

• РАДИО И СВЯЗЬ •

# СПРАВОЧНИК

---

Б. В. ШЕВКОПЛЯС

## МИКРО- ПРОЦЕССОРНЫЕ СТРУКТУРЫ

•  
ИНЖЕНЕРНЫЕ  
РЕШЕНИЯ

# СПРАВОЧНИК

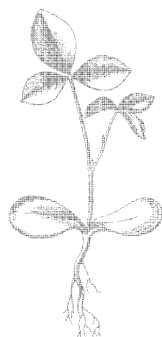
Б. В. ШЕВКОПЛАС

## МИКРО- ПРОЦЕССОРНЫЕ СТРУКТУРЫ • ИНЖЕНЕРНЫЕ РЕШЕНИЯ

ДОПОЛНЕНИЕ ПЕРВОЕ



МОСКВА „РАДИО И СВЯЗЬ”  
1993



Scan AAW

ББК 32.97  
Ш 37  
УДК 621.395.34 (075)

**Федеральная целевая программа книгоиздания России**

**Редакция литературы по электронике**

Ш  $\frac{2302037000-036}{046(01)-93}$  13-93

**ISBN 5-256-00973-7**

© Шевкопляс Б. В., 1993

Многие вещи нам непонятны не потому, что наши понятия слабы; но потому, что сии вещи не входят в круг наших понятий.

*К. Прутков*

## **Предисловие**

Предлагая вашему вниманию третью сотню избранных зарубежных изобретений (с предыдущими вы, вероятно, уже знакомы по второму изданию справочника, 1990 г.), мы рассчитываем пробудить творческий дух тех читателей, кто недооценивает уровень своих идей, вольно или невольно сопоставляя его с неким ускользающим Мировым Уровнем. Последний, однако, материализуется в конкретных и по сути простых технических решениях, лучшие из них патентуются. Иными словами, здесь вы найдете пробный камень для проверки своих идей. Они витают в воздухе, не хватает только центров кристаллизации, которыми, мы надеемся, могут послужить собранные здесь решения—аналоги и прототипы ваших будущих изобретений.

Но дело, конечно, не только в моральной поддержке изобретателей. Просмотрите оглавление—не правда ли, знакомые темы? Сколько раз приходилось нам решать головоломки адресации, бороться с гонками и т. д., и т. п.! И обратите внимание, как четко поставлены задачи, как красиво они решены—будем же и мы так работать!

## Глава 1

# Взаимодействие микроЭВМ с объектами

### 1.1. Опрос резисторного датчика температуры и определение положения движка переменного резистора с использованием порта ввода—вывода микроЭВМ [1]

В системе управления бытовой стиральной машиной использована микроЭВМ (рис. 1.1), которая помимо выполнения прочих функций следит за положением движка переменного резистора R1 (задающего температуру воды) и измеряет сопротивление терморезистора R2, установленного в баке с водой.

Каждый выход Q1—Q4 микроЭВМ может быть программно и независимо от других выходов установлен в состояние лог.0 и лог.1 (напряжения низкого и высокого уровней) или «отключено» (высокое выходное сопротивление). В исходном состоянии выходы Q1—Q4 отключены, конденсатор C разряжен, на входе А микроЭВМ за счет высокоомного резистора R4 поддерживается напряжение, близкое к напряжению  $+U$  источника питания.

Для определения положения движка переменного резистора R1 микроЭВМ проводит два эксперимента. Сначала она формирует на выходе Q4 сигнал лог.0 (остальные выходы отключены). Конденсатор C начинает заряжаться с постоянной времени R1C, R1—сопротивление между выходом Q4 и входом А, высокоомным резистором R4 пренебрегаем. Напряжение на входе А экспоненциально уменьшается до порога срабатывания приемного элемента входного порта, микроЭВМ регистрирует интервал времени T1 между выдачей сигнала лог.0 на выход Q4 и срабатыванием этого элемента.

После восстановления напряжения высокого уровня на входе А (микроЭВМ может форсировать процесс восстановления одновременной выдачей сигналов лог.1 на выходы Q1—Q4) и перевода выходов Q1—Q4 в отключенное состояние микроЭВМ проводит второй эксперимент, аналогичный первому, но сигнал лог.0 формируется на выходе Q3, а не Q4. В результате

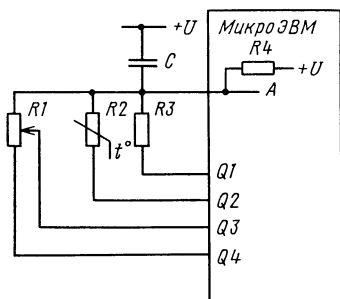


Рис. 1.1. Подключение к микроЭВМ переменного резистора R1 (датчика температуры воды в баке стиральной машины) и терморезистора R2 (датчика температуры воды)

конденсатор  $C$  заряжается с постоянной времени  $\alpha R_1 C$ , где  $0 \leq \alpha \leq 1$  — коэффициент, отражающий угол поворота движка переменного сопротивления. При срабатывании приемного элемента микроЭВМ фиксирует интервал времени  $T_2$ . Затем искомым коэффициент  $\alpha$  вычисляется по формуле  $\alpha = T_2/T_1$ .

Для измерения сопротивления терморезистора  $R_2$  микроЭВМ также проводит два эксперимента. В первом сигнал лог. 0 подается на выход  $Q_1$ , во втором — на выход  $Q_2$  (остальные три выхода отключены). Резистор  $R_3$  является эталонным, имеет высокую температурную стабильность и находится при комнатной температуре. Сопротивление резистора  $R_2$  вычисляется по формуле  $R_2 = R_3 T_4/T_3$ , где  $T_3$  и  $T_4$  — интервалы времени между выходным и входным сигналами, полученные при проведении первого и второго экспериментов.

## 1.2. Упрощение схемы связи микроЭВМ с клавиатурой и светодиодными индикаторами [2]

В схеме (рис. 1.2, а) клавиатура содержит шесть нормально разомкнутых ключей и два светодиодных индикатора (число ключей и индикаторов можно увеличить, но это усложнило бы рисунок). Опрос клавиатуры сопровождается перебором строк матрицы. Изменяющийся двоичный код  $A$  номера строки преобразуется дешифратором  $D1$  в сигнал типа «бегущий 0», который проходит через усилители  $D2$  с активными выходами и поочередно воздействует на строки матрицы.

Если все ключи разомкнуты, то благодаря резисторам, соединенным с шиной питания  $+U$ , на входах усилителей  $D3$  поддерживаются сигналы лог. 1, которые проходят через эти усилители (предпочтительно, имеющие гистерезисные передаточные характеристики вход—выход) и в виде кода

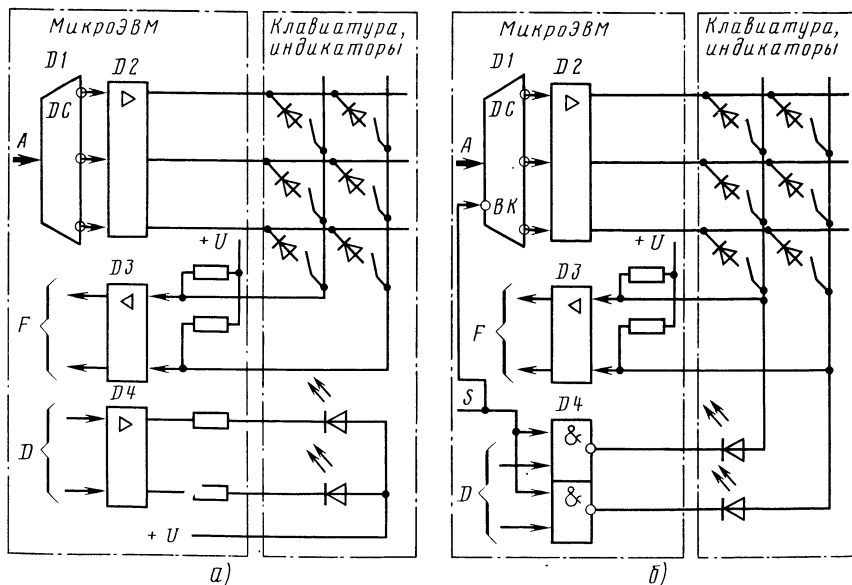


Рис. 1.2. Традиционная (а) и предлагаемая (б) схемы связи микроЭВМ с клавиатурой и светодиодными индикаторами.

Уменьшено число резисторов и соединительных проводов

F поступают в процессор или контроллер. Если в строке, на которую с усилителей D2 подан сигнал лог. 0, замкнут один или оба ключа, то этот сигнал передается через один или два диода на один или оба входа усилителей D3, в результате формируется соответствующий код F.

Для отображения состояния микроЭВМ (Останов, Ошибка и т. п.) используются два светодиодных индикатора. Данные D хранятся в некотором регистре (не показан) и через усилители D4 с активными выходами или открытым коллектором передаются на светодиоды. Светодиод включен, если на соответствующем выходе усилителя D4 сформирован сигнал лог. 0; последовательно включенный резистор задает рабочий ток.

Упрощение схемы связано с уменьшением числа резисторов и исключением соединительного провода, по которому передавалось напряжение питания +U (рис. 1.2, б). Вместо усилителей D4 применены элементы И—НЕ с активными выходами.

В отсутствие опроса клавиатуры  $S=1$ , дешифратор D1 выключен, на его выходах—сигналы лог. 1, на строках матрицы и входах усилителей также присутствуют напряжения высокого уровня, элементы И—НЕ D4 открыты по верхним входам и транслируют на выходы сигналы D, выполняя роль инверторов. Резисторы ограничивают прямые токи через светодиоды (если они включены). Замыкания ключей не меняют состояния схемы.

На время опроса клавиатурной матрицы сигнал S принимает нулевое значение, дешифратор D1 активизируется (на его инверсный вход выбора кристалла (ВК) подан разрешающий сигнал—лог. 0), элементы И—НЕ D4 закрываются, ранее включенные светодиоды выключаются и не препятствуют повышению напряжения на входах усилителей D3 при опросе разомкнутых ключей строки.

