнду команды должен предшествовать амперсанд (&), чтобы указать на необходимость перемещения в ячейку, обозначенную в выходном операнде команды (указателе), адреса ячейки памяти, а не ее содержимого.

Ввод астериска (*) перед операндом команды указывает, что этот операнд является указателем. Как показано на рис. 4.11, ввод *AC1 указывает, что AC1 является указателем на слово, на которое ссылается команда "Переместить слово" (MOVW). В этом примере значения, хранящиеся в VB200 и VB201, перемещаются в аккумулятор AC0.

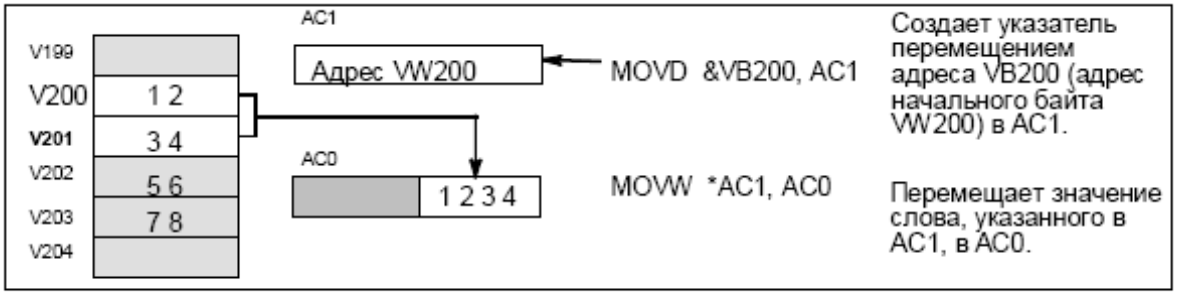


Рис. 4–11. Создание и использование указателя

Как показано на рис. 4.12, вы можете изменить значение указателя. Так как указатели имеют размер 32 бита, то для изменения значений указателей используйте операции над двойными словами. Для изменения значений указателей могут использоваться такие простые математические операции, как сложение или инкрементирование.

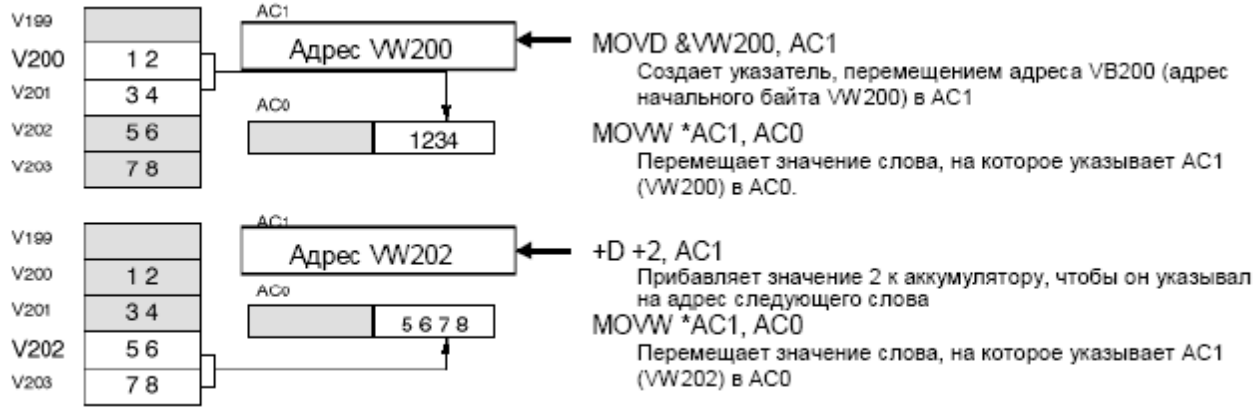
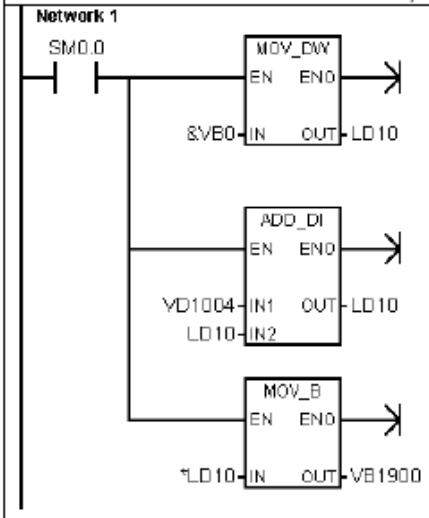


Рис. 4–12. Изменение указателя

Пример программы для обращения к данным в памяти переменных с использованием смещения

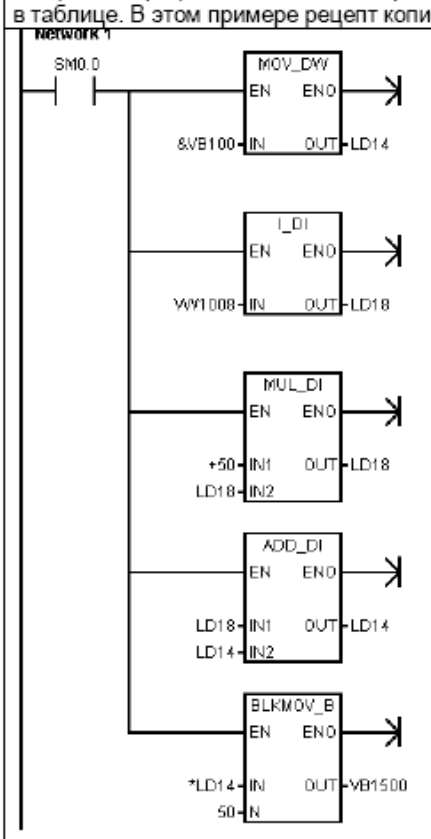
В этом примере используется LD10 как указатель на адрес VB0. Затем вы увеличиваете указатель на величину смещения, хранящуюся в VD1004. Теперь LD10 указывает на другой адрес в памяти переменных (VB0 + смещение). Значение, хранящееся в памяти переменных по адресу, на который указывает LD10, копируется в VB1900. Изменяя значение в VD1004, вы можете обратиться к любому адресу в памяти переменных.




```
Сегмент 1  //Чтение значения из произвольного адреса VB
           //с помощью смещения:
           //1. Загрузить в указатель начальный адрес памяти
           //   переменных.
           //2. Прибавить к указателю величину смещения.
           //3. Скопировать значение из адреса в памяти
           //   переменных (смещение) в VB1900.
LD     SM0.0
MOVD   &VB0, LD10
+D     VD1004, LD10
MOVB   *LD10, VB1900
```

Пример программы для обращения к данным в таблице с использованием смещения

Этот пример использует LD14 как указатель на рецепт, хранящийся в таблице рецептов, которая начинается с VB100. В этом примере VW1008 хранит индекс места конкретного рецепта в таблице. Если каждый рецепт в таблице имеет длину 50 байтов, умножьте индекс на 50, чтобы получить смещение для начального адреса конкретного рецепта. Добавив смещение к указателю, вы можете получить доступ к каждому отдельному рецепту в таблице. В этом примере рецепт копируется в 50 байтов, которые начинаются с VB1500.




```
Сегмент 1  //Передача рецепта из таблицы с рецептами:
           // - каждый рецепт имеет длину 50 байтов.
           // - Индексный параметр (VW1008) идентифицирует
           //   рецепт, подлежащий загрузке.
           //
           //1. Создание указателя на начальный адрес таблицы
           //   рецептов.
           //2. Преобразование индекса рецепта в значение
           //   двойного слова.
           //3. Умножение смещения для учета длины рецепта.
           //4. Прибавление измененного смещения к указателю.
           //5. Передача выбранного рецепта в ячейки с VB1500 по
           //   VB1549.
LD     SM0.0
MOVD   &VB100, LD14
ITD    VW1008, LD18
*D     +50, LD18
+D     LD18, LD14
BMB    *LD14, VB1500, 50
```

7. Упражнения для лабораторных работ

1. Из руководства для выполнения лабораторных работ на учебном стенде «Мега», из руководства для программиста выполнить первые пять примеров (2.1-2.5).
2. Из руководства для выполнения лабораторных работ на учебном стенде «Мега», из руководства для программиста выполнить четыре примеры 2.6-2.9.
3. Написать программу для контроллера для вывода чисел на семисегментном индикаторе в зависимости от входных сигналов с клавиатуры.
4. Написать программу для контроллера для системы сигнализации уровня воды в баке.
5. Написать программу для контроллера для системы управления углом поворота шагового двигателя.
6. Подключение контроллера SIMATIC S7-2000, настройка связи с PC, загрузка тестовой программы-примера.
7. Выполнить пример программы для обращения к данным в памяти переменных с использованием смещения. Выполнить пример программы для обращения к данным в таблице с использованием смещения.

IV. КОНСПЕКТ ЛЕКЦИЙ

Тема №1 Программируемые контроллеры (16 часов)

Программируемые контроллеры предназначены для построения систем автоматизации и решают задачи сбора данных и управления. Кроме того, они существляют обмен информацией с системами верхнего уровня (операторскими и инженерными станциями).

Программируемые контроллеры обычно имеют модульную архитектуру. В их состав входят:

- Модуль центрального процессора (CPU). В зависимости от степени сложности решаемой задачи в контроллерах могут быть использованы различные типы центральных процессоров, отличающихся производительностью, объемом памяти, наличием или отсутствием встроенных входов-выходов и специальных функций, количеством и видом встроенных коммуникационных интерфейсов и т.д.
- Модули блоков питания (PS(Power Supply)), обеспечивающие возможность питания контроллера от сети переменного тока напряжением 120/230В или от источника постоянного тока напряжением 24/48/60/110В.
- Сигнальные модули (SM), предназначенные для ввода-вывода дискретных и аналоговых сигналов с различными электрическими и временными параметрами.
- Коммуникационные процессоры (CP(communication processor)) для подключения к сетям CAN, PROFIBUS, Industrial Ethernet, AS-Interface и др. или организации связи по PtP (point to point) интерфейсу.
- Функциональные модули (FM), способные самостоятельно решать задачи автоматического регулирования, позиционирования, обработки сигналов. Функциональные модули снабжены встроенным микропроцессором и способны выполнять возложенные на них функции даже в случае отказа центрального процессора ПЛК.
- Интерфейсные модули (IM), обеспечивающие возможность подключения к базовому блоку (стойка с CPU) стоек расширения ввода-вывода.

Конструктивно промышленные контроллеры выпускаются в приборном или шкафном исполнении.

В приборном исполнении контроллер представляет собой набор приборов (блоков), имеющих лицевые панели и допускающих «утопленный» монтаж на щитах управления или монтаж на DIN-рельс. Пример: контроллеры Р130, КР300И.

В шкафном исполнении контроллер представляет собой набор плат, вставляемых в специальный конструктив (крейт), который монтируется в электротехническом шкафу.

При выборе контроллера основными критериями являются:

Количество подключаемых дискретных и аналоговых входов-выходов. По этому критерию системы автоматизации подразделяют на системы:

– малой мощности (десятки сигналов), централизованное управление отдельным оборудованием или производственной линией;

– средней мощности (сотни сигналов), централизованное управление технологическим процессом;

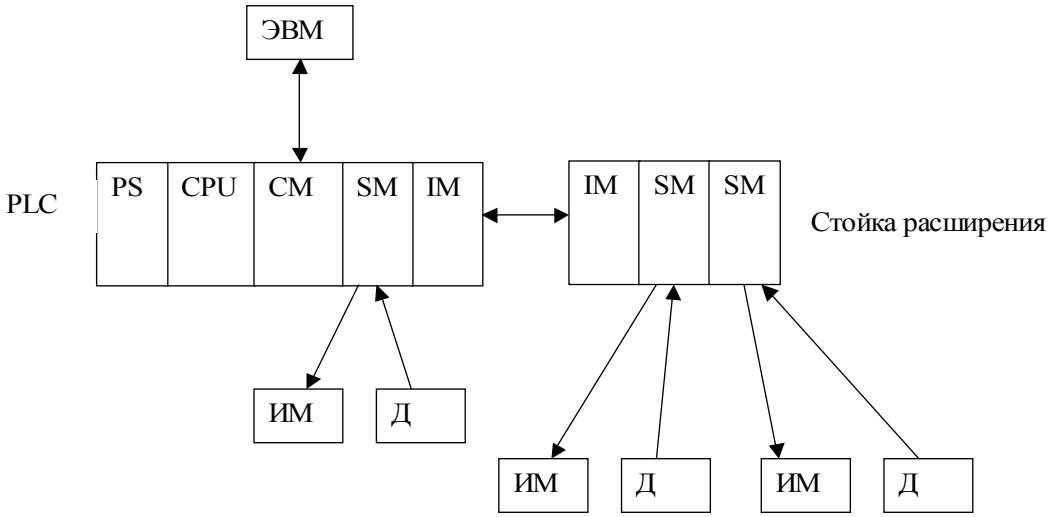
– большой мощности (тысячи сигналов), централизованное управление производственным процессом.

Большинство промышленных контроллеров допускают изменять состав модулей ввода-вывода, однако максимальное количество сигналов всегда ограничено.

В настоящее время основной тенденцией при построении АСУ является переход на децентрализованное (распределенное) управление с приближением УСО и контроллеров непосредственно к объектам измерения и управления.