Процесс опроса аналогичен рассмотренному ранее; после его завершения сигнал S принимает единичное значение, дешифратор закрывается, элементы И—НЕ D4 возобновляют выдачу данных на индикаторы. Кратковременное выключение светодиодов, связанное с периодом опроса клавиатуры, незаметно для глаза, поэтому с точки зрения пользователя обе рассмотренные схемы одинаковы.

### 1.3. Идентификация нажатых клавиш с использованием преобразователя сопротивление—частота [3]

В микроЭВМ (рис. 1.3), входящей в состав торгового автомата, имеется ряд нормально разомкнутых ключей S. Покупатель вводит в автомат кредитную карточку или опускает монеты, затем нажатием клавиши замыкает

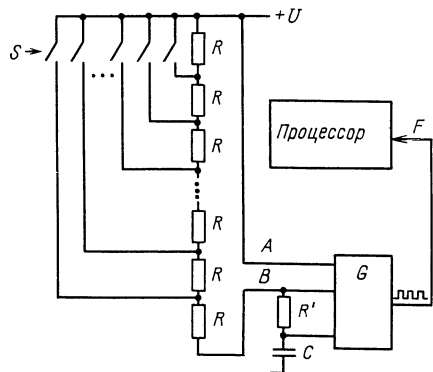


Рис. 1.3. Схема связи клавиатуры с микроЭВМ.

Нажатие ключа S вызывает изменение частоты сигнала F на выходе генератора G, что регистрируется процессором

один из ключей, выбирая нужный товар. Из всех протекающих при этом процессов нас будет интересовать только опознание позиции замкнутого ключа.

В данном примере число ключей равно 21,  $R=2,8 \text{ кОм}$ ,  $R'=720 \text{ Ом}$ ,  $C=0,1 \text{ мкФ}$ . Генератор  $G$  (типа NE555, ф. Texas Instruments) формирует непрерывную последовательность импульсов с частотой, зависящей от параметров внешней RC-цепи. Каждый ключ шунтирует больший или меньший участок цепи из последовательно включенных резисторов, поэтому в большей или меньшей степени меняется частота сигнала  $F$ .

Приоритеты ключей убывают слева направо, так что замыкание нескольких ключей эквивалентно замыканию одного из них — крайнего левого. Изменение периода сигнала  $F$  по отношению к исходному периоду, когда все ключи были разомкнуты, регистрируется процессором с использованием внутреннего таймера (на рисунке не показан). По результатам измерения соотношения периодов процессор определяет положение нажатой клавиши.

Резисторы  $R$ ,  $R'$  изготавливаются в едином технологическом процессе, поэтому относительный разбор их сопротивлений мал и слабо зависит от температуры и других дестабилизирующих факторов.

#### **1.4. Схема дешифрации кода, поступающего от источника импульсов инфракрасного излучения, размещенного на дистанционном пульте управления бытовым прибором [4]**

Для удобства пользования бытовыми приборами — телевизором, видеомagnetофоном и т. п. применяют пульты дистанционного управления. При нажатии на клавишу такого пульта формируется код команды, который преобразуется в последовательность импульсов инфракрасного излучения. Эти импульсы достигают приемника, размещенного в управляемом приборе, и преобразуются в последовательность электрических сигналов, которая дешифрируется, после чего полученная команда принимается к исполнению.

В предлагаемой системе передачи информации сигналы  $A$ , сформированные на выходе приемника инфракрасного излучения, после преобразования поступают на входы микроЭВМ, которая восстанавливает принятую команду и выполняет ее (рис. 1.4).

В паузах между командными послылками  $A=0$ , триггеры  $D1$  и  $D4$  установлены в 0, сигнал лог. 1 с выхода нулевого плеча триггера  $D1$  поступает на вход  $R$  установки 0 счетчика  $D3$  и поддерживает его выходы в состоянии лог. 0, сигнал на входе  $EN$  разрешения работы счетчика отсутствует (равен 0), триггер  $D2$  находится в произвольном состоянии, микроЭВМ обслуживает управляемый прибор (телевизор и т. п.).

Командная посылка представлена последовательностью импульсов, временные параметры которой приведены на рисунке. Одиночный импульс соответствует передаче лог. 0, а пара близлежащих импульсов — передаче лог. 1. В приведенном примере посылка такова: 0110...0.

Первый импульс командной посылки переводит триггер  $D1$  в состояние лог. 1, счетчик (таймер)  $D3$  начинает подсчет синхронимпульсов  $CLK$ , непрерывно поступающих из микроЭВМ на его счетный вход  $CL$ . Через время, равное  $1,1 \text{ мс}$ , на верхнем выходе счетчика (точнее, на выходе встроенного в него дешифратора некоторой кодовой комбинации) формируется сигнал лог. 1, фронт которого вызывает прием информации в триггер данных  $D2$ , триггер прерывания  $D4$  при этом устанавливается в 1. В данном случае (см. момент  $t$  на временной диаграмме) в триггер  $D2$  принимается лог. 0, так как при поступлении на вход  $S$  этого триггера положительного фронта сигнала на его  $D$ -входе присутствует сигнал  $A=0$ .

Процессор микроЭВМ, получив сигнал  $INT$  прерывания, временно приостанавливает выполнение текущего задания, запоминает параметры возврата к нему и приступает к выполнению прерывающей программы, в которой

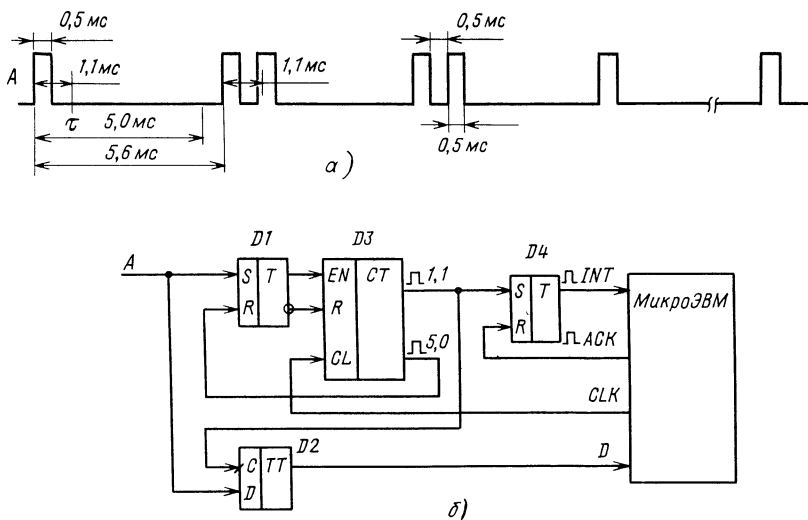


Рис. 1.4. Временная диаграмма сигналов А с выхода приемника инфракрасного излучения (а) и схема (б) дешифрации этих сигналов

предписано прочесть бит D из входного порта. Выполнив это действие, процессор «упаковывает» полученный бит в память и затем программно формирует импульсный ответный сигнал ACK в выходном порте микроЭВМ, триггер D4 устанавливается в 0. После завершения работы по прерывающей программе процессор возвращается к выполнению прерванного задания.

По истечении 5 мс от момента пуска счетчика на его нижнем выходе формируется сигнал лог. 1, который устанавливает триггер D1 в 0. Счетчик возвращается в нулевое состояние, схема готова к приему следующего разряда команды. Далее процессы повторяются.

В течение интервала времени между моментом приема информации в триггер D2 и моментом установки триггера D1 в 0 схема нечувствительна к помехам, которые могут появиться во входном сигнале А. Другое привлекательное качество схемы связано с предоставлением процессору некоторой «свободы действий» в пределах периода следования сигналов D, равного 5,6 мс. Иными словами, не обязательно, чтобы процессор «мгновенно» реагировал на сигнал прерывания INT=1. Он может запретить реакцию на сигнал прерывания (с помощью внутренней схемы маскирования), занимаясь выполнением более приоритетного задания, и снять запрет позже, но в пределах периода длительностью 5,6 мс. Это обеспечивает непрерывность обслуживания управляемого от микроЭВМ прибора в процессе получения команд от дистанционного пульта управления.

## 1.5. Метод и аппаратура кодирования—декодирования цифровой информации при ее записи—считывании с бытового кассетного магнитофона [5]

Для записи цифровой информации от источника (например, персональной микроЭВМ) на кассетный магнитофон предлагается использовать схему кодирования, представленную на рис. 1.5; временные диаграммы ее работы показаны на рис. 1.6.

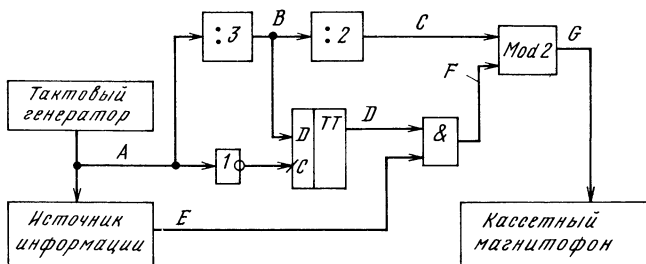


Рис. 1.5. Схема кодирования информации при ее записи на магнитную ленту

Источник информации формирует выходной сигнал Е синхронно с сигналом А тактового генератора — каждый битовый интервал соответствует трем периодам сигнала А. Информационные сигналы Е=0 и Е=1 представлены соответственно напряжениями низкого и высокого уровней. Схема деления на три формирует сигнал В, который задерживается на треть битового интервала D-триггером. Схема деления на два (счетный триггер) преобразует сигнал В в сигнал С, который суммируется по модулю два с сигналом  $F = D \oplus E$  схемой Иключающее ИЛИ (Mod 2). Результирующий сигнал G записывается на магнитную ленту. Отметим, что интервал времени, когда  $B=1$ , должен соответствовать началу (а не середине или концу) битового интервала сигнала Е. Поэтому либо нужна соответствующая синхронная начальная установка внутренних схем источника информации и схемы деления на три, либо, в отсутствие линии начальной установки, сигнал В должен поступать в источник информации, чтобы тот «знал», когда начинаются битовые интервалы.