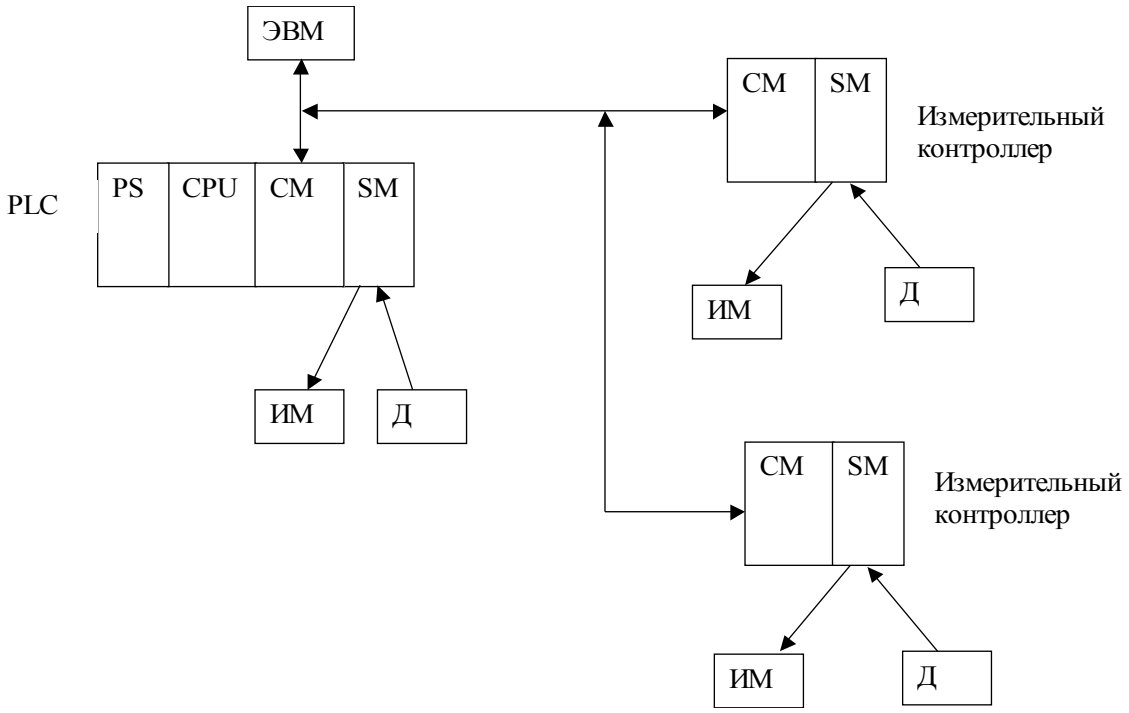
Варианты подключения контроллеров в составе АСУ:

1) централизованный контроль и управление



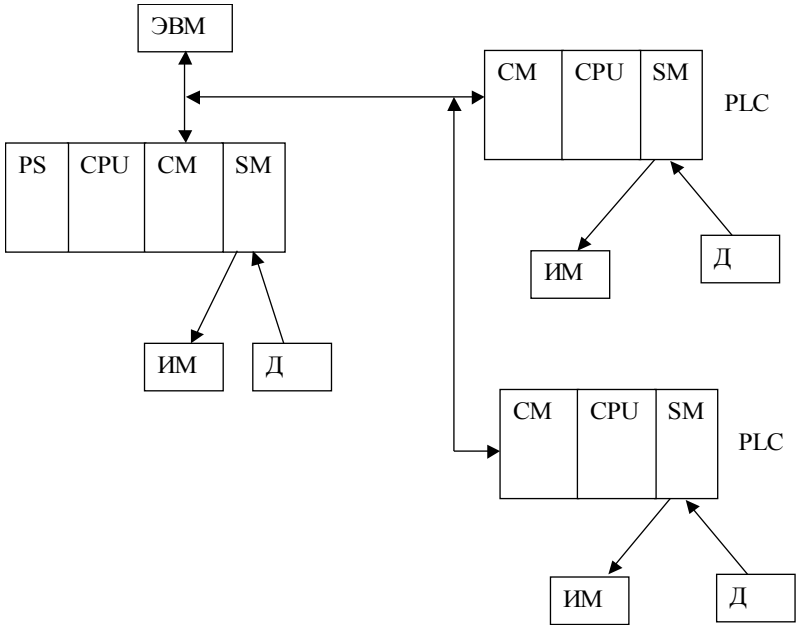
Недостатки: большой расход кабеля, необходимость резервирования дорогого CPU, сложность программного обеспечения (в больших системах, как правило, требуется применение многозадачных операционных систем реального времени с организацией работы по прерываниям, так как циклический опрос всех датчиков занимает очень много времени).

2) распределенная система с интеллектуальными УСО (распределенный контроль)



Недостатки: необходимость резервирования дорогого CPU, сетевого оборудования и кабеля, так как отказ любого из этих компонентов приводит к краху системы, сложность программного обеспечения.

3) распределенная система с «активными» контроллерами (распределенный интеллект)



В таком варианте управляемость процесса не теряется даже в случае отказа сети, так как «ответственные» вычисления, например, расчет управляющих воздействий при регулировании, выполняются «на местах». При этом существенно снижается нагрузка на сеть, а, следовательно, и требования к сетевому оборудо-

ванию. Резервированию подлежат недорогие контроллеры, причем только те, отказ которых приводит к существенным проблемам.

Кроме того, существенно упрощается программное обеспечение контроллеров.

Выбор архитектуры АСУ и контроллеров при заданном числе точек ввода-вывода является достаточно сложной задачей. Очевидно, что определяющим фактором при этом является

Стоимость проекта, которая складывается из стоимости оборудования, стоимости его внедрения и обслуживания.

Существуют компании-гиганты, например Siemens, предлагающие комплексные решения «под ключ».

С другой стороны, благодаря стандартизации оборудования, можно выбрать намного более дешевый, но не менее функциональный, вариант системы, используя оборудование разных производителей. Однако при этом потребуется большая квалификация инженеров-автоматчиков (предприятия или сторонних фирм) при разработке и внедрении проекта.

Стоимости оборудования сильно зависит от географии производителя. Наиболее дорогие (но, конечно, очень качественные) контроллеры производятся в США и Европе (Fastwell, Ortagon Systems, Grayhill, Allen Bradley, Honeywell, Motorola (США), Siemens, VIPA, WAGO и многие другие. Более дешевое оборудование производится в Японии (Mitsubishi) и России (Контраст (Чебоксары), Текон (Москва). Самыми недорогими являются контроллеры и сетевые УСО Тайваньских производителей Advantech (семейство ADAM).

При выборе PLC к основным критериям нужно также отнести такой, как наличие и номенклатура сетевых интерфейсов для построения распределенных систем и связи с верхним уровнем АСУ. На сегодняшний день известно несколько десятков сетевых стандартов, основными из которых являются Modbus, Profibus, CAN, Faundation Fieldbus, Interbus. Если контроллер имеет какой-либо «популярный» интерфейс, то в большинстве случаев он может быть подключен и к другой «популярной» сети посредством специальных переходных устройств.

Дополнительными критериями при выборе контроллеров являются:

Быстродействие и объем памяти. Эти параметры, как правило, увязываются производителями с числом точек ввода-вывода, так как большая часть ресурсов микропроцессорной системы отводится на работу с УСО, а вычислительные алгоритмы в большинстве случаев не очень сложны. Быстродействие контроллеров обычно оценивается временем цикла: опрос датчиков – вычисление управляющих сигналов – их реализация. Это время составляет от единиц миллисекунд до нескольких секунд. Для большинства промышленных применений такое быстродействие является достаточным. В «крайних» случаях для повышения времени реакции системы применяется специальное программное обеспечение реального времени (большие централизованные системы) или специальные аппаратные решения, например, функциональные модули FM (управ-

ление быстродействующим позиционным приводом). Объем памяти для промышленных контроллеров, как правило, очень небольшой: от десятков КБ до нескольких МБ.

Еще один критерий – возможность работать в опасных зонах как и для других устройств определяется степенью защиты корпуса (для приборных исполнений).

В Европе для обозначения степени защиты корпусов от пыли и влаги применяется так называемая система IP–кодов, определяемая стандартом МЭК 529 (IEC 529) Международной Электротехнической Комиссии (International Electrotechnical Commission). Ему полностью соответствует ГОСТ 14254_80 «Изделия электротехнические. Оболочки. Степени защиты. Обозначения. Методы испытаний». В Европейском сообществе стандарту МЭК 529 соответствует стандарт EN 60529. Согласно стандарту МЭК 529, степень защиты корпуса обозначается латинскими буквами IP и следующими за ними двумя цифрами, например IP 54. При этом первая цифра обозначает степень защиты персонала от находящихся под напряжением или движущихся частей внутри корпуса, а также степень защиты изделия от попадания внутрь твердых посторонних тел, в частности, пыли. Вторая цифра означает степень защиты изделия от попадания внутрь воды.

Степень защиты	Защита от твердых тел	Защита от воды
0	Защита отсутствует	Защита отсутствует
1	Защита от проникновения внутрь оболочки большого участка поверхности человеческого тела, например рук, и от проникновения твердых тел диаметром более 50 мм	Капли воды, вертикально падающие на оболочку, не должны оказывать вредного воздействия на изделие
2	Защита от проникновения внутрь корпуса пальцев или предметов длиной более 80 мм и от проникновения твердых тел диаметром более 12 мм	Капли воды, падающие на оболочку под углом до 15° от вертикали, не должны оказывать вредного воздействия на изделие
3	Защита от проникновения внутрь оболочки инструментов, проволоки, твердых тел и т. п. диаметром или толщиной более 2,5 мм	Дождь, падающий на оболочку под углом 60° от вертикали, не должен оказывать вредное воздействие на изделие
4	Защита от проникновения внутрь оболочки проволоки и твердых тел диаметром более 1,0 мм	Вода, разбрызгиваемая на оболочку в любом направлении, не должна оказывать вредно-

		го воздействия на изделие
5	Проникновение внутрь корпуса пыли не предотвращено полностью, однако количество проникающей пыли не может нарушить работу изделия	Струя воды, выбрасываемая в любом направлении на оболочку, не должна оказывать вредного воздействия на изделие
6	Проникновение пыли предотвращено полностью	Сильная струя воды (100 л/мин при давлении 100 кПа) или волны воды не должны вызывать попадание в оболочку воды в количестве, достаточном для повреждения изделия
7	Не предусмотрено	Вода не должна проникать в оболочку, погруженную в воду на глуби, ну примерно 15 см, при примерном равенстве температуры оболоч, ки и воды, в количестве, достаточном для повреждения изделия
8	Не предусмотрено	Изделие пригодно для длительного погружения в воду при условиях, устанавливаемых изготовителем

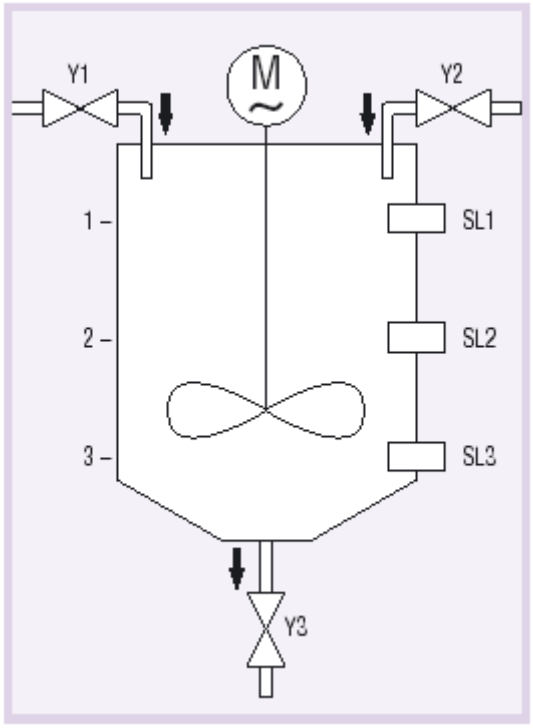
Таким образом, корпус со степенью защиты IP 54, упоминавшейся нами ранее, обеспечивает защиту от пыли, не исключая, правда, ее ограниченного проникновения, а также обеспечивает полную брызгозащиту, но не выдерживает попадание струй воды. Корпуса с подобной степенью защиты наиболее приспособлены к использованию в условиях промышленного производства, но не на улице. Для постоянного использования вне помещений необходимо применять корпуса со степенью защиты не менее IP 65, то есть обеспечивающие полную защиту от пыли и струй воды.

Примеры программируемых контроллеров

1. Контроллеры для малой автоматизации (управление отдельной установкой или производственной линией). Решение задач программно-логического управления.

Контроллер LOGO! Фирмы Siemens

Рассмотрим сравнительно простую и вместе с тем достаточно типичную схему смесительной установки, приведенную на рис. Алгоритм работы установки очень прост. После её запуска в работу открывается клапан Y1, и ёмкость начинает заполняться компонентом № 1. При достижении уровня 2 срабатывает датчик SL2, закрывается клапан Y1 и открывается клапан Y2. В ёмкость начинает поступать компонент № 2. После заполнения ёмкости до уровня 1 по сигналу от датчика SL1 закрывается клапан Y2 и включается привод мешалки M. Через 15 минут он выключается, смесь готова. Для её выгрузки открывается клапан Y3. Окончание процесса фиксируется датчиком SL3 (уровень 3). После закрытия клапана Y3 установка готова к новому циклу приготовления смеси.



Как поступил бы разработчик системы управления, скажем, лет 10-15 назад? Не мудрствуя лукаво, он использовал бы схему, подобную той, что приведена на рис. 2. Вся система реализуется с помощью четырёх промежуточных реле и одного реле времени.

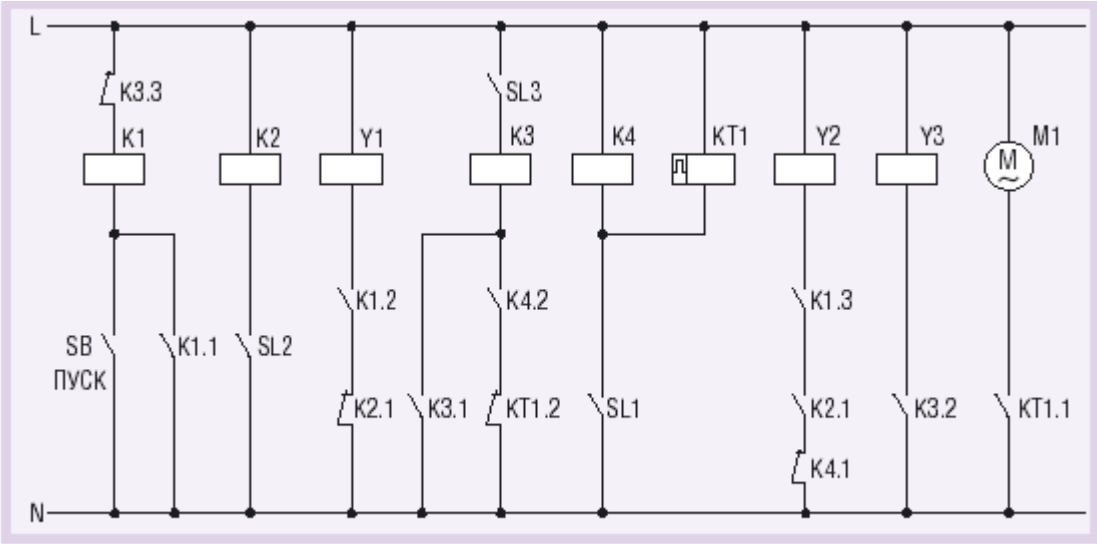


Рис. 2. Электрическая схема системы управления на реле

Работа схемы начинается с нажатия кнопки SB «Пуск». Катушка реле K1 получает питание (контакт K3.3 замкнут). При этом замыкаются контакты:

– K1.1, шунтируя кнопу SB «Пуск»;

– K1.2, который включает электромагнитный клапан Y1;

– K1.3, подготавливая цепь Y2 к работе.

Начинается подача ингредиента №1. При достижении уровня 2 срабатывает датчик уровня SL2, через его контакт подается питание на катушку промежуточного реле K2. Реле K2:

– размыкает нормально закрытый контакт K2.1 в цепи электромагнитного клапана Y1, последний закрывается и подача ингредиента №1 прекращается;

– замыкает нормально открытый контакт в цепи электромагнитного клапана Y2, последний открывается и начинается подача ингредиента №2 (контакт K1.3 уже замкнут).

При достижении уровня 1 срабатывает датчик уровня SL1, через его контакт подается питание на катушки промежуточного реле K4 и реле времени KT1. Реле K4:

– размыкает нормально закрытый контакт в цепи Y2, выключая подачу ингредиента №2;

– замыкает нормально открытый контакт в цепи катушки промежуточного реле K3, подготавливая его к включению (контакт датчика уровня SL3 давно уже замкнут).

Реле времени KT1, получая питание, мгновенно:

– замыкает контакт KT1.1 в цепи привода мешалки M. Мешалка включается;

– размыкает контакт KT1.2 в цепи катушки реле K3, не допуская его включения;

– начинает выдержку времени.