При считывании информации с кассетного магнитофона она декодируется схемой, представленной на рис. 1.7. Временные диаграммы ее работы показаны на рис. 1.8. С помощью формирователей импульса F1 и F2 выделяются положительный и отрицательный фронты считанного сигнала Н, затем полученные короткие положительные импульсы суммируются элементом ИЛИ. Формирователь импульса F3 вырабатывает отрицательный (по фазе)

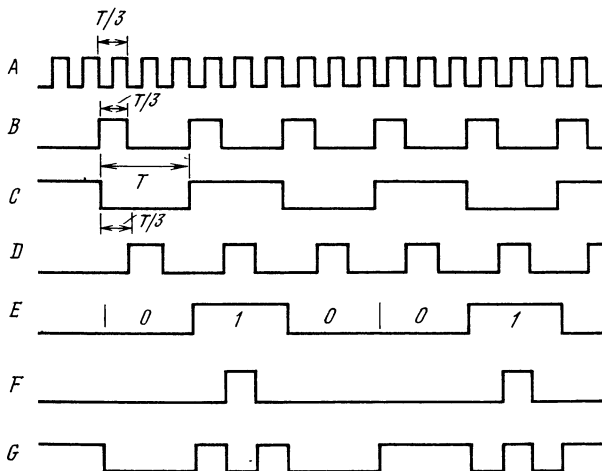


Рис. 1.6. Временные диаграммы работы схемы кодирования (рис. 1.5)

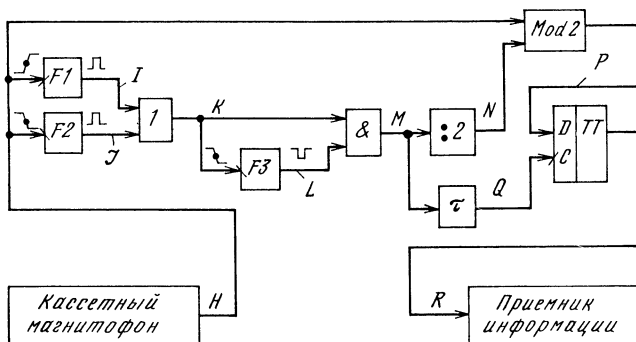


Рис. 1.7. Схема декодирования информации при ее считывании с магнитной ленты

сигнал  $L$  достаточно большой длительности при переходе сигнала  $K$  из 1 в 0. Пока  $L \neq 0$ , формирователь импульса  $F3$  нечувствителен к входным сигналам. Поэтому он в процессе работы адаптируется к входным импульсам  $K$ , выделяя в каждой встречающейся пачке (четверке) импульсов только первый и последний. Одиночные импульсы  $K$  также приводят к срабатыванию формирователя импульса  $F3$ . В результате формируется сигнал  $M = K \& L$ , положительный фронт которого соответствует середине битового интервала. Далее последующие элементы схемы восстанавливают исходный сигнал (сравните временные диаграммы сигналов  $R$  и  $E$  на приведенных рисунках).

Отметим, что результат  $P$  суммирования по модулю два сигналов  $H$  и  $N$  будет содержать кратковременные положительные всплески напряжения (на рисунке не показаны) из-за неидеального совпадения фронтов обоих входных сигналов. Это, однако, не приводит к сбоям, так как в моменты стробирования  $D$ -триггера сигналом  $Q$  сигнал  $P$  принимает правильное установившееся значение.

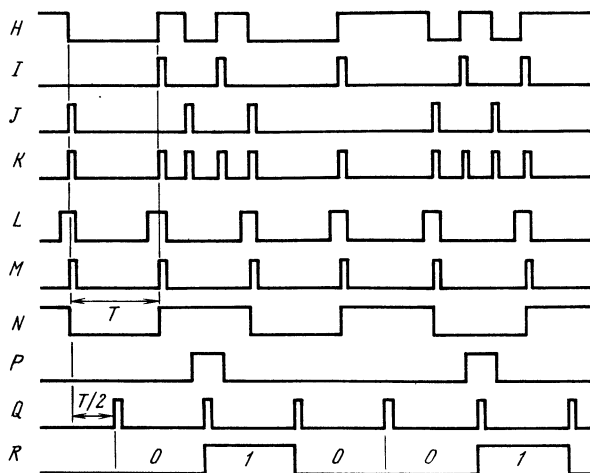


Рис. 1.8. Временные диаграммы работы схемы декодирования (рис. 1.7)

По сравнению с применяющимися в персональных микроЭВМ методами кодирования — декодирования, например методом FSK (frequency shift keying), который предусматривает передачу лог. 0 четырьмя периодами сигнала частотой 1200 Гц, а лог. 1 восемью периодами сигнала частотой 2400 Гц, данный метод позволяет повысить плотность записи информации и обеспечить большую устойчивость к вариациям скорости протяжки магнитной ленты.

## 1.6. Коммутатор линий интерфейса RS-232 [6]

В схеме рис. 1.9 дисплей подключается к одной из семи удаленных микроЭВМ с помощью коммутатора (показано соединение дисплея с микроЭВМ2). Сигналы между дисплеем и выбранной оператором микроЭВМ передаются по двум витым парам проводов; в каждой паре один провод экранирующий (он соединен с шинами нулевого потенциала дисплея и выбранной микроЭВМ), второй — сигнальный.

Каждая витая пара проводов представляет собой однонаправленную линию связи для передачи данных; TD — выход передаваемых, RD — вход принимаемых данных. В соответствии с интерфейсом RS=232 лог. 1 и лог. 0 представлены в сигнальных проводах напряжениями +12 и -12 В относительно проводов нулевого потенциала.

Непосредственная коммутация линий связи с помощью механических контактных элементов (кнопочного или галетного переключателя, реле и т. п.) ненадежна и неэкономична, так как велико число коммутируемых контактов. В предлагаемом решении (рис. 1.10) механический переключатель SW использован только для выбора микроЭВМ, а сигнальные цепи коммутируются «бесконтактно».

Приемники D1 — D8 (две микросхемы типа MC1489) преобразуют сигналы, поступающие из линий связи, в сигналы ТТЛ-уровней: +12 В → ТТЛ лог. 1, -12 В → ТТЛ лог. 0; передатчики D11 — D18 (две микросхемы типа MC1488) осуществляют обратное преобразование — на их выходах формируются сигналы с уровнями +12 и -12 В.

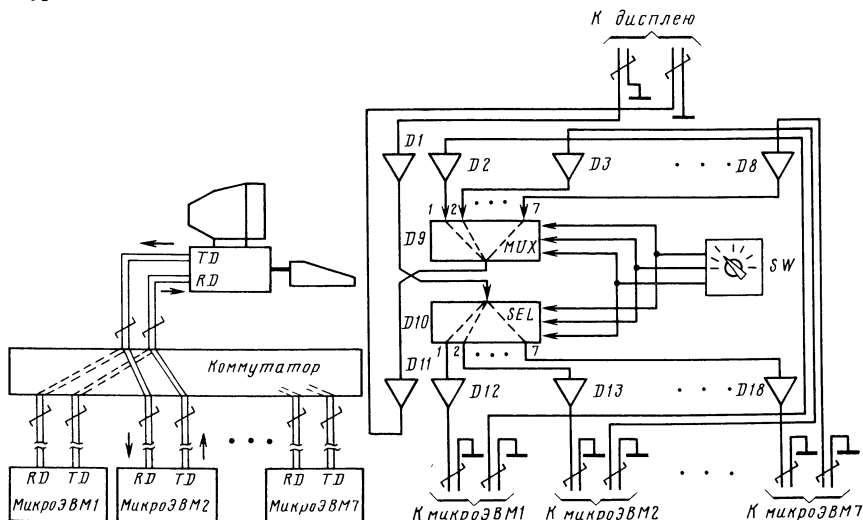


Рис. 1.9. Схема связи дисплея с удаленными микроЭВМ

Рис. 1.10. Схема коммутатора

С галетного переключателя SW на управляющие входы ТТЛ-мультиплексора D9 и ТТЛ-селектора D10 поступает 3-разрядный код, в соответствии с которым выбирается нужный канал. Точнее, передаваемые из дисплея данные проходят через приемник D1, селектор D10 и соответствующий передатчик из группы D12—D18 к выбранной микроЭВМ; данные от нее поступают через один из приемников D2—D8 на вход мультиплексора D9 и с его выхода через передатчик D11—на вход дисплея.

## 1.7. Увеличение числа информационных входов и выходов микроЭВМ подключением внешних сдвиговых и параллельных регистров (по материалам [7])

МикроЭВМ, показанная на рис. 1.11, содержит порт для подключения алфавитно-цифрового печатающего устройства (АЦПУ) или иного приемника информации. Обычно такой порт включает восемь выходных линий для передачи информации (использованы три линии, условно названные CL, OE, D/C) и линию готовности внешнего устройства к приему очередного кода, которая в данном примере объединена с линией D/C. Другие линии порта,

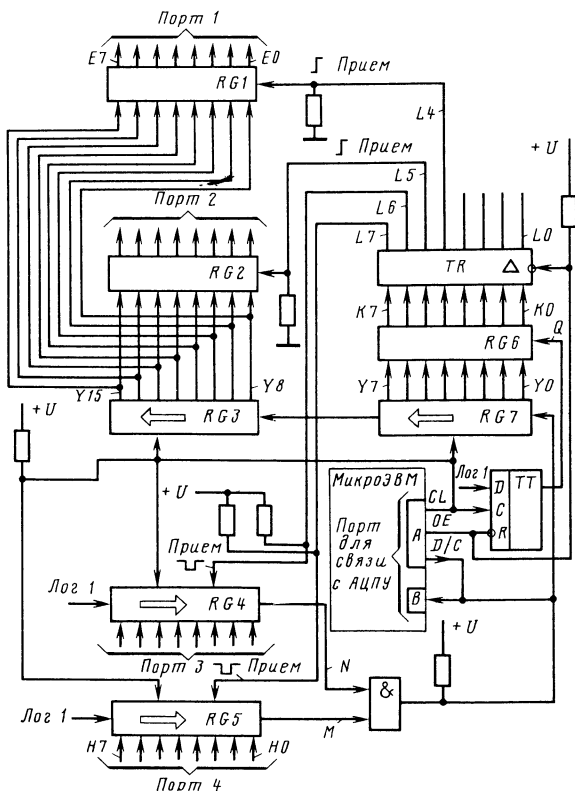


Рис. 1.11. Подключение к микроЭВМ дополнительных портов ввода—вывода

такие как линия подтверждения истинности передаваемых данных, линия готовности к передаче информации, линия начальной установки внешнего устройства и прочие, в данном примере не используются. Задача заключается в увеличении числа информационных входов и выходов микроЭВМ.