По истечении заданного времени контакт KT1.1 размыкается и мешалка останавливается, KT1.2 замыкается и на катушку промежуточного реле K3 подается напряжение. Срабатывая, реле K3

– замыканием контакта K3.1 шунтирует контакт K4.2, не позволяя последнему обесточить реле при падении уровня смеси ниже уровня 1;

– контактом K3.2 включает электромагнитный клапан Y3. Начинается слив смеси;

– контактом K3.3 обесточивает катушку реле K1, не допуская тем самым включения клапанов Y1 и Y2 во время слива.

При достижении уровнем смеси уровня 3, датчик уровня SL3 размыкает свой контакт в цепи катушки K3, последняя теряет питание, клапан Y3 закрывается и схема приходит в первоначальное состояние.

Попробуем воссоздать приведенную выше схему, анализируя описанный выше процесс.

Вся последовательность действий делится на две фазы:

– приготовление смеси;

– слив смеси.

Фаза приготовления начинается только если закончен слив нажатием кнопки «Пуск»:

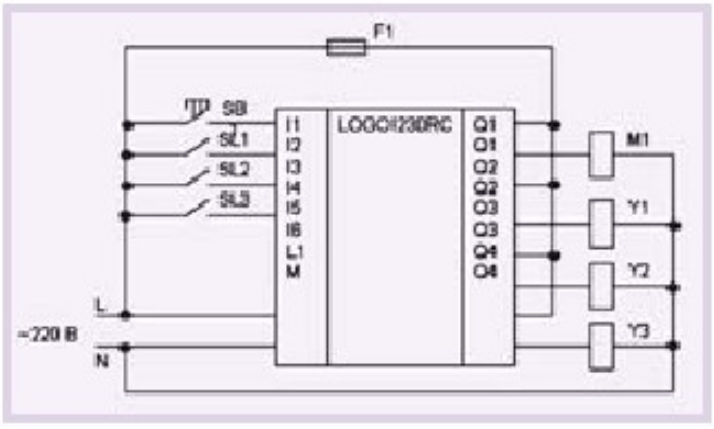
Рассмотрим реализацию описанного алгоритма в контроллере LOGO!.

Краткая характеристика контроллеров LOGO! (по документации)

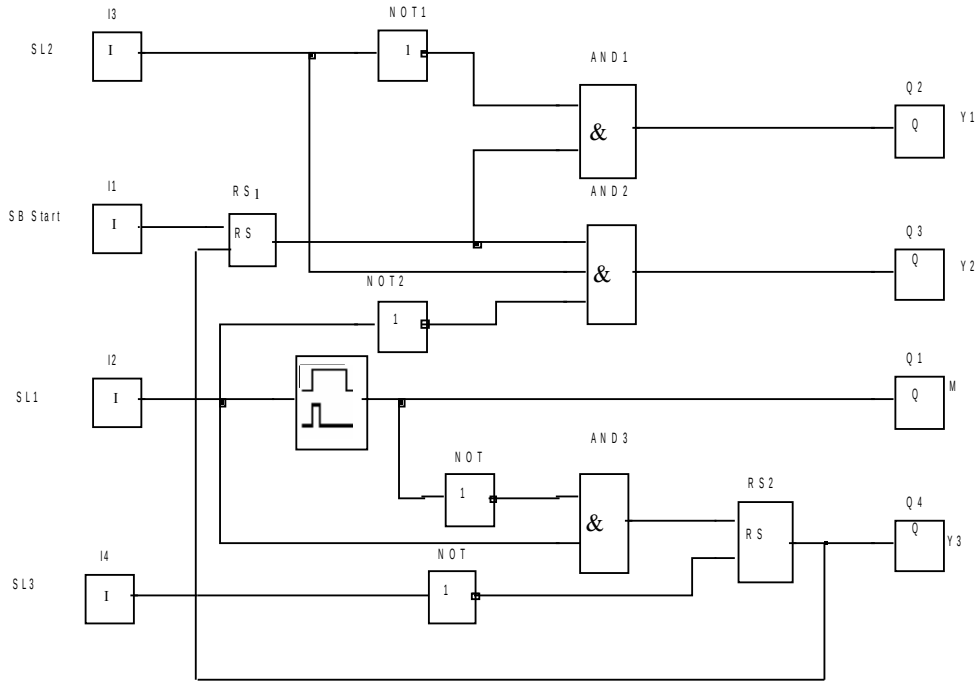
1. Ввод-вывод;
2. Программирование;
3. Цена: контроллеров 130 – 150 евро, модулей ввода вывода 20 – 130 евро (shop.elektrostyle.ru).

Решение задачи автоматизации работы мешалки с помощью LOGO!.

Схема подключения контактов датчиков уровня, электромагнитов клапанов и пускателя мешалки к контроллеру LOGO! :



Алгоритмическая схема:



В алгоритмической схеме использованы:

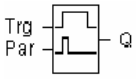
– входы I (нумерация полностью соответствует схеме подключения);

– выходы Q (нумерация полностью соответствует схеме подключения);

– логические (основные) функции НЕ, И;

– самоблокирующееся реле RS: вход S устанавливает выход Q, вход R снова сбрасывает выход Q;

– интервальное реле (вывод импульса): входной импульс вызывает появление сигнала заданной длительности на выходе:

Символ в LOGO!	Подключение	Описание
	Вход Trg	Сигнал на входе Trg (trigger = запустить) запускает отсчет времени для интервального реле.
	Параметр	T – это время, через которое выключается выход (выходной сигнал переключается с 1 на 0). Сохраняемость: / = Сохраняемость отсутствует R = Состояние сохраняется.
	Выход Q	Q включается одновременно с Trg и остается включенным в течение времени Ta, если входной сигнал остается равным 1.

Преимуществом данного решения является возможность продолжения работы после временного пропадания напряжения питания благодаря сохраняемости выходов алгоритмов RS. Релейная схема в случае отключения питания на этапе слива теряет работоспособность.

В контроллерах LOGO! отсутствует возможность программирования на процедурном языке.

Гипотетически процедурный вид программы на языке типа C/C++ может быть примерно таким:

```
while (!SL3) // пока уровень ниже отметки 3
{
  if (SB)    // кнопка пуск нажата
  {
        Y1 = 1;       // открытие клапана Y1
        while (!SL2); // пока уровень ниже отметки 2
        Y1 = 0;    // закрытие клапана Y1
        Y2 = 1;       // открытие клапана Y2
        while (!SL1); // пока уровень ниже отметки 1
        Y2 = 0;    // закрытие клапана Y2
        M = 1;    // включение мешалки
        delay(…);  //временная задержка
        M = 0;    // выключение мешалки
        Y3 = 1; // открытие клапана Y3
        while (SL3); // пока уровень выше отметки 3
        Y3 = 0; // закрытие клапана Y3
  }
}
```

Очевидное преимущество такой программы – ее простота и доступность для понимания. Недостаток – отсутствие возможности восстановления режима работы после кратковременного сбоя (вызванного любыми причинами). Программа «не помнит», в каком месте она была прервана. Кроме того, сервисные функции (диагностика) могут выполняться только по прерываниям.

2. Контроллеры для автоматизации систем средней и большой сложности (управление технологическим и производственным процессами). Программно-логическое управление и регулирование.

SIMATIC S7-200 – семейство микроконтроллеров

Программируемые контроллеры SIMATIC S7-200 предназначены для построения относительно простых систем автоматического управления, отличающихся минимальными затратами на приобретение аппаратуры и разработку системы. Контроллеры способны работать в реальном масштабе времени и могут быть использованы как для построения узлов локальной автоматики, так и узлов, поддерживающих интенсивный коммуникационный обмен данными че-

рез сети Industrial Ethernet, PROFIBUS-DP, MPI, AS-Interface, MPI, PPI, MODBUS, а также через модемы.

Программируемые контроллеры SIMATIC S7-200 имеют ряд международных сертификатов, а также

– сертификат Госстандарта России № РОСС DE.АЯ46.В61141 от 14.03.2003г., подтверждающий соответствие требованиям стандартов ГОСТ Р 50377-92 (стандарт в целом), ГОСТ29125-91 (п.2.8), ГОСТ 26329-84 (п.п. 1.2, 1.3), ГОСТ Р51318.22-99, ГОСТ 51318.24-99;

– метрологический сертификат Госстандарта России № 11991 от 4.04.2002г;

– экспертное заключение о соответствии функциональных показателей интегрированной системы автоматизации SIMATIC S7 отраслевым требованиям и условиям эксплуатации энергопредприятий РАО "ЕЭС России".

Программируемые контроллеры SIMATIC S7-200 характеризуются следующими показателями:

– программирование на языках STL, LAD и FBD;

– высокое быстродействие. Время выполнения 1К логических инструкций не превышает 0.22мс;

– наличие конфигурируемых реманентных областей памяти для необслуживаемого сохранения данных при перебоях в питании контроллера;

– 3-уровневая парольная защита программы пользователя;

– универсальность входов и выходов центральных процессоров: стандартные дискретные входы и выходы, входы скоростного счета, импульсные выходы;

– наращивание количества обслуживаемых входов и выходов за счет использования модулей расширения и/или систем распределенного ввода-вывода на основе AS-Interface;

– универсальность встроенного интерфейса центральных процессоров: поддержка протоколов PPI/ MPI/ USS/ MODBUS, свободно программируемый порт;

– наличие съемных терминальных блоков для подключения внешних цепей, упрощающих выполнение операций монтажа и замены вышедших из строя модулей;

– поддержка обработки рецептурных данных;

– использование картриджа памяти для регистрации данных и сохранения электронных версий технической документации;

– возможность редактирования программы без перевода центрального процессора в режим STOP.

– использование страничной адресации блоков данных.

Аппаратура SIMATIC S7-200

Семейство объединяет в своем составе модули центральных процессоров; коммуникационные модули; модуль позиционирования EM 253; модули ввода-вывода дискретных и аналоговых сигналов; модули блоков питания. Все модули способны работать в диапазоне температур от 0 до +55°С. Для более

жестких условий эксплуатации могут использоваться модули семейства SIPLUS S7-200 с диапазоном рабочих температур от -25 до +70°C.

Конструктивные особенности:

– компактные пластиковые корпуса со степенью защиты IP20.

– простое подключение внешних цепей через терминальные блоки с контактами под винт. Защита всех токоведущих частей открывающимися пластиковыми крышками;

– наличие штатных или опциональных съемных терминальных блоков, позволяющих выполнять замену модулей без демонтажа их внешних цепей;

– монтаж на стандартную 35мм профильную шину или на плоскую поверхность с креплением винтами;

– соединение модулей с помощью плоских кабелей, вмонтированных в каждый модуль расширения.

Центральные процессоры

В S7-200 используется 5 моделей центральных процессоров, отличающихся объемами встроенной памяти, количеством и видом встроенных входов и выходов, количеством встроенных интерфейсов RS 485, количеством потенциометров аналогового задания цифровых величин и другими показателями. Каждая модель имеет две модификации:

– с напряжением питания =24В и дискретными выходами =24В/0.75А на основе транзисторных ключей;

– с напряжением питания ~115/230В и дискретными выходами в виде замыкающих контактов реле с нагрузочной способностью до 2А на контакт.

Встроенный интерфейс RS 485 (один или два) используется:

без дополнительного программного обеспечения:

- для программирования контроллера,

- для включения контроллера в сети PPI ((Programmable Peripheral Interface) программируемый интерфейс периферийных устройств используется для связи с медленными периферийными устройствами. Часто реализовывался на микросхеме Intel 8255) или MPI ((Multibus II Peripheral Interface) периферийный интерфейс шины Multibus II) со скоростью передачи данных до 187.5 Кбит/с,

- в качестве свободно программируемого порта с поддержкой ASCII протокола и скоростью передачи данных до 38.4 Кбит/с;

с дополнительным программным обеспечением Instruction Library:

- для поддержки протокола USS со скоростью передачи данных до 19.2 Кбит/с и возможностью подключения до 30 преобразователей частоты (например, преобразователей серий MICROMASTER или SINAMICS),

- для поддержки протокола MODBUS RTU и работы в режиме ведомого сетевого устройства.

Все центральные процессоры оснащены встроенным блоком питания =24В для питания датчиков или другой нагрузки.

Дискретные входы всех центральных процессоров рассчитаны на входное напряжение =24В.

Коммуникационные модули

– CP 243-1: для подключения к сети Industrial Ethernet, 10/100 Мбит/с, TCP/IP;

– CP 243-1 IT: для подключения к сети Ethernet, 10/100 Мбит/с, TCP/IP. Поддержка функций HTTP/FTP-сервера, FTP-клиента. Flash память объемом 8 Мбайт для хранения файловой системы;

– CP 243-2: коммуникационный процессор ведущего устройства AS-Interface, способный обслуживать до 62 ведомых устройств;

– EM 277: для подключения к сети PROFIBUS-DP и выполнения функций ведомого устройства, до 12 Мбит/с.

– EM 241: модем для непосредственного соединения двух S7-200 через телефонную сеть, передачи SMS-сообщений, поддержки функций ведущего/ ведомого устройства MODBUS.

Модуль позиционирования EM 253

Для решения простых задач позиционирования по одной оси сервоприводов или приводов с шаговыми двигателями.

Модули ввода-вывода дискретных и аналоговых сигналов

С помощью модулей ввода-вывода программируемые контроллеры S7-200 легко адаптируются к требованиям решаемой задачи.

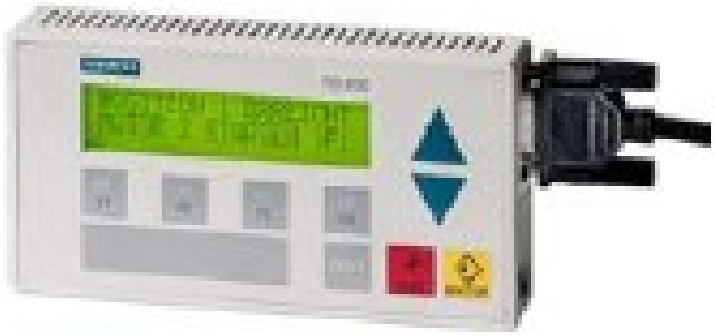
Они позволяют увеличивать количество входов и выходов, обслуживаемых одним центральным процессором, дополнять систему ввода-вывода не только дискретными, но и аналоговыми каналами с требуемыми параметрами входных и выходных сигналов.