Порт микроЭВМ для связи с АЦПУ с точки зрения программиста представлен двумя независимыми программно-доступными (адресуемыми) регистрами — выходным регистром данных А и входным регистром состояния В (нас интересует только один разряд этого регистра). Программа работы с портом предусматривает хранение в ОЗУ текущей копии содержимого выходного регистра А, так что возможна коррекция этой копии с последующей ее записью в регистр А. Это позволяет индивидуально или одновременно менять значения сигналов в линиях CL, OE, D/C в нужной последовательности, т. е. создавать любые желаемые комбинации выходных сигналов в соответствии с заранее заданной временной диаграммой (подобные порты поэтому часто называют «программируемым интерфейсом»). Информация из входного регистра В поступает в процессор микроЭВМ при выполнении команды чтения этого регистра.

Выходы регистра А выполнены по схеме с открытым коллектором. При выдаче в этот регистр сигналов лог. 1 напряжение высокого уровня поддерживается на выходах за счет их подключения через резисторы к источнику питающего напряжения. Двухвходовой элемент И также выполнен по схеме с открытым коллектором. В исходном состоянии на обоих его входах присутствуют сигналы лог. 1, так что выходной транзистор этого элемента выключен и не препятствует передаче сигналов из регистра А в линию D/C.

Регистры RG1 и RG2 фиксируют входную информацию при поступлении положительных фронтов сигналов Прием. В исходном состоянии эти сигналы равны 0, так как соответствующие линии подключены через резисторы к шине нулевого потенциала, а усилители TR выключены, их выходное сопротивление высокое. Сдвиговые регистры RG7 и RG3 соединены последовательно и имеют общую линию синхронизации CL. Информация сдвигается влево по положительному фронту сигнала в линии CL. При каждом сдвиге в разряд Y0 регистра RG7 заносится информация с линии D/C.

Регистры RG4 и RG5 работают в режиме параллельного приема и последующего сдвига вправо принятой информации. Прием или сдвиг осуществляются по положительному фронту сигнала в линии CL, выбор режима работы регистров определяется значениями сигналов Прием. В исходном состоянии эти сигналы равны 1 (режим сдвига), так как соответствующие линии подключены через резисторы к источнику питающего напряжения, а выходы усилителей TR отключены от линий. На информационные входы сдвига регистров RG4 и RG5 поданы сигналы лог. 1, так что после ряда последовательных сдвигов оба регистра заполняются единицами. Далее предполагается, что к началу процедур обмена информацией с дополнительно введенными портами регистры RG4 и RG5 уже заполнены единицами,  $N=M=1$ , элемент И с открытым коллектором находится в пассивном состоянии и не шунтирует линию D/C.

Регистр-защелка RG6 при  $Q=1$  транслирует информацию с входов на выходы, а при переходе сигнала Q из 1 в 0 фиксирует передаваемую информацию и хранит ее на протяжении периода действия сигнала  $Q=0$ . Триггер ТГ устанавливается в 0 сигналом лог. 0 на входе R. Запись лог. 1 в этот триггер осуществляется по положительному фронту сигнала на входе С при условии отсутствия сигнала лог. 0 на входе R.

Схема, приведенная на рис. 1.11, позволяет увеличить число информационных входов и выходов микроЭВМ, а именно дополнительно ввести два выходных и два входных 8-разрядных порта (порты 1, 2; 3, 4). Рассмотрим в качестве примера процессы передачи информации из микроЭВМ в выходной порт 1 и считывания информации в микроЭВМ из входного порта 4.

При передаче информации из микроЭВМ в порт 1 в линиях CL, OE и D/C программно генерируются последовательности сигналов, показанные

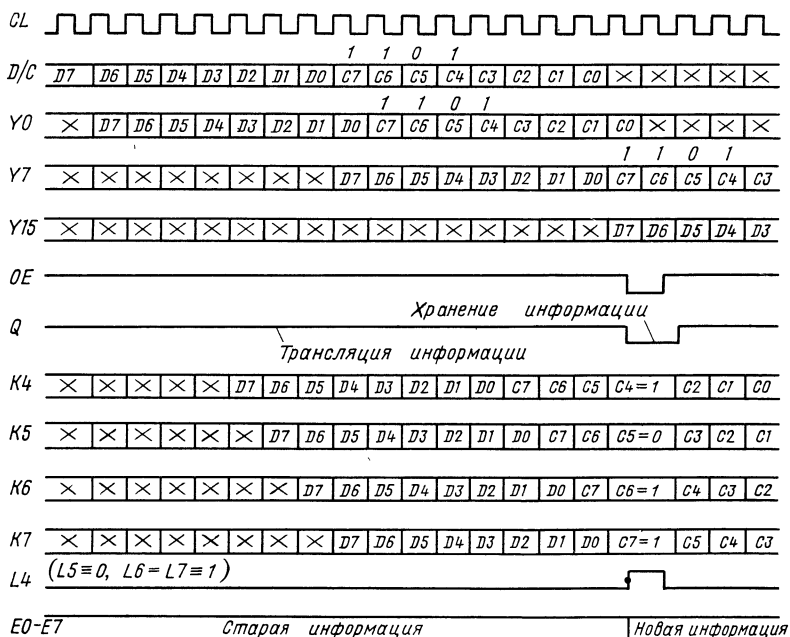


Рис. 1.12. Временные диаграммы работы схемы (рис. 1.11) при выдаче информации в порт 1.

× — произвольное значение сигнала

на рис. 1.12. В линию D/C последовательно выдаются данные D7—D0, а затем сигналы управления C7—C0. В линии CL формируется последовательность синхриопульсов, возможно, с переменной скважностью (на рисунке скважность постоянна и равна двум). После выполнения серии сдвигов данные D7—D0 размещаются в регистре RG3 ( $Y15=D7$ ,  $Y14=D6$ , ...,  $Y8=D0$ ), а сигналы управления—в регистре RG7 ( $Y7=C7$ ,  $Y6=C6$ , ...,  $Y0=C0$ ). При этом сигнал  $OE=0$  разрешает выдачу управляющих воздействий на выходы L7—L0, регистр RG6 временно запоминает сигналы управления во избежание возникновения «гонок» между информационными и управляющими сигналами.

В данном примере  $C7=C6=1$ ,  $C5=0$ ,  $C4=1$ , поэтому на выходах L7—L4 формируется комбинация сигналов 1 1 0 1, отличающаяся от исходной (1 1 0 0) только в одном разряде. В результате по положительному фронту сигнала на выходе L4 в регистр RG1 заносится информация  $E7=D7$ ,  $E6=D6$ , ...,  $E0=D0$ , первоначально хранившаяся в памяти микроЭВМ, что и требовалось. Выдача информации в порт 2 производится аналогично при формировании на выходах L7—L4 сигналов 1110.

Для правильной работы сдвиговых регистров положительный фронт сигнала в линии CL должен предшествовать смене информации в линии D/C. Это соотношение сигналов обеспечивается программным или аппаратным способом, т. е. либо введением в программу последовательных команд смены информации в разрядах CL и D/C регистра A, либо подключением элемента с небольшой задержкой между выходом регистра A и линией D/C (при этом программа может менять значения указанных разрядов одновременно). На временных диаграммах (рис. 1.12) эта задержка не отражена.

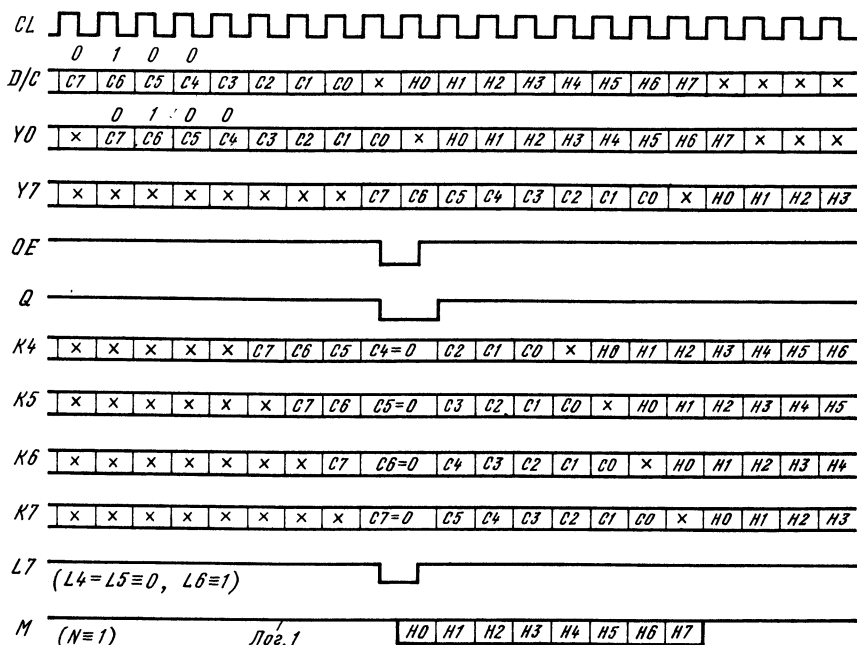


Рис. 1.13. Временные диаграммы работы схемы (рис. 1.11) при считывании информации из порта 4.

× — произвольное значение сигнала

При опросе порта 4 входная информация Н7—Н0 сначала заносится в регистр RG5, а затем последовательно сдвигается вправо, проходит через элемент И и попадает в линию D/C. Состояние этой линии анализируется процессором микроЭВМ после программного считывания содержимого регистра В. Разряд D/C выходного порта А при сдвиге информации в регистре RG5 программно установлен в 1, так что линией D/C управляет только элемент И. Более детально процесс считывания информации из порта 4 отражен на рис. 1.13.

Сначала микроЭВМ выдает в регистр RG7 последовательный управляющий код С7...С0, причем в данном примере С7=0, С6=1, С5=С4=0, остальные разряды С3—С0 (используются для расширения системы) могут иметь произвольные значения. Загрузка регистра RG7, фиксация кода в регистре RG6 и выдача управляющих воздействий в линии L4—L7 выполняются примерно так же, как и при записи во внешний порт (см. рис. 1.12), отличие состоит в числе циклов сдвига (8 вместо 16), так как сначала выдается управляющий код, а затем принимаются данные.

В результате действия сигнала OE=0 в линиях L7—L4 устанавливается код 0100, отличающийся от исходного (1100) в левом разряде. Сигнал лог. 0 в линии L7 переводит регистр RG5 в режим сдвига. По положительному фронту сигнала CL в этот регистр заносится информация Н7—Н0 из порта 4. Затем регистр RG5 переводится в режим сдвига, в результате информация последовательно продвигается через регистр В в процессор, который программно компонует в памяти 8-разрядный код Н7—Н0, что и требовалось. Ввод информации из порта 3 осуществляется аналогично, с использованием управляющего кода 1000 в линиях L7—L4.

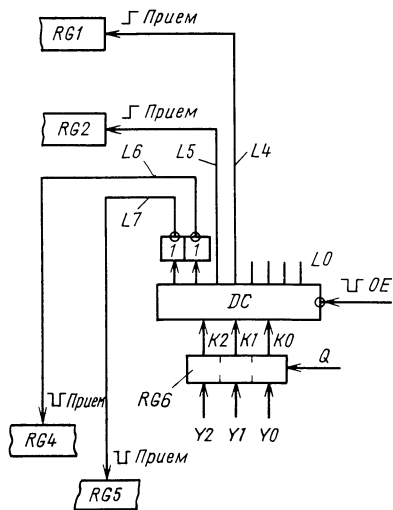


Рис. 1.14. Усовершенствование схемы (рис. 1.11) для уменьшения разрядности управляющего кода.

Вместо усилителей TR (рис. 1.11) введен дешифратор DC, разрядность регистров RG6 и RG7 уменьшена от 8 до 3

Для уменьшения числа управляющих разрядов  $C_i$  или, что по сути то же самое, для увеличения числа портов ввода—вывода с сохранением разрядности управляющего кода можно использовать дешифратор, как показано на рис. 1.14. Управляющий код содержит три разряда, на одном из восьми выходов дешифратора DC при поступлении сигнала  $OE=0$  (разрешения его работы) формируется сигнал лог. 1, который при необходимости инвертируется и воздействует на соответствующий регистр. При  $OE=1$  работа дешифратора запрещена—на всех его выходах сформированы активные сигналы лог. 0. Если сохранить

разрядность управляющего кода равной 8, то с использованием дешифраторов можно получить  $2^8=256$  сигналов управления внешними портами, число которых составит 256.

## 1.8. Повышение надежности передачи команды Сброс от главной микроЭВМ к подчиненной по последовательному каналу связи [8]

В двухмашинной системе (рис. 1.15) взаимодействие между главной и подчиненной микроЭВМ осуществляется по линии связи из двух витых пар проводов с использованием стандартного последовательного интерфейса, например типа ИРПС. При нормальной работе системы микроЭВМ обмениваются информацией в соответствии с некоторым протоколом. Элементарная кодовая посылка А (В) содержит стартовый, 7 информационных и стоповый биты. Информация в каждой паре проводов линии связи представлена импульсами тока. Схемы оптронной развязки повышают помехозащищенность системы связи.

В режиме приема информации из главной микроЭВМ стандартный приемопередатчик преобразует полученный из линии связи последовательный код в параллельный и формирует в своем регистре состояния признак готовности кода. Центральный процессор (ЦП) обнаруживает этот признак при периодической программной проверке регистра состояния приемопередатчика или при получении сигнала прерывания в связи с готовностью кода. Далее процессор считывает код из регистра данных приемопередатчика, при этом признак готовности кода автоматически гасится.

Проблема заключается в повышении надежности передачи команды Сброс из главной микроЭВМ к подчиненной. Если бы эти микроЭВМ находились рядом с оператором, то он мог бы при необходимости привести систему в исходное состояние нажатием соответствующих кнопок, размещенных на корпусах обеих микроЭВМ. Однако подчиненная микроЭВМ удалена на значительное расстояние, поэтому приходится использовать дистанционное управление. Точнее, команда Сброс передается

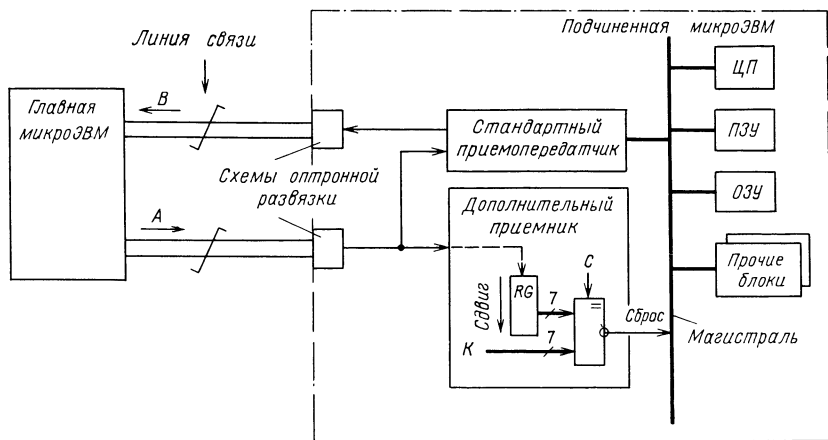


Рис. 1.15. Двухмашинная система.

Дополнительный приемник настроен на поиск в проходящем потоке информации кода К, при обнаружении которого формируется сигнал Сброс

в виде некоторого последовательного 7-разрядного кода «на общих основаниях».

Здесь наблюдается следующее противоречие. С одной стороны, для восприятия команды Сброс подчиненная микроЭВМ должна быть работоспособной, ЦП должен периодически опрашивать регистр состояния приемопередатчика или быть способным правильно воспринимать и реагировать на сигнал прерывания, свидетельствующий о готовности кода. После получения кода команды Сброс ЦП должен сформировать импульс в линии Сброс магистрали, чтобы привести все устройства, подключенные к магистрали, в исходное состояние. С другой стороны, может случиться так, что команда Сброс передается из главной микроЭВМ в подчиненную именно в связи с тем, что последняя работает неправильно или вообще «зависла». А если это так, то команда Сброс, переданная по каналу связи, может просто не восприняться подчиненной микроЭВМ.

Другими словами, если подчиненная микроЭВМ в результате сбоя, ошибки при загрузке программы или по каким-либо иным причинам потеряла способность обрабатывать сигналы, поступающие из приемопередатчика, то установить ее в исходное состояние по линии связи становится невозможно.

Чтобы повысить надежность восприятия команды Сброс, в подчиненную микроЭВМ введен дополнительный приемник. Он получает все элементарные кодовые посылки А (параллельно со штатным приемопередатчиком), поступающие по линии связи, и преобразует их в сдвиговом регистре RG в параллельные 7-разрядные коды. Эти коды сравниваются с кодом К, отображающим команду Сброс, с помощью компаратора. Код К может быть задан подключением соответствующих входов компаратора к шинам лог. 0 и лог. 1. Сигнал С разрешения работы компаратора формируется в момент готовности очередного кода в регистре RG.

При обнаружении команды Сброс на выходе компаратора формируется отрицательный импульс, который передается в соответствующую линию магистрали. Таким образом, «интеллект» подчиненной микроЭВМ может в данный момент полностью отсутствовать, тем не менее эта микроЭВМ будет приведена в рабочее состояние при воздействии сигнала по магистральной линии Сброс.

## 1.9. Обмен данными между двумя удаленными друг от друга микроЭВМ по двухпроводной линии связи [9]

В схеме рис. 1.16 микроЭВМ1—главная, микроЭВМ2—подчиненная; они взаимодействуют между собой через приемопередатчики и линию связи. МикроЭВМ2 обслуживает некоторый управляемый объект. МикроЭВМ1 посылает в линию команды, микроЭВМ2 получает их и передает в микроЭВМ1 ответные сигналы. Протокол взаимодействия не представляет для нас интереса; покажем лишь принципиальную возможность двусторонней связи.

В схеме применены транзисторные V2, V4 и тиристорные V1, V3 оптопары для гальванической развязки линии связи с приемопередатчиками, как в обычных «токовых петлях» (это повышает помехозащищенность системы). Транзистор оптопары V2 (V4) насыщен, когда светодиод включен; если свет не излучается, то транзистор выключен. Тиристор оптопары V1 (V3) включается при наличии светового потока от светодиода, если напряжение анод—катод положительно, и выключается при снижении тока через тиристор до некоторого порогового значения.

Генератор G переменного напряжения (например, вторичная обмотка силового трансформатора) питает цепи сопряжения приемопередатчиков с линией связи. Его можно физически разместить в первом или втором приемопередатчике (мысленно переместите точки A и B влево или вправо) либо включить «в разрыв» линии, как, по существу, и показано на рисунке (устраните незамкнутые отрезки линии).

В установившемся режиме микроЭВМ1 периодически посылает в микроЭВМ2 данные H1. При включении транзистора VT1 светодиод оптопары V1 излучает свет, поэтому во время действия положительной полуволны переменного напряжения ( $U_{AB} > 0$ ) по цепи точка A—тиристор оптопары V1—резистор R3—провод F—резистор R9—светодиод оптопары V4—точка B протекает ток  $i$  (рис. 1.17, а). На выходе оптопары V4 формируются сигналы H5, которые поступают на информационный вход микроЭВМ2 (или, например, на вход прерывания) и воспринимаются ею.