Модули ввода-вывода дискретных сигналов	EM 221	8 DI =24В
		8 DI ~120/230В
		16 DI =24В
	EM 222	4 DO =24В/5А
		4 DO (реле), до 10А на выход
		8 DO =24В/0.75А
		8 DO ~120/230В/0.5А
		8 DO (реле), до 2А на выход
	EM 223	4 DI =24В + 4 DO =24В/0.75А
		4 DI =24В + 4 DO (реле), до 2А на выход
		8 DI =24В + 8 DO =24В/0.75А
		8 DI =24В + 8 DO (реле), до 2А на выход
		16 DI =24В + 16 DO =24В/0.75А
		16 DI =24В + 16 DO (реле), до 2А на выход
Модули ввода-вывода аналоговых сигналов	EM 231	2 AI Pt100/200/500/1000/10000, Ni100/120/1000, Cu10, 150/300/600 Ом
		4 AI 0…5В/0…10В/±2.5В/±5В/0…20мА, 12 бит, 250 мкс
		4 AI ±80мВ, термопары типов J/K/S/T/R/E/N, 15 бит + знак
	EM 232	2 AO ±10В/0…20мА, 12 бит
	EM 235	4 AI ±10В/0…20мА + 1 AO ±10В/0…20мА
Коммуникационные модули	CP 243-1	Коммуникационный процессор для подключения к Industrial Ethernet
	CP 243-1 IT	Коммуникационный процессор для подключения к Industrial Ethernet
	CP 243-2	Коммуникационный процессор ведущего устройства AS-Interface
	EM 241	Модем, 300 бод … 33.6 Кбод, SMS, MODBUS
	EM 277	Модуль ведомого устройства PROFIBUS-DP, до 12 Мбит/с
Модуль позиционирования	EM 253	Для позиционирования приводов с серво- или шаговыми двигателями по 1 оси, частота следования выходных импульсов 12Гц … 200кГц

Аппаратура человеко-машинного интерфейса

Для решения задач человеко-машинного интерфейса в системах управления на основе программируемых контроллеров S7-200 может использоваться практически весь спектр продуктов семейства SIMATIC HMI. В то же время в состав этого семейства входит целый ряд текстовых дисплеев и панелей оператора, предназначенных для работы только с контроллерами S7-200.

Текстовые дисплеи SIMATIC TD 200/TD 200C



Текстовый дисплей TD 200 может быть использован со всеми контроллерами семейства SIMATIC S7-200. Он подключается к контроллеру соединительным кабелем через PPI интерфейс. При длине линии связи до 2.5м через этот же кабель осуществляется питание дисплея. При длине линии более 2.5м для питания дисплея необходим дополнительный блок питания =24В. К одному контроллеру может быть подключено несколько дисплеев TD200. Дисплей позволяет:

– отображать до 80 текстовых сообщений, в каждое из которых допускается включать до 6 переменных;

– отображать и модифицировать значения параметров со встроенной клавиатуры;

– управлять состоянием входов и выходов контроллера для реализации функций ручного управления, тестирования и диагностики системы.

Конфигурирование текстового дисплея TD 200 производится с помощью специального мастера пакета STEP 7 Micro/Win.

Дополнительного программного обеспечения не требуется. Параметры конфигурации сохраняются в памяти центрального процессора S7-200.

Панель оператора SIMATIC OP 73 Micro

Панель оператора OP 73 Micro оснащена графическим монохромным дисплеем с разрешением 160x48 точек, 8 системными и 4 свободно конфигурируемыми функциональными клавишами. Подключение к центральному процессору S7-200 выполняется через PPI интерфейс. Она позволяет:

– использовать поля ввода-вывода для отображения значений параметров и модификации переменных;

– конфигурировать до 16 функций, запускаемых с помощью функциональных клавиш;

– использовать точечные графические изображения для оформления экрана и маркировки клавиш и кнопок;

– использовать масштабируемые шрифты;

– формировать текстовые сообщения и тексты подсказок на различных языках, выбираемых из списка 32 поддерживаемых языков;

– поддержку парольного доступа к системе управления;

– обслуживать до 250 сообщений и до 250 экранных изображений.

– выполнять мониторинг граничных значений входных и выходных параметров и т.д.

Конфигурирование панели выполняется с помощью программного обеспечения SIMATIC WinCC flexible Micro.

Сенсорные панели SIMATIC TP 177Micro

Панель оператора TP 177 Micro оснащена 5.7’ графическим дисплеем голубого свечения с разрешением 320x240 точек или 240x320 точек и сенсорной резистивной аналоговой клавиатурой. Подключение к S7-200 выполняется через PPI интерфейс центрального процессора. По основному набору функций человеко-машинного интерфейса она близка к панели OP 73 Micro, но имеет лучшие графические возможности и позволяет обслуживать до 500 сообщений и до 250 экранных изображений.

Программное обеспечение

Основной набор стандартных инструментальных средств для работы с программируемыми контроллерами SIMATIC S7-200 сконцентрирован в пакете **STEP 7 Micro/WIN**. Пакет позволяет:

– программировать контроллеры на языках LAD, FBD и STL, выполнять автономную или интерактивную отладку программы;

– выполнять настройку параметров аппаратуры;

– использовать символьную адресацию;

– использовать широкий набор мастеров для конфигурирования коммуникационных модулей, модуля позиционирования, текстовых дисплеев TD 200/TD 200C, ПИД-регуляторов, скоростных счетчиков и импульсных выходов, встроенных интерфейсов, управления рецептурными данными и т.д.;

– выполнять удобный просмотр всех данных проекта;

– загружать необходимые данные в опциональный картридж памяти и т.д.

Оболочка пакета **STEP 7 Micro/WIN** переведена на русский язык.

Пакет **S7-200 PC Access** обеспечивает возможность организации обмена данными между компьютерными приложениями и центральными процессорами или коммуникационными модулями программируемого контроллера S7-200 через OPC интерфейс. Для организации обмена данными могут использоваться любые варианты связи, поддерживаемые контроллером S7-200. К одному компьютеру может подключаться не более 8 контроллеров

Micro/WIN Instruction Library является опциональным пакетом, интегрируемым в среду STEP 7 Micro/WIN от V3.2 и выше.

Он содержит библиотеку функциональных блоков, позволяющих использовать встроенный интерфейс центрального процессора S7-S7-200 для под-

держки USS протокола или протокола MODBUS RTU в режиме ведомого устройства.

Тема № 2 Схемотехника блоков питания (2 часа)

Блоки питания аппаратуры, предназначенные для питания от сети переменного тока, в зависимости от назначения и мощности могут быть выполнены по различным схемам. Схема простейшего *блока питания с трансформаторным входом* приведена на рис. 3.1.

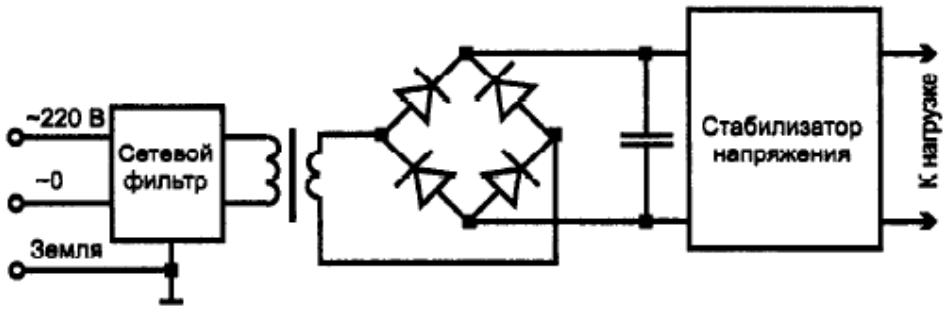


Рис. 3.1. Блок питания с трансформаторным входом

Здесь понижающий трансформатор, работающий на частоте питающей сети 50/60 Гц, обеспечивает требуемое напряжение и гальваническую развязку питаемых цепей от сети переменного тока. Выходное напряжение может стабилизироваться непрерывным или импульсным низковольтным стабилизатором напряжения. Главный недостаток такого блока — большие габариты низкочастотного силового трансформатора. Трансформатор блока питания, рассчитанный на частоту 60 Гц (зарубежные питающие сети), на частоте 50 Гц (наши сети) может ощутимо нагреваться. Естественно, от сети постоянного тока (такие изредка встречаются) такой блок работать не может. Блоки питания с трансформаторным входом применяются при небольшой выходной мощности, чаще всего — в выносных адаптерах (старых моделей), обеспечивающих питание модемов, хабов и прочих маломощных устройств внешнего исполнения. Такие блоки довольно часто монтируются прямо на вилке питания.

В *блоках питания с бестрансформаторным входом* понижающий трансформатор работает на высокой частоте — в десятки и даже сотни килогерц, что позво-

ляет уменьшить габариты и вес блока питания. В этом случае входное напряжение сразу выпрямляется и после фильтрации поступает на высокочастотный преобразователь. Высокочастотные импульсы преобразователя попадают на понижающий импульсный трансформатор, который обеспечивает гальваническую развязку выходных и входных цепей. Преобразователь чаще всего делают управляемым, так что на него возлагаются еще и функции регулирующего элемента стабилизатора напряжения. Управляя шириной импульса, можно изменять величину энергии, поступающей через трансформатор в выпрямитель, и, следовательно, регулировать (стабилизировать) его выходное напряжение. В зависимости от мощности стабилизатор строится по однотактной или двухтактной схеме. Однотактная схема несколько проще (рис. 3.2), ее применяют в блоках питания, где мощность обычно не превышает сотни ватт (например, в мониторах). В мониторах частоту импульсного блока обычно синхронизируют с частотой генератора строчной развертки во избежание видимых помех. В настоящее время выпускается широкий ассортимент управляющих микросхем со встроенным ключевым транзистором и развитыми функциями защиты и управления. Блоки питания на их основе получаются предельно простыми и компактными; маломощные блоки могут размещаться прямо в вилках-адаптерах.

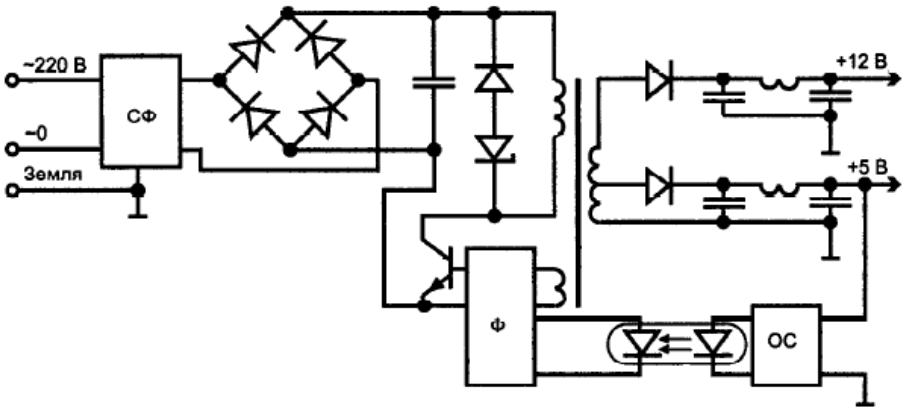


Рис. 3.2. Однотактный блок питания (СФ – сетевой фильтр, Ф – формирователь импульсов, ОС – усилитель обратной связи)

Двухтактные преобразователи сложнее, но они обеспечивают большую выходную мощность. Такие блоки широко используются в блоках питания PC (см. 3.2).

Если блок питания должен вырабатывать несколько выходных напряжений, сам преобразователь может стабилизировать лишь одно из них. Остальные напряжения могут быть стабилизированы дополнительными выходными стабилизаторами, но часто их оставляют нестабилизированными. При этом появляется взаимозависимость: чем больше нагрузка по основной (стабилизированной) цепи, тем выше напряжения на остальных шинах.

Импульсные блоки питания имеют малые габариты, но компактный трансформатор представляет собой довольно сложное изделие. Импульсные помехи, которые могут проникать как в питаемые, так и в питающие цепи, подавляют тщательно разработанными фильтрами. Внешнее излучение подавляется металлическим экраном, в который заключают весь блок.

Импульсные блоки питания не критичны к частоте сети (50 или 60 Гц), могут работать от постоянного тока и часто в широком диапазоне входных напряжений. Современные блоки, у которых указано свойство Autoswitching Power Supply, работают в диапазоне 110-230 В без переключателя напряжения. Такие блоки применяются в большинстве современных мониторов.

Самый тяжелый режим функционирования элементов блока питания возникает в момент включения. После выключения блока питания (любой конструкции) включать его повторно рекомендуется не раньше, чем через 10 с. Несоблюдение этой рекомендации может сократить жизнь блока питания.

Наличие выпрямителя и накопительного конденсатора на входе бестрансформаторного блока питания обусловливает ярко выраженную динамическую нелинейность входной цепи. На рис. 3.3 приведены осциллограммы напряжения сети и потребляемого тока, которые иллюстрируют эту нелинейность. Пока мгновенное значение напряжения ниже напряжения на накопительном конденсаторе выпрямителя, ток практически не потребляется. На верхушках синусоиды ток резко возрастает, так что в его спектре очень сильно выра-

жена 3-я гармоника. Для питающей сети такой характер нагрузки нежелателен, но с ним приходится мириться. Конечно, нелинейность имеется и в трансформаторном блоке питания, но она несколько сглаживается низкочастотным трансформатором.

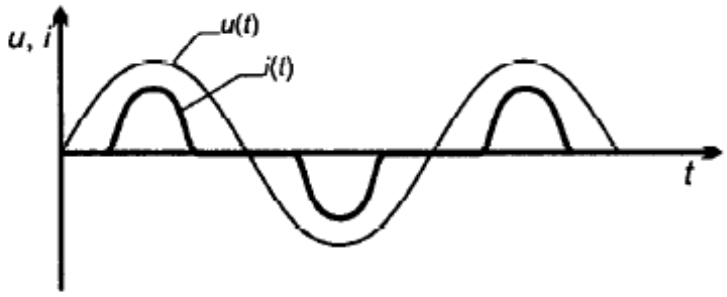


Рис. 3.3. Нелинейность входной цепи бестрансформаторного блока питания

Тема №3 ЦАП АЦП (4 часа)

ЦАП

ЦАП служит для преобразования цифровой информации в аналоговую форму, т.е. выходной сигнал ЦАП в общепринятых единицах измерения тока или напряжения (мВ, В, мА) соответствует численному значению входной кодовой комбинации.

Например, при подаче на вход ЦАП кодовой комбинации (в десятичном эквиваленте) равной 150 на его выходе при этом имеется напряжение 1500 мВ, это значит, что изменение значения входной кодовой комбинации (входного числа) на единицу приводит к изменению выходного напряжения на 10 мВ. В этом случае мы имеем ЦАП с шагом преобразования цифровой информации 10 мВ. Величина напряжения, соответствующая одной единице цифровой информации, называется шагом квантования $\Delta u_{кв}$. При подаче на вход ЦАП последовательной цифровой комбинации, меняющейся от 0 до N, на его выходе появится ступенчато-нарастающее напряжение (рис. 5.1). Высота каждой ступени соответствует одному шагу квантования $\Delta u_{кв}$.

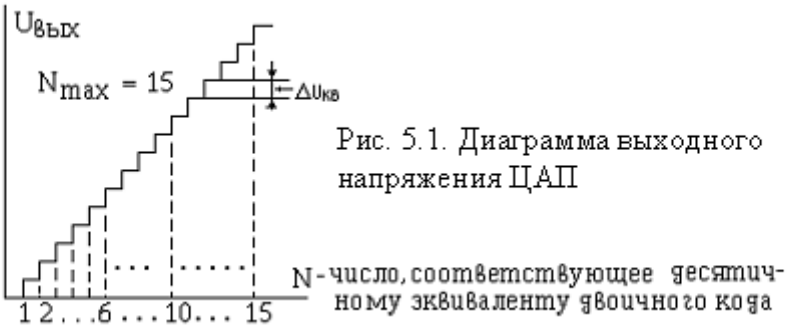


Рис. 5.1. Диаграмма выходного напряжения ЦАП

Если число входной кодовой комбинации соответствует N, то выходное напряжение $U_{вых\ ЦАП} = N\Delta \times u_{кв}$. Таким образом, можно вычислить значение выходного напряжения для любой входной кодовой комбинации. Нетрудно убедиться в том, что $\Delta u_{кв}$ является масштабным коэффициентом преобразователя, имеющим размерность тока или напряжения (так как цифровая комбинация на входе ЦАП размерности не имеет). Обычно, значение $\Delta u_{кв}$ выбирают кратным десяти, что облегчает процесс пересчета соответствия преобразованного и исходного сигналов. Так как $\Delta u_{кв}$ определяет минимальное значение выходного напряжения аналогового сигнала $U_{вых\ мин.} = \Delta u_{кв}$, при выборе его значения необходимо учитывать также шумовые факторы, погрешности усиления масштабирующих усилителей и компаратора.