Ответные сигналы H3 от микроЭВМ2 передаются в микроЭВМ1 во время действия отрицательных полуволн переменного напряжения ( $U_{AB} < 0$ ) при срабатывании транзистора VT2 и распространяются по цепи: точка B—тиристор оптопары V3—резистор R6—провод F—резистор R5—светодиод оптопары V2—точка A (ток  $i$  отрицателен). МикроЭВМ1 принимает

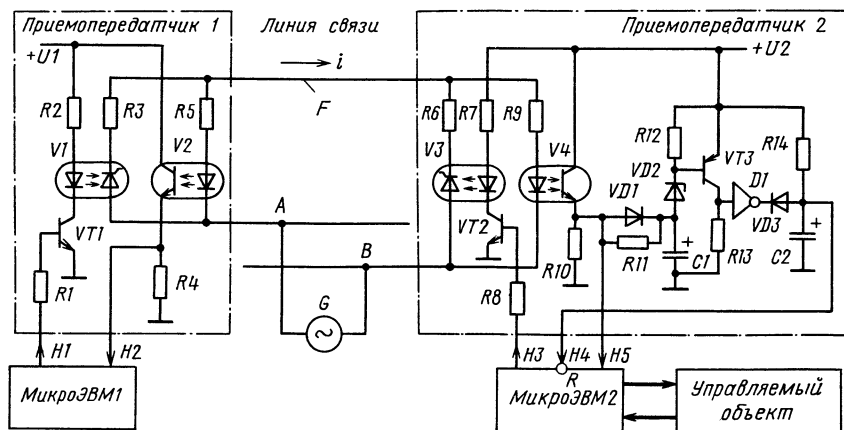


Рис. 1.16. Схема связи между двумя микроЭВМ

сигналы Н2 и интерпретирует их в соответствии с принятым в системе протоколом обмена.

При нормальной работе системы конденсатор С1 заряжен до некоторого достаточно высокого уровня напряжения  $U_{C1}$  (см. рис. 1.17, б, период времени до момента  $t_0$ ), стабилитрон VD2 и транзистор VT3 выключены, на входе инвертора D1 присутствует напряжение, близкое к 0, на его выходе сигнал лог. 1, диод VD3 закрыт, напряжение на конденсаторе C2 близко к 0 ( $H4=1$ ), что соответствует отсутствию сигнала R сброса микроЭВМ2.

Если по тем или иным причинам микроЭВМ1 сочла необходимым осуществить перезапуск микроЭВМ2, то в момент  $t_0$  она прекращает выдачу сигналов Н1. Ток  $i$  в линии прекращается (по крайней мере, в периоды действия положительных полуволн переменного напряжения), транзистор оптопары V4 выключается, конденсатор C1 начинает разряжаться через резисторы R11 и R10, напряжение на базе транзистора VT3 снижается.

В момент  $t_1$  транзистор VT3 открывается, на вход инвертора D1 передается напряжение высокого уровня, сигнал лог. 0 с выхода этого инвертора проходит через диод VD3 и поступает на вход сброса микроЭВМ2. С возобновлением периодической посылки из микроЭВМ1 сигналов Н1 конденсатор C1 заряжается через диод VD1, транзистор VT3 закрывается, сигнал H4 сброса снимается, система переходит в нормальный режим работы. Таким образом, без использования дополнительной линии связи гарантируется надежная передача сигнала сброса из микроЭВМ1 в микроЭВМ2. Кроме того, сигнал сброса микроЭВМ2 формируется цепью R14C2 при включении напряжения питания  $+U_2$ .

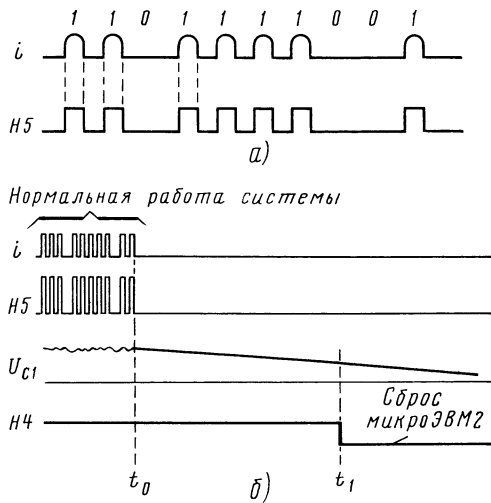


Рис. 1.17. Временные диаграммы работы системы (рис. 1.16)

## 1.10. МикроЭВМ с групповым обслуживанием контроллеров [10]

МикроЭВМ (рис. 1.18) содержит ЦП, ряд контроллеров K0—K18 и другие блоки (на рисунке не показаны). Контроллеры первой группы (K0—K15) имеют один вход и один выход для подключения к простейшим внешним устройствам, например датчику двоичного сигнала и исполнительному реле; контроллеры второй группы (K16—K18) служат для сопряжения с внешними устройствами, имеющими 5 входов и 5 выходов.

Задача состоит в том, чтобы процессор с помощью одной команды чтения мог опросить сразу все внешние входы контроллеров выбранной группы, а с помощью одной команды записи—передать одновременно на все внешние выходы нужные коды. Решение этой задачи основано на том, что при взаимодействии с процессором все контроллеры одной группы реагируют на один и тот же адрес и активизируются одновременно, но используют разные разряды шины данных. Распределение разрядов этой шины между контроллерами осуществляется процессором при выполнении программы начального пуска после включения напряжения питания.

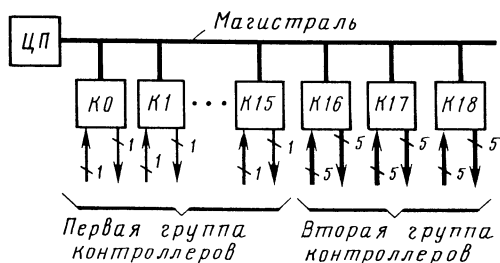


Рис. 1.18. МикроЭВМ с групповым обслуживанием контроллеров

Контроллер первой группы (рис. 1.19) реагирует на один индивидуальный и два общих (групповых) адреса. Эти адреса распознаются дешифратором DC1 в момент поступления на его вход ВК сигнала  $MSYN=0$  (напряжение низкого уровня), подтверждающего истинность кода, установленного процессором в адресной шине (в данном примере используется интерфейс Общая шина).

Индивидуальные адреса контроллеров позволяют процессору загрузить с шины данных D их регистры RG уникальными 4-разрядными кодами, которые определяют, какая линия этой шины выделяется каждому из них при групповом обмене данными с процессором. При загрузке регистра RG дешифратор DC1 формирует сигнал  $F1=1$ , по которому данные, установленные процессором в линиях D0—D3, принимаются в регистр. На выходе элемента ИЛИ—НЕ формируется ответный сигнал  $SSYN=0$ , подтверждающий факт выполнения операции записи. Получив этот сигнал, процессор освобождает магистраль.

После загрузки кода в регистр RG мультиплексор MS настроен на прием, а дешифратор DC2—на выдачу сигнала в некоторую линию D<sub>i</sub> шины данных при взаимодействии с процессором в режиме группового обращения. Дешифратор DC2 выполнен по схеме с открытым коллектором. Если хотя бы на одном его входе ВК присутствует сигнал лог. 0, то все его выходы отключены от магистрали. При  $F2=G=1$  дешифратор формирует напряжение низкого уровня в соответствующей линии шины D. Нагрузочные резисторы (или резисторные делители напряжения), подключенные к физическим концам магистрали, обеспечивают поддержание напряжения высокого уровня в ее линиях в пассивном состоянии (эти резисторы на рисунках не показаны).

При выполнении команды чтения 16-разрядного слова по некоторому (групповому) адресу на среднем выходе дешифратора DC1 формируется сигнал  $F2=1$ . В зависимости от сигнала G, поступающего от внешнего устройства, дешифратор DC2 либо остается в пассивном состоянии, либо формирует в линии D<sub>i</sub> выделенной данному контроллеру, напряжение низкого уровня. Аналогичные процессы протекают одновременно в контроллерах K0—K15, в результате в шине данных D формируется 16-разрядный код, каждый разряд которого отражает состояние выхода соответствующего внешнего устройства.

Этот код считывается процессором через 150 нс (или более) после получения им ответного сигнала  $SSYN=0$ . Согласно интерфейсу Общая шина сигнал  $SSYN=0$  должен формироваться не раньше, чем считанные данные поступят в шину D. Поэтому самый «быстрый» элемент ИЛИ—НЕ в группе (сигналы  $SSYN$  объединяются в общей управляющей линии типа «открытый коллектор») должен иметь большую задержку, чем самый «медленный» дешифратор DC2.

При выполнении команды записи 16-разрядного слова по групповому адресу на нижнем выходе дешифратора DC1 формируется сигнал  $F3=1$ . Разряд слова данных, выбранный мультиплексором MS, фиксируется в D-триггере и в виде сигнала Н поступает во внешнее устройство. Ответные сигналы  $SSYN=0$  всех контроллеров K0—K15 суммируются в общей управляющей линии и поступают в процессор, который освобождает магист-

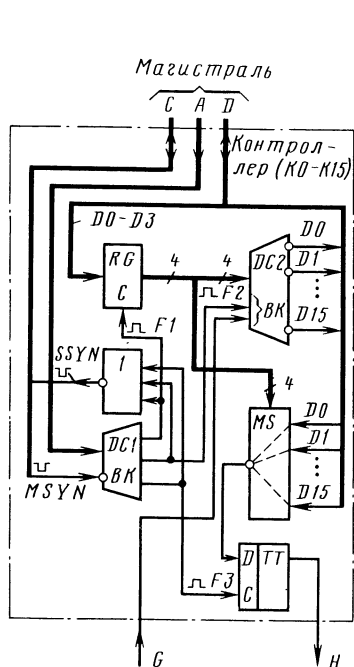


Рис. 1.19. Структура контроллера первой группы

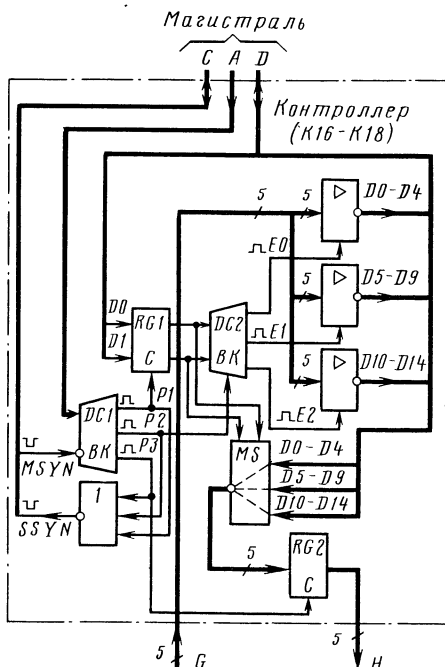


Рис. 1.20. Структура контроллера второй группы

раль, завершая выполнение операции записи. В данной схеме для ее упрощения не использована линия «чтение—запись» (C1) управляющей шины С магистральной: первый групповой адрес по договоренности с программистом использован только для чтения, второй—только для записи.