Основные параметры ЦАП. Точность преобразования и качество работы ЦАП характеризуют следующие параметры: относительная разрешающая способность, абсолютная разрешающая способность, абсолютная погрешность преобразования, нелинейность преобразования, дифференциальная нелинейность, скорость преобразования (время одного преобразования) и максимальная частота преобразования.

1. **Относительная разрешающая способность**

$$\delta_o = \frac{1}{2^n - 1},$$

здесь n- количество разрядов двоичного числа, подаваемого на вход АЦП (n - соответствует числу разрядных входов ЦАП). Относительная разрешающая способность - это обратная величина от максимального числа уровней квантования.

2. **Абсолютная разрешающая способность**

$$\delta_a = \frac{U_{пш}}{2^n - 1} = \Delta u_{кв},$$

где $U_{пш}$ - напряжение полной шкалы, соответствующее опорному напряжению ЦАП. Это напряжение можно считать равным максимальному выходному напряжению; $2^n - 1 = N$ - количество ступеней квантования.

Численно абсолютная разрешающая способность равна шагу квантования $\Delta u_{кв}$.

3. **Абсолютная погрешность преобразования** $\delta_{пш}$ показывает максимальное отклонение выходного напряжения $U_{вых}$ в точке пересечения с идеальной характеристикой (прямой) на уровне напряжения полной шкалы (рис.5.2). Абсолютная погрешность преобразования оценивается в процентах или же в единицах младшего значащего разряда (МР). При оценке значения абсолютной погрешности преобразования знак напряжения не учитывается.

4. **Нелинейность преобразования** ЦАП $\delta_{лн}$ определяет максимальное отклонение реальной характеристики от идеальной (рис. 5.2) и оценивается также в процентах или в единицах младшего значащего разряда.

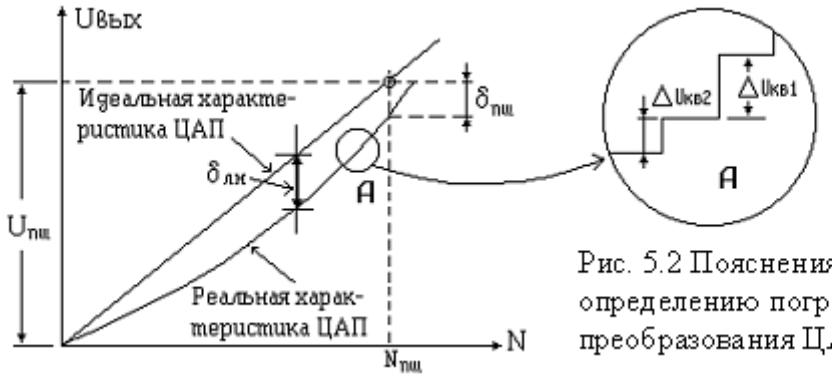


Рис. 5.2 Пояснения к определению погрешностей преобразования ЦАП

5**. Дифференциальная нелинейность преобразования** ЦАП $\delta_{дф.лн}$ численно равна максимальной разности двух соседних приращений (шагов квантования)

$$\delta_{дф.лн} = \Delta u_{кв\,1} - \Delta u_{кв2}.$$

Дифференциальная нелинейность оценивается в младших значащих разрядах и обычно не превышает нескольких единиц МР.

Младший значащий разряд численно определяет минимальное значение выходного напряжения, т.е. квант напряжения. Для оценки дифференциальной нелинейности $\delta_{дф.лн}$ в процентах можно воспользоваться выражением

$$\delta\% = \frac{\delta_{м\,зр} \cdot \Delta u_{к\,в}}{U_{п\,ш}} \cdot 100\%.$$

Время установления выходного напряжения или тока $t_{уст}$ - интервал времени от подачи входного двоичного входного кода до вхождения выходного сигнала в заданные пределы.

Максимальная частота преобразования $f_{пр}$ - наибольшая частота дискретизации, при которой параметры ЦАП соответствуют заданным значениям. Максимальная частота и время установления определяют быстродействие ЦАП.

Виды ЦАП условно можно разделить на две группы: с резисторными матрицами, безматричные ЦАП. В интегральном исполнении применяются только ЦАП с прецизионными резисторными матрицами, формирующими выходные сигналы путем суммирования токов.

ЦАП содержит элементы цифровой и аналоговой схемотехники. В качестве аналоговых элементов используются операционные усилители, аналоговые ключи (коммутаторы), резисторные матрицы и т.д.

Аналоговые элементы, входящие в состав ЦАП, практически полностью определяют его качественные и эксплуатационные параметры, основную роль при этом играют точность подбора номиналов резисторов резисторной матрицы и параметров операционного усилителя (ОУ).

Операционный усилитель представляет собой усилитель постоянного тока, имеющий коэффициент усиления по напряжению более тысячи. Он имеет дифференциальный входной каскад, т.е. имеет два входа: инвертирующий и неинвертирующий.

Своему названию ОУ “обязан” аналоговым вычислительным машинам, так как первоначально он был ориентирован на моделирование различных математических операций. Появление ОУ в виде интегральных микросхем привело к быстрому росту популярности ОУ в реализации аналоговой и гибридной электронной схемотехники. Условное обозначение ОУ показано на рис. 5.3.

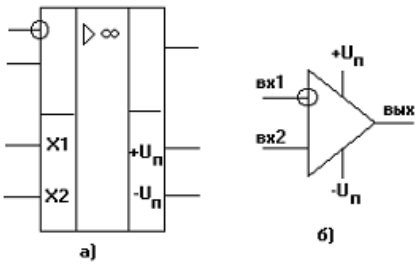


Рис. 5.3. Условные обозначения операционного усилителя: а) – стандартное, по ГОСТу; б) – для рабочей документации

Благодаря большому коэффициенту усиления (современные ОУ имеют коэффициент усиления К=10^5 ...10^6) и малым входным токам, усилители, построенные на базе ОУ, обладают уникальными свойствами. В частности, параметры многих устройств определяются только внешними цепями - цепями обратной связи, соединяющими выход ОУ с его входом. Например, коэффици-

ент усиления усилителя, схема которого показана на рис. 5.4 (а), определяется с высокой точностью отношением сопротивлений двух резисторов $K=-R_{oc}/R$.

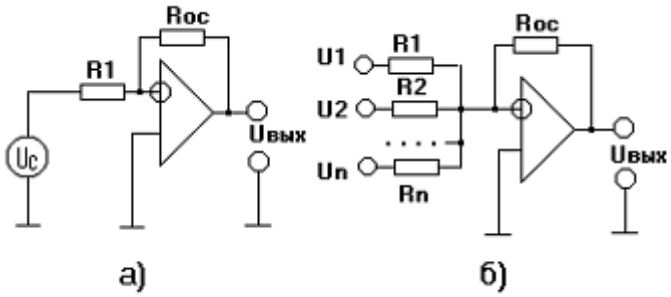


Рис. 5.4. Схемы усилителей на ОУ: а – инвертирующий; б – суммирующий усилитель

Если на инвертирующий вход усилителя на ОУ подать сигнал от нескольких источников (рис. 5.4, б), то выходной сигнал определяется как произведение суммы входных токов на величину сопротивления резистора обратной связи

$$U_{вых} = -R_{oc}(I_{вх1}+I_{вх2}+. . . . +I_{вх.n}).$$

Входной ток от каждого источника определяется как отношение

$$I_{вх}=U_{вх}/R_i,$$

где R_i - сопротивление резистора в цепи i-того входа.

Свойство ОУ суммировать входные токи с последующим преобразованием в напряжение широко используется при построении ЦАП и АЦП. На базе ОУ можно построить компараторы напряжения (сравнивающие устройства). При использовании ОУ в качестве компаратора напряжения на один его вход подается опорное напряжение $U_{оп}$, на второй - напряжение обрабатываемого (преобразуемого) сигнала U_x. При соответствующих условиях на выходе компаратора формируется сигнал логической“1”, если $(U_{оп} - U_x) > \Delta u_{кв}$, и логического “0”, если $(U_{оп} - U_x) < \Delta u_{кв}$ (рис. 5.5). Шаг квантования $\Delta u_{кв}$ обычно выбирается в пределах 5 . . . 10 мВ. Значение опорного напряжения и время установки компартора зависят от конкретного типа используемой интегральной микросхемы и условий его эксплуатации.

При реализации ЦАП в интегральном исполнении большие трудности вызывает подгонка высокоточных резисторов с сопротивлениями, отличающимися по номиналам друг от друга на несколько порядков. Поэтому, в интегральном исполнении применяются исключительно резистивная матрица R-2R. В качестве примера рассмотрим четырехразрядный ЦАП, использующий схему суммирования токов на ОУ (рис. 5.6).

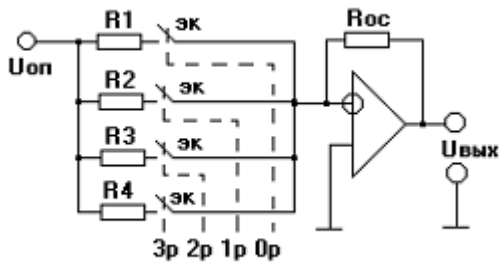


Рис. 5.6. Схема простейшего ЦАП. ЭК – электронные ключи; 0р, 1р, 2р, 3р, - соответствующие разряды цифровых входов управления

Относительная разрешающая способность рассматриваемого ЦАП:

$$\delta_o = \frac{1}{2^n - 1} = 0{,}0625.$$

Сопротивление резистора в цепи ключа, управляемого старшим разрядом двоичного кода, должно быть в два раза больше сопротивления резистора обратной связи R_{oc}. Сопротивление каждого последующего младшего разряда в два раза больше, чем сопротивление соседнего старшего разряда. Отсюда следует, что с увеличением количества разрядов цифровых входов ЦАП резко увеличивается соотношение сопротивлений резисторов нулевого и самого старшего разрядов ($R_0=2^nR_n$):

$$R_0/R_n=2^n = T.$$

Если n=8, то это отношение составляет 256. Увеличение T может привести к чрезмерному увеличению сопротивления резистора младшего разряда или же к сильному уменьшению номинала резистора самого старшего разряда. Поэтому ЦАП с резистивной матрицей R-2^nR применяется при небольшом количестве разрядов (при n<8). При больших T затруднительным становится также изготовление резистивных матриц в интегральном исполнении. Известно, что номиналы резисторов в интегральном исполнении не должны превышать 50...100 кОм. Поэтому, в ЦАП, выполненных по интегральной технологии, в основном применяются резистивные матрицы R-2R. Функциональная схема ЦАП с матрицей R-2R показана на рис. 5.7.

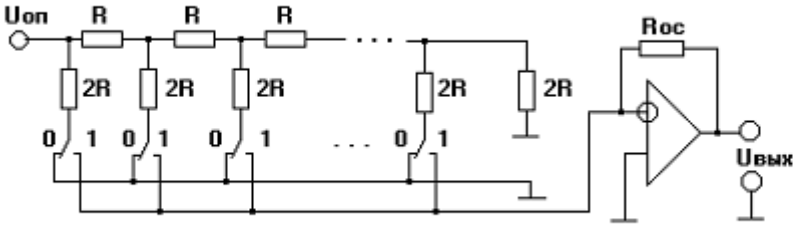


Рис. 5.7. Схема ЦАП с резистивной матрицей R - 2R

Напряжение на выходе ЦАП (рис. 5.7) определяется как:

$$U_{\text{вых}Ц\,АП} = (U_{оп}\frac{R_{ос}}{2R} + \frac{U_{оп}}{2}\frac{R_{ос}}{2R} + \frac{U_{оп}}{4}\frac{R_{ос}}{2R} + \ldots\ldots + \frac{U_{оп}}{2^n}\frac{R_{ос}}{2R}) =$$

$$= \frac{R_{оп}}{R}\left(\frac{U_{оп}}{2} + \frac{U_{оп}}{2\cdot 2^1} + \ldots + \frac{U_{оп}}{2\cdot 2^n}\right).$$

Чтобы выполнить условие формирования выходного напряжения в соответствии с двоичным кодом входного числа, необходимо получить равенство $R_{ос}=R$, тогда

$$U_{\text{вых}ЦАП} = U_{оп}\left(\frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^n}\right).$$

Сравнительные характеристики АЦП. Наибольшим быстродействием обладают АЦП прямого преобразования. Время преобразования $t_{пр}$ достигает 10 -20 нсек. Они используются для преобразования сигналов сверхбыстро протекающих процессов и сигналов телевизионного изображения (цифровое телевидение). Они отличаются высокой стоимостью и большой потребляемой мощностью. Функциональная схема АЦП прямого преобразования приведена на рис. 5.10. Она содержит 2^n компараторов, делитель опорного напряжения и преобразователь позиционного кода в параллельный двоичный код. Промышленностью выпускаются 4, 6, 8 - разрядные АЦП прямого преобразования. Время преобразования этих АЦП определяется исключительно только временем

распространения сигнала в компараторах $t_{здкр}$ и преобразователе кодов $t_{здпр}$, т.е. $t_{пр} = t_{здкр} + t_{здпр}$.

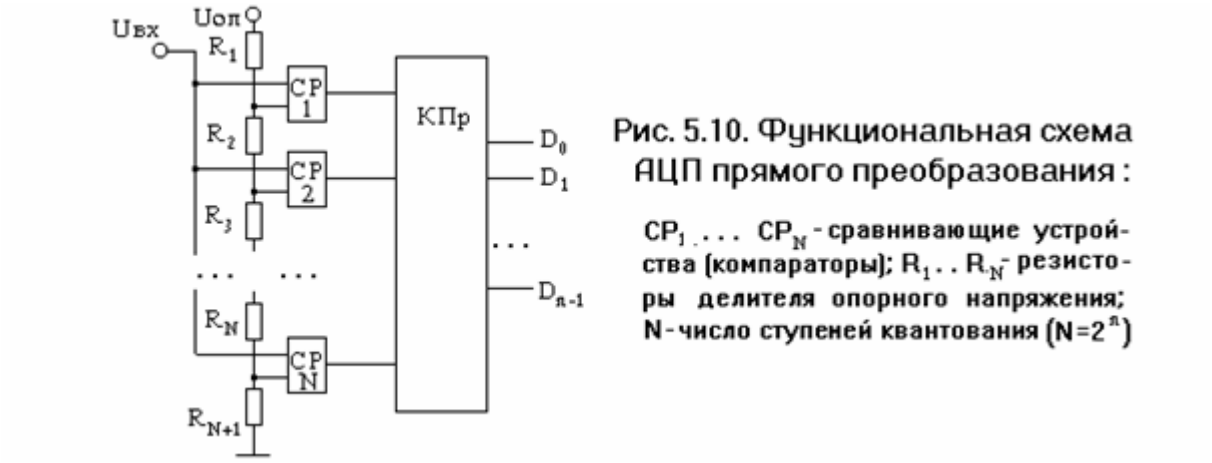


Рис. 5.10. Функциональная схема АЦП прямого преобразования :

$CP_1 \ldots CP_N$ - сравнивающие устройства (компараторы); $R_1 \ldots R_N$ - резисторы делителя опорного напряжения; N - число ступеней квантования ($N=2^n$)

По своему быстродействию на втором месте находятся АЦП последовательного приближения (рис. 5.11). Время преобразования n- разрядного АЦП определяется как $t_{пр} = nT + 3T$, где T - период следования тактовых импульсов, соответствующий времени выборки одного кванта. Дополнительные 3 такта используются для старта(запуска) и формирования сигналов признака завершения процесса преобразования (сигнала “конец преобразования”).

Наибольшим временем преобразования (среди АЦП с использованием ЦАП) обладает **АЦП последовательного счета** $t_{пр} = 2^n T$. Они проще в изготовлении и имеют наименьшую стоимость. Погрешность преобразования таких АЦП определяется, в основном, погрешностью ЦАП и может быть доведена до значений прецизионных преобразователей. АЦП последовательного счета переводит аналоговый сигнал в цифровой последовательно, начиная с младшего значащего разряда до цифрового кода на выходе, соответствующего уровню входного аналогового напряжения АЦП. Структурная схема такого АЦП приведена на рис. 5.13, а.

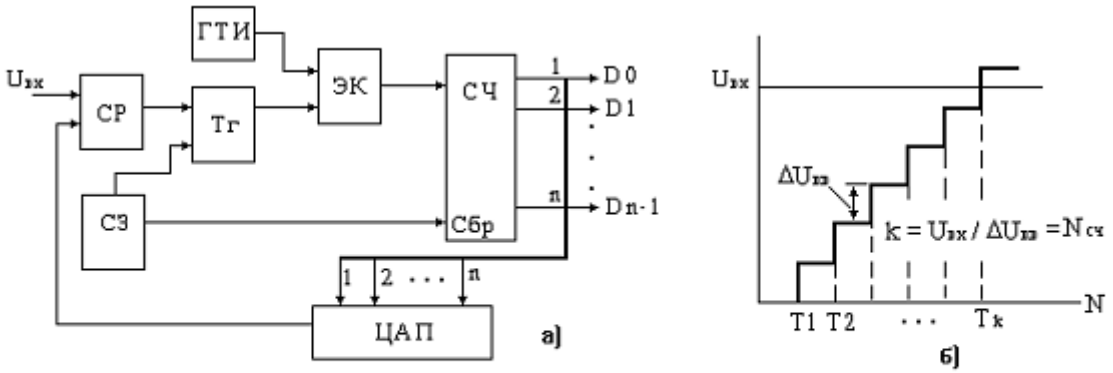


Рис. 5.13. АЦП последовательного счета (а) и его временная диаграмма (б)

С генератора тактовых импульсов через электронный ключ ЭК, который открывается в момент выборки входного аналогового сигнала схемой запуска (СЗ), последовательность импульсов поступает на n- разрядный двоичный счетчик (СЧ). Выход счетчика является выходом АЦП и одновременно управляет схемой ЦАП, вырабатывающей ступенчато нарастающее напряжение (см. рис. 5.13, б). В момент, когда выходное напряжение ЦАП станет равным входному, компаратор (СР) вырабатывает сигнал, опрокидывающий триггер (ТГ). При этом, сигнал с выхода триггера закроет электронный ключ и остановит счетчик. Содержание счетчика $N_{сч}$ после его остановки будет соответствовать числу, определяемому входным аналоговым сигналом

$$N_{сч} = U_{вх} / \Delta U_{кв}.$$

Наибольшее число в счетчике соответствует входному напряжению, равному $U_{пш}$. При этом $N_{сч} = 2^n$.

Тема № 4 Структура МПС (4 часа)

Общая структура МПС.

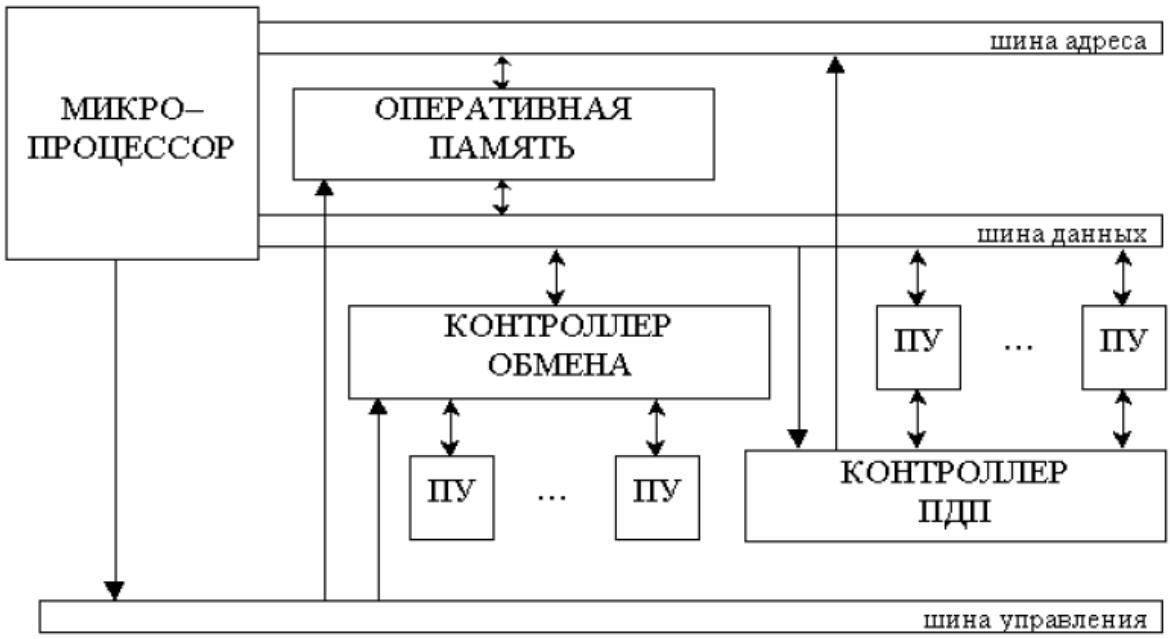


Рис.4.1. Структурная схема микропроцессорной системы.

Здесь же рассказать о **механизме прерываний.**

Понятие интерфейса в МПС. Примеры. Аппаратный интерфейс. Программный интерфейс.

Соединение всего многообразия внешних устройств с шинами микроконтроллера осуществляется с помощью интерфейсов, которые следует понимать как унифицированное средство объединения различных устройств в единую систему. Любой интерфейс должен обеспечить решение следующих двух задач.

Во-первых, интерфейс в своей аппаратной части должен обеспечить электрическое соединение различных внешних устройств с различными электрическими и конструктивными параметрами, с единой системой шин конкретного микроконтроллера. При этом должны быть учтены такие параметры, как количество линий связи, уровни и мощности электрических сигналов, длина и помехозащищенность линий связи.

Во-вторых, интерфейс должен обеспечить гибкое программное управление всеми подключенными внешними устройствами. В этой части интерфейс должен обеспечить не только работоспособность ВУ, но и согласование по быстродействию различных ВУ и центрального МП. Таким образом, под интерфейсом

следует понимать унифицированное программно-аппаратное устройство, предназначенное для организации обмена информацией между микропроцессором и внешними устройствами, объединенными в единую систему.

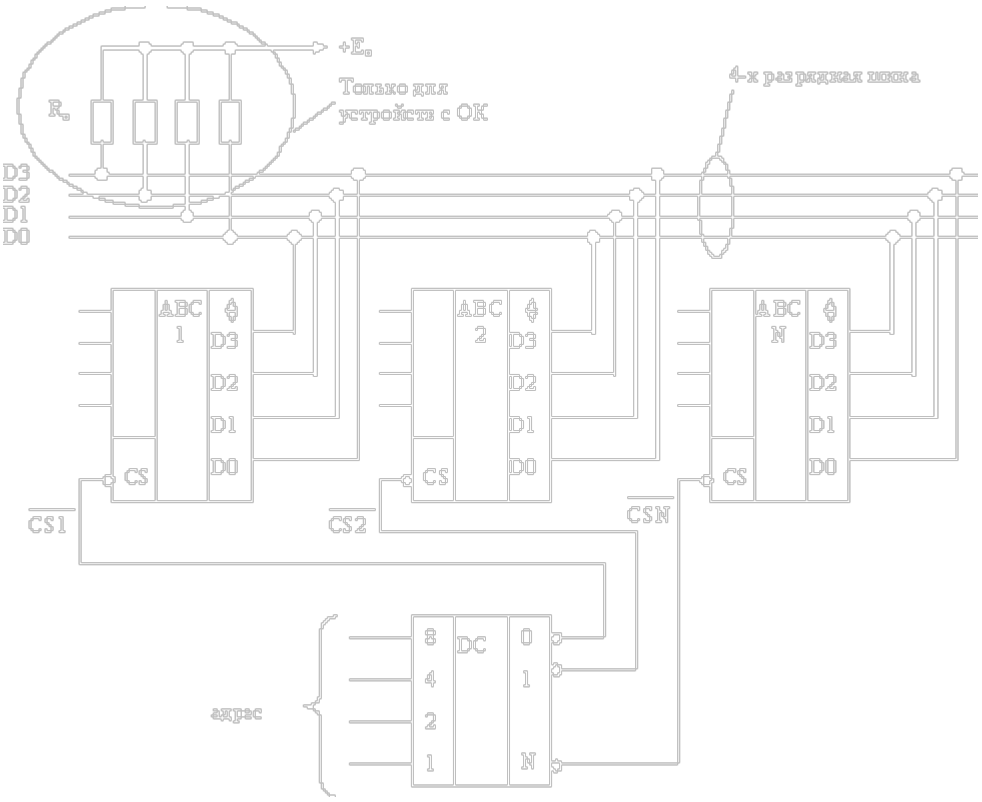


Рис.1.3. Принципиальная схема мультиплексирования многоразрядной шины

Шина — это группа проводников, соединяющих различные устройства. Шины можно разделить на группы в соответствии с выполняемыми функциями. Они могут быть внутренними по отношению к процессору и служить для передачи данных в АЛУ и из АЛУ, а могут быть внешними по отношению к процессору и связывать процессор с памятью или устройствами ввода-вывода.

Хотя разработчики процессоров могут использовать любой тип шины для микросхемы, должны быть введены четкие правила о том, как работает шина, и все устройства, связанные с шиной, должны подчиняться этим правилам, чтобы платы, которые выпускаются третьими лицами, подходили к системной шине. Эти правила называются протоколом **шины.**

Дешифратор адреса в МПС. Основные функции. Пример (схема).

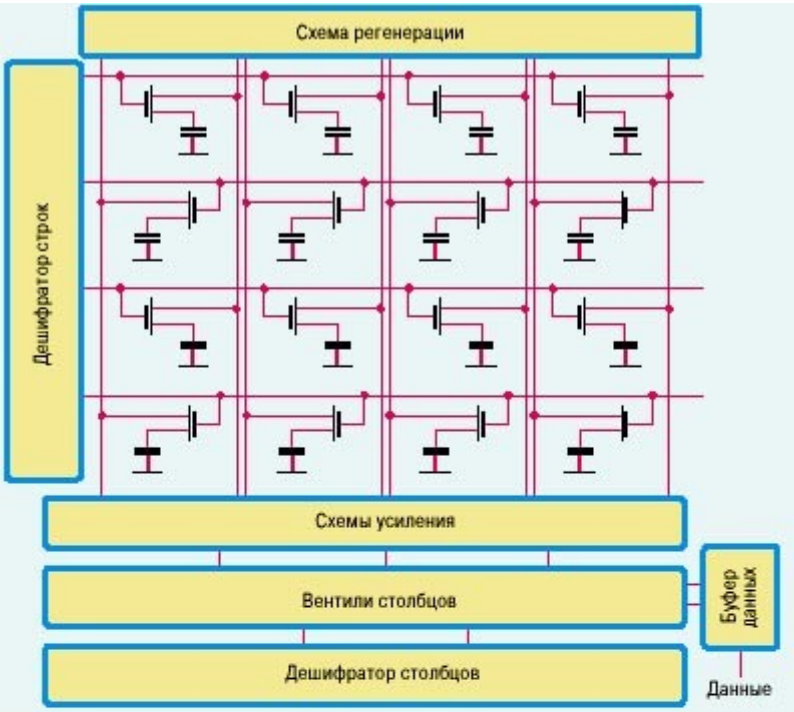


Рис. 2. Ячеистая структура динамической памяти

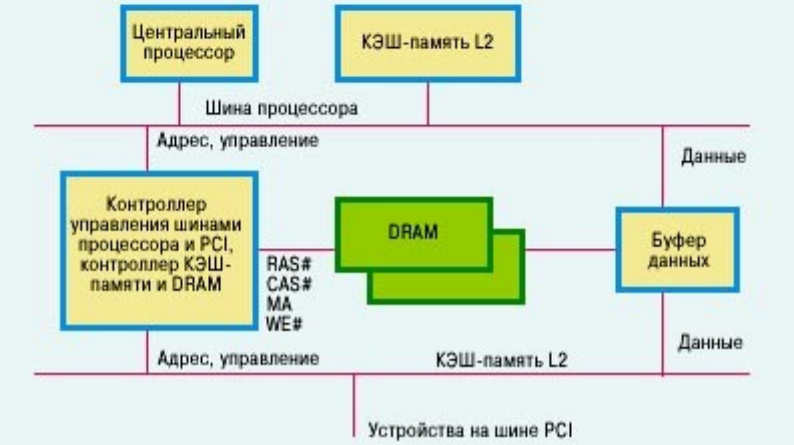
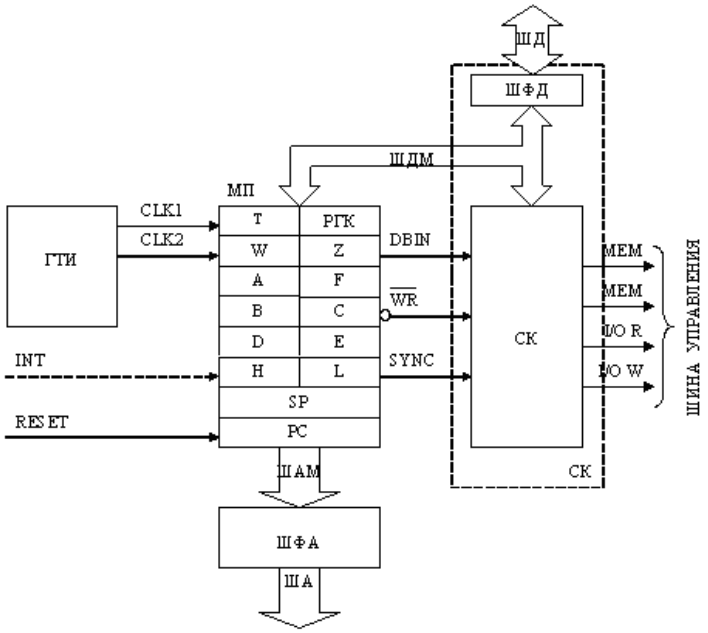


Рис. 3. Один из классических вариантов взаимодействия с подсистемой памяти



МП — микропроцессор (представлен множеством своих регистров);
ГТИ — генератор тактовых импульсов;
СК — системный контроллер;
ШФД — шинный формирователь шины данных;
ШФА — шинный формирователь шины адреса;
ШД – восьмиразрядная шина данных микро — ЭВМ;
ШДМ – восьмиразрядная шина данных микропроцессора;
ША – шестнадцатиразрядная шина адреса микро – ЭВМ;
ШАМ – шестнадцатиразрядная шина адреса микропроцессора.