Контроллер второй группы (рис. 1.20) построен аналогично. Сигналы P1—P3 соответствуют сигналам F1—F3 предыдущей схемы. Дешифратор DC2 при P2=0 поддерживает на выходах сигналы лог. 0, а при выполнении операции группового чтения (P2=1) формирует один из сигналов E0—E2, который разрешает работу одного из 5-разрядных магистральных усилителей с открытым коллектором. При выполнении операции групповой записи в регистре RG2 каждого из контроллеров K16—K18 фиксируется 5-разрядный код с выходов мультиплексора MS, предварительно настроенного на передачу нужной группы сигналов с шины D.

### 1.11. Подключение группы контроллеров к порту ввода—вывода микроЭВМ [11]

Для увеличения числа устройств ввода—вывода, подключаемых к микроЭВМ, можно воспользоваться схемой (рис. 1.21), в которой стандартный порт обслуживает дополнительно введенную шину с подключенными к ней контроллерами K1—K<sub>N</sub>. Порт содержит программно-доступные выходные и входные регистры, например регистры данных и состояния канала связи с принтером. В данном случае вместо принтера или параллельно ему к порту подключена дополнительная подсистема ввода—вывода.

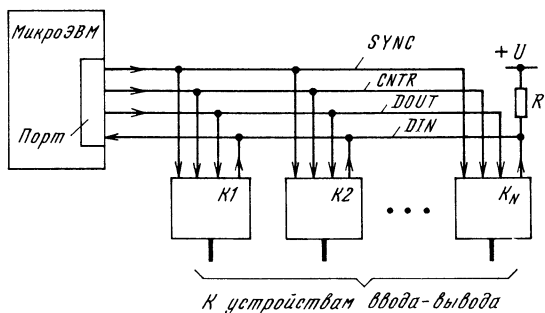


Рис. 1.21. Подключение контроллеров  $K_1$ — $K_N$  к микроЭВМ

Шина содержит три выходные (по отношению к микроЭВМ) и одну входную линию, по которым передаются сигналы синхронизации SYNC, управления CNTR, выходных и входных данных: DOUT и DIN. Линия передачи сигнала DIN выполнена по схеме с открытым коллектором; R—нагрузочный резистор.

Каждому контроллеру присвоен собственный адрес, который задается микропереключателями, перемычками, установленными на плате, или иным способом. Взаимодействие микроЭВМ с выбранным контроллером начинается с последовательной передачи по линии DOUT кода его адреса, после чего по линиям DOUT или (и) DIN в соответствующих направлениях пересылаются данные. Синхронизация передаваемых по линиям DOUT и DIN сигналов осуществляется сигналами SYNC, текущий режим обмена данными отображается сигналом CNTR. Рассмотрим примеры построения контроллеров.

В схеме контроллера (рис. 1.22, а), предназначенного для ввода данных в микроЭВМ, применены сдвиговые регистры D1, D4 (направления сдвига показаны стрелками), компаратор D2, мультиплексор D3 и элемент И—НЕ D5 с открытым коллектором. Регистр D1 при  $EN=0$  хранит данные и не реагирует на сигнал С. При  $EN=1$  под действием положительных фронтов сигнала С данные D поступают в регистр и продвигаются по нему сверху вниз.

Регистр D4 при  $LD=1$  транслирует данные D с входов на выходы и не реагирует на сигналы С. При переходе сигнала LD из 1 в 0 текущие данные фиксируются в регистре; в дальнейшем с поступлением положительных фронтов сигнала С данные сдвигаются сверху вниз и проходят на выход OUT (рис. 1.23).

В момент  $t_0$  (см. рис. 1.22, б) процессор микроЭВМ записывает в выходной регистр порта код, содержащий в разряде CNTR лог. 1, в разряде SYNC—лог. 0, в разряде DOUT—первый бит данных. Регистр D1 по сигналу CNTR=1 настраивается на прием и сдвиг данных D по положительным фронтам сигнала SYNC. Процессор микроЭВМ последовательно передает биты данных в регистр, формируя в линии DOUT соответствующие сигналы (лог. 0, лог. 1) и подтверждая их истинность положительными фронтами сигнала SYNC. Загрузка регистра D1 завершается в момент  $t_1$ .

В процессе загрузки регистра D1 (период  $t_0—t_1$ ) регистр D4 находится в режиме передачи входных данных D на выходы, в частности на выход OUT нижнего разряда. Поскольку эти данные могут изменяться, сигнал OUT в период  $t_0—t_1$  не определен, а так как при загрузке регистра D1 на входах компаратора D2 могут возникать случайные совпадения кодов (ID—собственный адрес контроллера), то элемент И—НЕ D5 может срабатывать в произвольные моменты, сигнал DIN также не определен (см. заштрихованную область на временной диаграмме этого сигнала). Напомним, что выходы элементов И—НЕ D5 во всех контроллерах выполнены по схеме с открытым коллектором, поэтому при одновременном срабатывании двух или более таких элементов их электрических перегрузок не возникает.

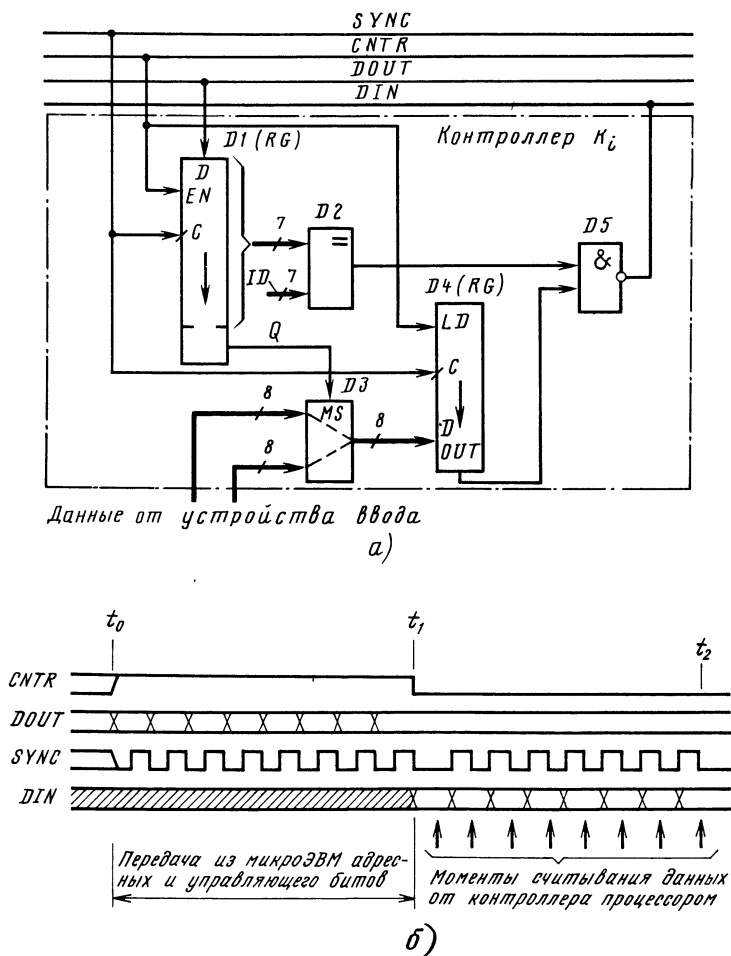


Рис. 1.22. Структура контроллера  $K_i$  (первый вариант) (а) и временные диаграммы его работы (б)

В момент  $t_1$  загрузка регистров  $D1$  во всех контроллерах закончена, регистры  $D4$  фиксируют 8-разрядные коды, поступающие с выходов мультиплексоров  $D3$ . Направление передачи данных через мультиплексор задается сигналом  $Q$ , принятым из микроЭВМ.

В одном из контроллеров, а именно в том, чей собственный адрес  $ID$  совпал с загруженным в регистр  $D1$  кодом, элемент И—НЕ  $D5$  открыт по верхнему входу, в остальных контроллерах эти элементы выключены и не препятствуют передаче данных из выбранного контроллера через порт в микроЭВМ. В период  $t_1$ — $t_2$  процессор периодически опрашивает входной регистр порта и программно формирует импульсные сигналы  $SYNC$ , по положительным фронтам которых данные в регистре  $D4$  сдвигаются вниз. Таким образом, схема (рис. 1.22, а) позволяет процессору считать выбранный байт данных с внешних входов адресуемого контроллера.