Тактовый генератор обеспечивает синхронизацию работы микропроцессора, шинные формирователи – усиление по мощности сигналов шин, а системный контроллер считывает слово состояния процессора SW и формирует сигналы шины управления MEMR, MEMW, IOR и IOW из выходных сигналов микропроцессора DBIN и с учетом этого слова.

Тема №5 Передача данных в промышленных сетях (4 часа)

1. Способы передачи информации в МПС. Асинхронная передача данных.
2. Способы передачи информации в МПС. Синхронная передача данных.
3. Интерфейс RS485. Основные характеристики. Преимущества и недостатки.
4. Интерфейс RS232. Основные характеристики. Преимущества и недостатки.

2) Синхронизация шины

Шины можно разделить на две категории в зависимости от их синхронизации. **Синхронная шина содержит линию,** которая запускается кварцевым генератором. Сигнал на этой линии представляет собой меандр с частотой обычно от 5 до 100 МГц. Любое действие шины занимает целое число так называемых **циклов шины. Асинхронная шина** не содержит задающего генератора. Циклы шины могут быть любой требуемой длины и необязательно одинаковы по отношению ко всем парам устройств. Ниже мы рассмотрим каждый тип шины отдельно.

Синхронные шины

В качестве примера того, как работает асинхронная шина, рассмотрим временную диаграмму на рис. 3.34. В этом примере мы будем использовать задающий генератор на 40 МГц, который дает цикл шины в 25 не. Хотя может показаться, что шина работает медленно по сравнению с процессорами на 500 МГц и выше, не многие современные шины работают быстрее. Например, шина ISA (она встроена во все персональные компьютеры с процессором Intel) работает с частотой 8,33 МГц, и даже популярная шина PCI — с частотой 33 МГц или 66 МГц. Причины такой низкой скорости современных шин были даны выше: такие технические проблемы, как перекос шины и требование совместимости.

В нашем примере мы предполагаем, что считывание информации из памяти занимает 40 не с того момента, как адрес стал постоянным. Как мы скоро увидим, понадобится три цикла шины, чтобы считать одно слово. Первый цикл

начинается на нарастающем фронте отрезка $T_{ь\ a}$ третий заканчивается на нарастающем фронте отрезка T_3 j^g показано на рис. 3.34. Отметим, что ни один из нарастающих и задних фронтов не нарисован вертикально, потому что ни один электрический сигнал не может изменять свое значение за нулевое время. В нашем примере мы предполагаем, что для изменения сигнала требуется 1 не. Генератор и линии ADDRESS, DATA, MREQ, ~R~D, WAIT показаны в том же масштабе времени.

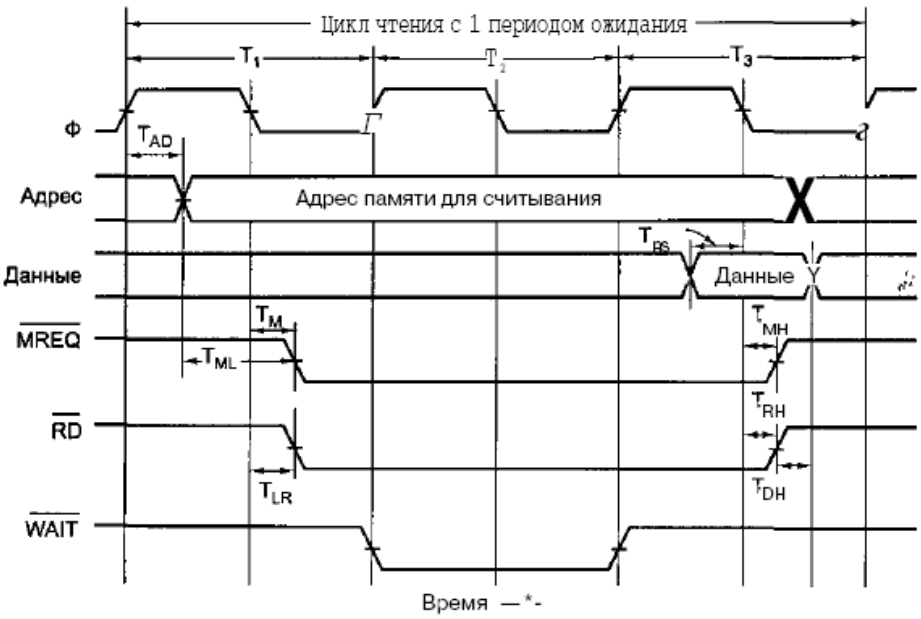


Рис. 3.34. Временная диаграмма процесса считывания на синхронной шине

Начало Ti определяется нарастающим фронтом генератора. За часть времени Ti центральный процессор помещает адрес нужного слово на адресные линии. Поскольку адрес представляет собой не одно значение (в отличие от генератора), мы не можем показать его в виде одной линии на схеме. Вместо этого мы показали его в виде двух линий с пересечениями там, где этот адрес меняется. Серый цвет на схеме показывает, что в этот момент не важно, какое значение принял сигнал. Используя то же соглашение, мы видим, что содержание линий данных не имеет значения до отрезка T_3

После того как у адресных линий появляется возможность приобрести новое значение, устанавливаются сигналы MREQ и RD. Первый указывает, что осуществляется доступ к памяти, а не к устройству ввода-вывода, а второй — что осуществляется чтение, а не запись. Поскольку считывание информации из памяти занимает 40 не после того, как адрес стал постоянным (часть первого цикла), память не может передать требуемые данные за период T_2. Чтобы центральный процессор не ожидал поступления данных, память устанавливает линию WAIT в начале отрезка T_a Это действие вводит периоды ожидания (дополнительные циклы шины), до тех пор пока память не сбросит сигнал WAIT. В нашем примере вводится один период ожидания (T_2) ? поскольку память работает слишком медленно. В начале Тз, когда есть уверенность в том, что память получит данные в течение текущего цикла, сигнал WAIT сбрасывается.

Во время первой половины T_3 память помещает данные на информационные линии. На заднем фронте T_3 центральный процессор стробирует (то есть считывает) информационные линии, сохраняя их значения во внутреннем регистре. Считав данные, центральный процессор сбрасывает сигналы MREQ. и RD. В случае необходимости на следующем нарастающем фронте может начаться еще один цикл памяти.

Далее проясняется значение восьми символов на временной диаграмме (см. рис. 3.34 и табл. 3.4). $T_{AD;}$ например, — это временной интервал между нарастающим фронтом Т(и установкой адресных линий. В соответствии с требованиями синхронизации $T_{AD} < 11$ $_{Н}$е. Значит, производитель процессора 1арантирует, что во время любого цикла считывания центральный процессор будет выдавать требуемый адрес в пределах 11 не от середины нарастающего фронта TV

Таблица 3.4. Некоторые временные характеристики процесса считывания на синхронной шине

Символ	Значение	Минимум, не	Максимум, не
TAD	Задержка выдачи адреса		11
$T_{мL}$	Промежуток между стабилизацией адреса и установкой сигнала $\overline{MREQ}$	6	
$T_{м}$	Промежуток между задним фронтом синхронизирующего сигнала в цикле Ti и установкой сигнала $\overline{MREQ}$		8
TR_{L}	Промежуток между задним фронтом синхронизирующего сигнала в цикле T_{t} и установкой сигнала $\overline{RD}$		8
Tos	Период передачи данных до заднего фронта синхронизирующего сигнала	5	
$T_{мн}$	Промежуток между задним фронтом синхронизирующего сигнала в цикле Тз и сбросом сигнала $\overline{MREQ}$		8
Тдн	Промежуток между задним фронтом синхронизирующего сигнала в цикле Тэ и сбросом сигнала $\overline{RD}$		8
Тон	Период продолжения передачи данных с момента сброса сигнала $\overline{RD}$	0	

Условия синхронизации также требуют, чтобы данные поступали на информационные линии по крайней мере за 5 не (T_{DS}) $_{до}$ заднего фронта $T_{3;}$ ^бы дать данным время установиться до того, как процессор стробирует их.' Сочетание ограничении на $_{A}D$ И T_D с означает, что в худшем случае в распоряжении памяти будет только 62,5-11-5=46,5 не с момента появления адреса и до момента, когда нужно выдавать данные. Поскольку достаточно 40 не, память даже в самом худшем случае может всегда ответить за период T_3 Если памяти для считывания требуется 50 не, то необходимо ввести второй период ожидания, и тогда память ответит в течение Т, Требования синхронизации гарантируют, что адрес будет установлен по крайней мере за 6 не до того, как появится сигнал MREQ. Это время может быть важно в том случае, если MKHQ запускает выбор элемента памяти, поскольку некоторые типы памяти требуют некоторого времени на установку адреса до выбора элемента памяти. Ясно, что разработчику системы не следует выбирать микросхему памяти, на установку которой нужно 10 не.

Ограничения на ^{T},, и $T_{R[}$. означают, что WREQ и RD будут установлены в пределах 8 не от заднего фронта $T_{ь}$ в худшем случае у микросхемы памяти по-

сле установки MREQ и "КБ останется всего 25+25-8-5=37 не на передачу данных по шине. Это ограничение дополнительно по отношению к интервалу в 40 не и не зависит от него.

Тмн и TRH определяют, сколько времени требуется на отмену сигналов MREQ и ТШ после того, как данные стробированы. Наконец, T_{oH} определяет, сколько времени память должна держать данные на шине после снятия сигнала КП. В нашем примере приданном процессоре память может удалить данные с шины, как только сбрасывается сигнал RT); при других процессорах, однако, данные могут сохраняться еще некоторое время.

Необходимо подчеркнуть, что наш пример представляет собой сильно упрощенную версию реальных временных ограничений. В действительности должно определяться гораздо больше таких ограничений. Тем не менее этот пример наглядно демонстрирует, как работает синхронная шина.

Отметим, что сигналы управления могут задаваться или с помощью низкого, или с помощью высокого напряжения. Что является более удобным в каждом конкретном случае, должен решать разработчик, хотя, по существу, выбор произволен.

3) Асинхронные шины

Хотя достаточно удобно использовать синхронные шины благодаря дискретным временным интервалам, здесь все же есть некоторые проблемы. Например, если процессор и память способны закончить передачу за 3,1 цикла, они вынуждены продлить ее до 4,0 циклов, поскольку неполные циклы запрещены.

Еще хуже то, что если однажды был выбран определенный цикл шины и в соответствии с ним были разработаны память и карты ввода-вывода, то в будущем трудно делать технологические усовершенствования. Например, предположим, что через несколько лет после выпуска системы, изображенной на рис. 3.34, появилась новая память с временем доступа не 40, а 20 не. Это избавило бы нас от периода ожидания и увеличило скорость работы машины. Теперь представим, что появилась память с временем доступа 10 не. При этом улучшения производительности уже не будет, поскольку в данной разработке минимальное время для чтения — 2 цикла.

Если синхронная шина соединяет ряд устройств, одни из которых работают быстро, а другие медленно, шина подстраивается под самое медленное устройство, а более быстрые не могут использовать свой полный потенциал.

По этой причине были разработаны асинхронные шины, то есть шины без задающего генератора, как показано на рис. 3.35. Здесь ничего не привязывается к генератору. Когда задающее устройство устанавливает адрес, MREQ, RD и любой другой требуемый сигнал, он выдает специальный сигнал, который мы будем называть MSYN (Master SYNchronization). Когда подчиненное устройство получает этот сигнал, оно начинает выполнять свою работу настолько быстро, насколькс это возможно. Когда работа закончена, устройство выдает сигнал SSYN (Slavf SYNchronization).

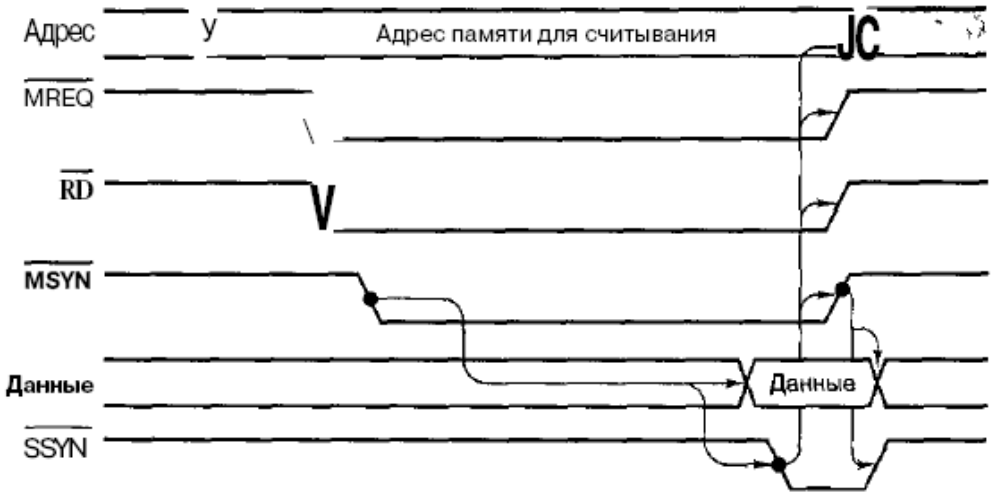


Рис. 3.35. Работа асинхронной шины

Сигнал SSYN означает для задающего устройства, что данные доступны. *Oi* фиксирует их, а затем отключает адресные линии вместе с MREQ, RD и M$YN Отмена сигнала MSYN означает для подчиненного устройства, что цикл закончен поэтому устройство отменяет сигнал SSYN, и все возвращается к первоначальному состоянию, когда все сигналы отменены.

Стрелочки на временных диаграммах асинхронных шин (а иногда и синхронных шин) показывают причину и следствие какого-либо действия (рис. 3.35) Установка сигнала MSYN приводит к запуску информационных линий, а также *ь* установке сигнала SSYN. Установка сигнала S"SYN, в свою очередь, вызывает отключение адресных линий, MREQ, RD и MSYN. Наконец, отключение MSYN вызывает отключение SSYN^ и на этом процесс считывания заканчивается.

Набор таких взаимообусловленных сигналов называется **полным квитированием.** Здесь, в сущности, наблюдается 4 события:

1. Установка сигнала MSYN.
2. Установка сигнала SSVN в ответ на сигнал MSYN.
3. Отмена сигнала MSYN в ответ на сигнал SSYN.
4. Отмена сигнала SSYN в ответ на отмену сигнала MSYN.

Следует уяснить, что взаимообусловленность сигналов не зависит от синхронизации. Каждое событие вызывается предыдущим событием, а не импульсами генератора. Если какая-то пара двух устройств (задающего и подчиненного) работает медленно, это никак не повлияет на следующую пару устройств, которая работает гораздо быстрее.

Преимущества асинхронной шины очевидны, но в действительности большинство шин являются синхронными. Дело в том, что синхронную систему построить проще, чем асинхронную. Центральный процессор просто выдает сигналы, а память просто реагирует на них. Здесь нет никакой причинно-следственной связи, но если компоненты выбраны удачно, все будет работать и без квитирования. Кроме того, в разработку синхронных шин сделано очень много вложений.

1.2. Последовательный интерфейс RS232

Последовательный интерфейс RS232 - это промышленный стандарт для последовательной двунаправленной асинхронной передачи данных. Он используется в компьютерах при подсоединении принтеров, модемов, мыши и т.д. Максимальное расстояние, позволяющее организовать связь, равно 20 м.

В отличие от параллельного порта, состоящего из восьми информационных линий и за один такт передающего байт, порт RS232 требует наличия только одной такой линии, по которой последовательно передается бит за битом. Это позволяет сократить количество информационных линий для передачи данных между устройствами, но уменьшает скорость.

1.2.1. Последовательная передача данных

Последовательный поток данных состоит из битов синхронизации и собственно битов данных. Формат последовательных данных содержит четыре части: стартовый бит, биты данных (5-8 бит), проверочный и столовый биты; вся эта конструкция иногда называется *символом.* На рис. 1.6 изображен типичный формат последовательных данных.

Когда данные не передаются, на линии устанавливается уровень логической единицы. Это называется режимом ожидания. Начало режима передачи данных характеризуется передачей уровня логического нуля длительностью в одну элементарную посылку. Такой бит называется *стартовым.* Биты данных посылаются последовательно, причем младший бит - первым; всего их может быть от пяти до восьми. За битами данных следует проверочный бит, предназначенный для обнаружения ошибок, которые возникают во время обмена данными. Последней передается стоповая посылка, информирующая об окончании символа. Столовый бит передается уровнем логической единицы. Длительность стоповой посылки -1, 1,5 или 2 бита. Специально разработанное электронное устройство, генерирующее и принимающее последовательные данные, *называется универсальным асинхронным приемопередатчиком* (Universal Asynchronous Receiver Transmitter, UART).

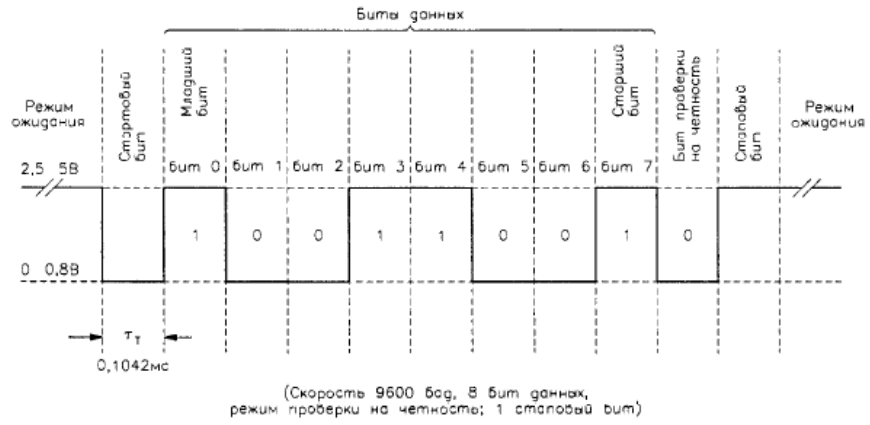


Рис. 1.6. Формат последовательных данных, формируемых UART

Обмен информацией с помощью микросхем UART происходит следующим образом. Приемник обнаруживает первый фронт стартового бита и выжи-

дает один или полтора тактовых интервала, поскольку считывание должно начаться точно в середине первой посылки. Через один тактовый интервал считывается второй бит данных, причем это происходит точно в середине второй посылки. После окончания информационного обмена приемник считывает проверочный бит для обнаружения ошибок и столовый бит, а затем переходит в режим ожидания следующей порции данных.

Скорость передачи информации в последовательном интерфейсе измеряется в *бодах* (бод - количество передаваемых битов за 1 с). Стандартные скорости равны 110, 150, 300, 600, 1200, 2400, 4800, 9600 и 19200 бод. Зная скорость в бодах, можно вычислить число передаваемых символов в секунду. Например, если имеется восемь бит данных без проверки на четность и один столовый бит, то общая длина последовательности, включая стартовый бит, равна 10. Скорость передачи символов соответствует скорости в бодах, деленной на 10. Таким образом, при скорости 9600 бод (см. рис. 1.6) будет передаваться 960 символов в секунду.

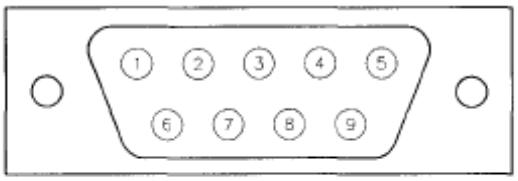
Проверочный бит предназначен для обнаружения ошибок в передаваемых битах данных. Когда он присутствует, осуществляется проверка на четность или нечетность. Если интерфейс настроен на проверку по четности, такой бит будет выставляться в единицу при нечетном количестве единиц в битах данных, и наоборот. Это простейший способ проверки на наличие одиночных ошибок в передаваемом блоке данных. Однако, если во время передачи искажению подверглись несколько битов, подобная ошибка не обнаруживается. Проверочный бит генерируется передающим UART таким образом, чтобы общее количество единиц было нечетным или четным числом в зависимости от настройки интерфейса; приемное устройство должно иметь такую же настройку. Приемный UART считает количество единиц в принятых данных. Если данные не проходят проверку, генерируется сигнал ошибки.

Большинство компьютеров, совместимых с IBM PC, использует UART 16450, с IBM PC XT - UART 8250. В UART применяются уровни напряжения ТТЛ. Для передачи данных по каналу связи напряжение с помощью специализированных преобразователей конвертируется с инверсией: логическому нулю соответствует диапазон напряжений от +3 до +12 В, логической единице - от -3 до -12 В.

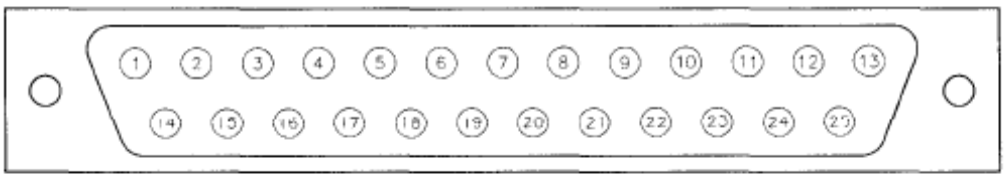
1.2.2. Разъем и кабель порта RS232

Стандартный последовательный порт имеет 25- или 9-контактный разъем. На рис. 1.7 приведены назначения контактов этих разъемов.

а)



б)



в)

25 контактов	9 контактов	Наименование	Направление (относительно ПК)	Описание
1		PROT		Защитное заземление
2	3	TD	Выход	Передаваемые данные
3	2	RD	Вход	Принимаемые данные
4	7	RTS	Выход	Запрос на передачу
5	8	CTS	Вход	Очищен для передачи
6	6	DSR	Вход	Готовность внешнего устройства
7	5	GND		Сигнальное заземление
8	1	DCD	Вход	Обнаружение информационного сигнала
20	4	DTR	Выход	Компьютер к обмену данными готов
22	9	RI	Вход	Индикатор звонка
23		DSRD	Вход/Выход	Детектор скорости передачи данных

Рис. 1.7. Функции контактов разъемов RS232 на компьютере а – блочная часть 9 контактного штыревого разъема, вид со стороны задней стенки компьютера, б – блочная часть 25-контактного штыревого разъема, вид со стороны задней стенки компьютера, в – назначение контактов разъемов последовательного порта

В табл. 1.2 указано назначение сигналов последовательного интерфейса.

На рис. 1.8 представлены два типа соединений между компьютером и внешним устройством по протоколу RS232. Стрелки показывают направление потоков данных. На рис. 1.8а представлено так называемое *нуль-модемное соединение.* На рис. 1.86 изображено соединение, использующее только три линии: первая - для передачи данных, вторая - для приема, третья - общая. Соединение организовано таким образом, что передаваемые данные от первого устройства поступают на приемную линию второго.

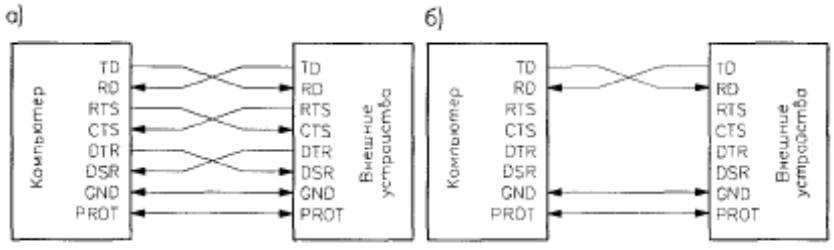


Рис. 1.8. Соединение компьютера и внешнего устройства по протоколу RS232 а – с использованием нуль-модемного кабеля, б – при помощи трех линий

RS 485

Стандарт RS-485 совместно разработан двумя ассоциациями: Ассоциацией электронной промышленности (**EIA - Electronics Industries Association**) и Ассоциацией промышленности средств связи (**TIA - Telecommunications Industry Associastion**). Ранее EIA маркировала все свои стандарты префиксом "RS" (Рекомендованный стандарт). Многие инженеры продолжают использовать это обозначение, однако EIA/TIA официально заменил "RS" на "EIA/TIA" с целью облегчить идентификацию происхождения своих стандартов. На сегодняшний день, различные расширения стандарта RS-485 охватывают широкое разнообразие приложений.

В стандарте **RS-485** для передачи и приема данных часто используется единственная витая пара проводов. Процедуры совместного использования линии передачи требуют применения определенного метода управления направлением потока данных. Наиболее широко распространенным методом является использование сигналов **RTS** (Request To Sent) и **CTS** (Clear To Sent).

Технические характеристики интерфейса **RS-485**

Интерфейс **RS-485** поддерживает стандарты: **EIA/TIA-422** 90kbit/1,2km, **EIA/TIA-422A, EIA/TIA -485**

BitBus
(Intel, Protocol HLDC)

Поддержка сетей:
BitBus, **ModBus**, **InterBus-S**, **DIN Measuring Bus**

Примечание:
1. Стандарт RS-485 оговаривает только электрические характеристики, физический уровень (среду), но не программную платформу.
2. Стандарт RS-485 не оговаривает:

• возможность объединения несимметричных и симметричных цепей,
• параметры качества сигнала, уровень искажений (%),
• методы доступа к линии связи,
• протокол обмена,
• аппаратную конфигурацию (среда обмена, кабель),
• типы соединителей, разъемов, колодок, нумерацию контактов,
• качество источника питания (стабилизация, пульсация, допуск),
• отраженность, уровень сигнала (reflect).

Электрические и временные характеристики интерфейса RS-485

32 приемопередатчика при многоточечной конфигурации сети (на одном сегменте, максимальная длина линии в пределах одного сегмента сети: 1200 метров (4000 футов)).
Только один передатчик активный.
Максимальное количество узлов в сети - 250 с учетом магистральных усилителей.
Характеристика скорость обмена/длина линии связи (зависимость экспоненциальная):
62,5 кбит/с 1200 м (одна витая пара)
375 кбит/с 300 м (одна витая пара)
500 кбит/с
1000 кбит/с
2400 кбит/с 100 м (две витых пары)
10000 кбит/с 10 м

Примечание. Скорости обмена 62,5 кбит/с, 375 кбит/с, 2400 кбит/с оговорены стандартом RS-485. На скоростях обмена свыше 500 кбит/с рекомендуется использовать экранированные витые пары.

Тип приемопередатчиков - дифференциальный, потенциальный. Изменение входных и выходных напряжений на линиях A и B: Ua (Ub) от -7В до +12В (+7В).
Требования, предьявляемые к выходному каскаду: - выходной каскад представляет собой источник напряжения с малым выходным сопротивлением, |Uвых| =1,5:5,0В (не <1,5В и не >6,0В);
- состояние логической "1": Ua больше Ub (гистерезис 200мВ) - MARK, OFF;
- состояние логического "0": Ua меньше Ub (гистерезис 200мВ) - SPACE, ON;
- выходной каскад должен выдерживать режим короткого замыкания, иметь максимальный выходной ток 250мА, скорость нарастания выходного сигнала 1,2В/мкс и схему ограничения выходной мощности.

Требования, предъявляемые к входному каскаду: - входной каскад представляет собой дифференциальный вход с высоким входным сопротивлением и пороговой характеристикой от -200мВ до +200мВ;
- допустимый диапазон входных напряжений Uag (Ubg) относительно земли (GND) от -7В до +12В;
- входной сигнал представлен дифференциальным напряжением (Ui+0,2В) и более;
- уровни состояния приемника входного каскада - см. состояния передатчика выходного каскада.

V. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ К САМОСТОЯТЕЛЬНОЙ РАБОТЕ

В качестве самостоятельной работы рекомендуется знакомство с публикациями в периодических изданиях, поиск информации в Internet (например, по ссылке www:/amursu.ru/citforum) по темам, предлагаемым в качестве самостоятельного изучения и подготовки лабораторных работ.

Вопросы для самостоятельного изучения:

1. Применение МПС в управлении синхронными двигателями.
2. Применение МПС в управлении асинхронными двигателями.
3. Цифровые датчики.
4. Цифровые регуляторы.
5. Частотное управление с помощью цифровых частотных преобразователей.

К видам самостоятельной работы также относится подготовка к лабораторным работам, составление отчетов, разбор материалов лекций во время семестра.

Контроль за выполнением самостоятельной работы осуществляется на экзамене.

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОРГАНИЗАЦИИ МЕЖСЕССИОННОГО КОНТРОЛЯ ЗНАНИЙ

Межсессионный контроль осуществляется на основе выполнения домашних заданий, лабораторных работ, промежуточных тестов.

По итогам выполнения перечисленных видов работ в сроки, установленные деканатом (как правило, на 6-ой и 12-ой неделе семестра) преподавателем выставляется аттестационная оценка по пятибалльной системе.

VII. ТРЕБОВАНИЯ К ЗНАНИЯМ СТУДЕНТОВ, ПРЕДЪЯВЛЯЕМЫЕ НА ЗАЧЕТЕ

Для получения зачета студент должен посещать занятия, проявлять активность в аудитории, знать теоретический материал в требуемом объеме, защитить отчет по самостоятельной работе, а так же на зачетном занятии в полном объеме ответить на вопросы заданные преподавателем.

Вопросы к зачету приведены в рабочей программе дисциплины.

VIII. КАРТА ОБЕСПЕЧЕННОСТИ ДИСЦИПЛИНЫ КАДРАМИ ПРОФЕССОРСКО-ПРЕПОДАВАТЕЛЬСКОГО СОСТАВА

Все виды занятий по дисциплине ведет ассистент Дрюков А.А.

СОДЕРЖАНИЕ

Дрюков Александр Александрович,
ассистент кафедры ИиУС АмГУ

Учебно-методический комплекс по дисциплине «Микропроцессорные средства» для спец. 230201 – Информационные системы и технологии

Изд-во АмГУ. Подписано к печати Формат 60х84/16. Усл. печ. л.

Тираж Заказ