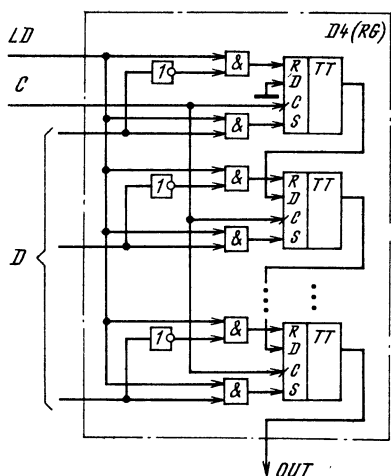


Рис. 1.23. Пример выполнения регистра D4 (рис. 1.22, а)

Схема (рис. 1.24, а) предназначена для передачи данных к устройству вывода. В период  $t_0 - t_1$  (рис. 1.24, б) в регистр D1 последовательно загружают адрес контроллера; регистр D5 хранит старые данные, так как элемент И D4 закрыт по верхнему входу (EN=0). При совпадении полученного адреса с собственным (ID) на выходе компаратора D3 формируется сигнал лог. 1. Начиная с момента  $t_1$  в выбранном контроллере разрешена работа регистра D5, который к моменту  $t_2$  заполняется новыми данными, поступающими из порта микроЭВМ. Смена данных на внешних выходах контроллера не «мгновенна» — в процессе загрузки регистра D5 они нестабильны, что не всегда приемлемо.

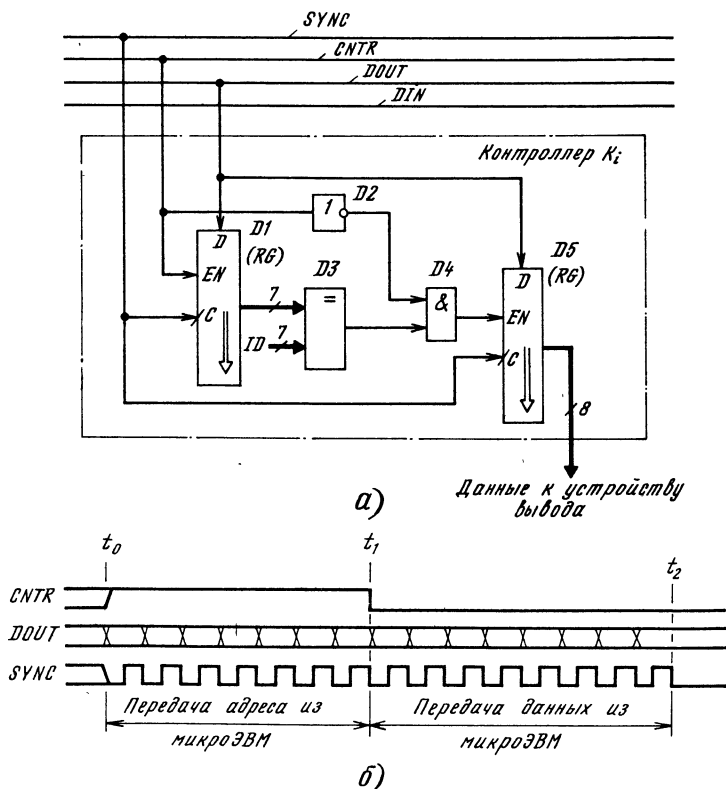


Рис. 1.24. Структура контроллера  $K_i$  (второй вариант) (а) и временные диаграммы его работы (б)

Приведенные в [11] схемы (рис. 1.22, 1.24) можно усовершенствовать, если факт опознавания адреса фиксировать в триггере, освободив регистр адреса для накопления данных. Возможный вариант построения контроллера устройства ввода (аналог схемы, рис. 1.22) показан на рис. 1.25, а. В этой схеме использован сдвиговой регистр D2 с параллельным приемом данных. При  $CNTR=0$  данные с входа IN по положительным фронтам сигнала SYNC вдвигаются в регистр, что позволяет процессору заполнить его адресной и управляющей информацией (см. рис. 1.25, б, период  $t_0-t_1$ ). В момент  $t_1$  триггер D4 выбранного контроллера устанавливается в единицу; в невыбранных контроллерах эти триггеры переводятся в нулевые состояния (или подтверждают их), так как компараторы D3 не зарегистрировали совпадения принятого адреса с собственным (ID). В период  $t_1-t_2$   $CNTR=1$ , что соответствует параллельному приему выбранного мультиплексором байта данных в регистр D2 по положительному фронту сигнала SYNC.

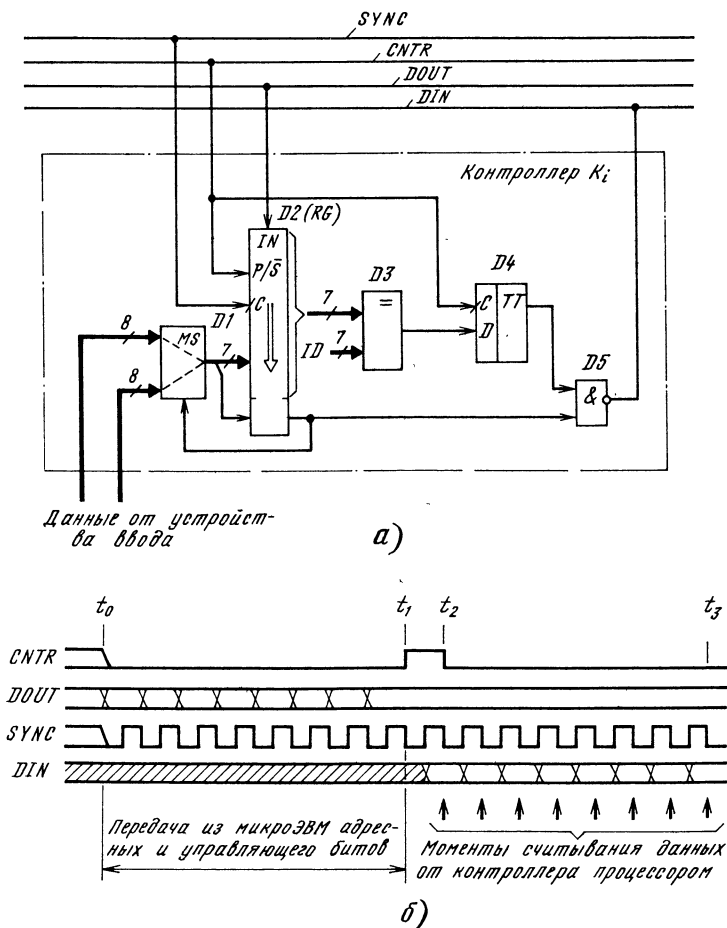


Рис. 1.25. Структура контроллера  $K_i$  (третий вариант) (а) и временные диаграммы его работы (б)

В период  $t_2 - t_3$  данные из регистра D2 выбранного контроллера последовательно передаются через элемент И—НЕ D5 в порт микроЭВМ и считываются процессором в моменты, показанные на рисунке стрелками. Неопределенное значение сигнала DIN соответствует заштрихованной части его временной диаграммы.

Контроллер, показанный на рис. 1.26, а, предназначен для двустороннего обмена данными с внешними устройствами. Ввод данных аналогичен только что рассмотренному (см. рис. 1.25), с тем отличием, что по мере освобождения регистра D2 от старых данных он загружается новыми, поступающими из порта микроЭВМ.

В момент  $t_3$  (см. рис. 1.26, б) процессор считывает последний бит старых данных, в момент  $t_4$  новые данные, последовательно заполнившие регистр D2 выбранного контроллера, «мгновенно» переписываются в параллельный регистр D5 по положительному фронту сигнала CNTR1 в дополнительно введенной выходной управляющей линии общей шины. В момент  $t_5$  процесс ввода—вывода данных завершается.

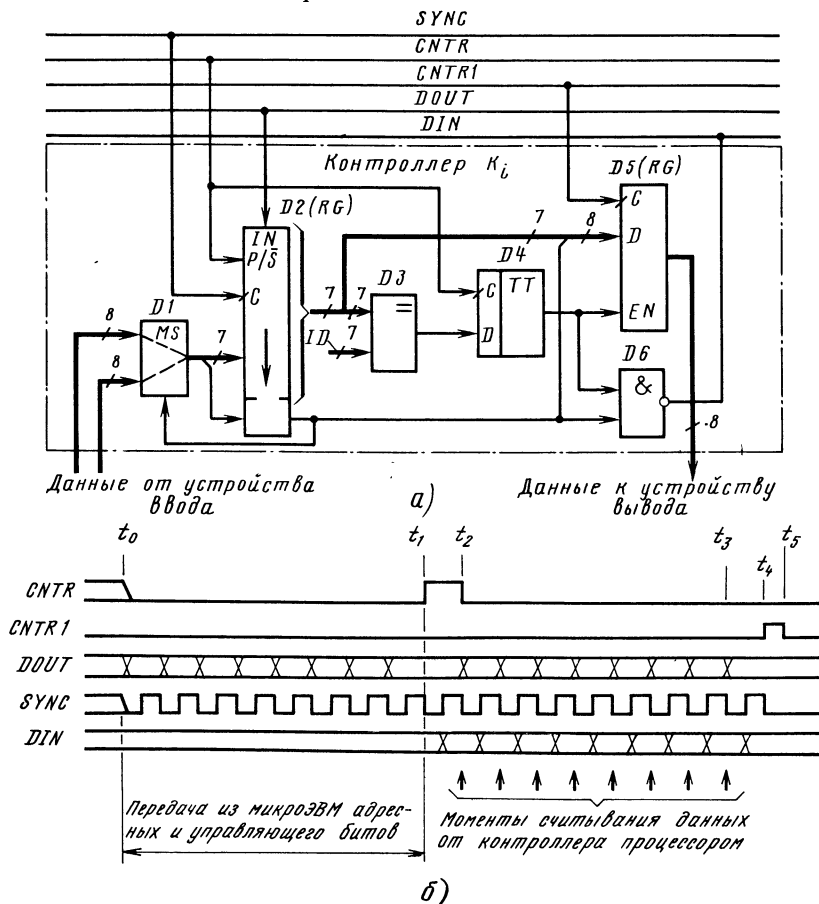


Рис. 1.26. Структура контроллера  $K_i$  (четвертый вариант) (а) и временные диаграммы его работы (б)

Если нет необходимости изменять выходные данные в регистре D5 (нужен только ввод), то процессор либо просто их подтверждает (копия старых данных хранится в ОЗУ микроЭВМ), либо вообще не формирует сигналы CNTR1 и DOUT после момента  $t_2$ , т.е. фактически пользуется временной диаграммой, показанной на рис. 1.25, б, постоянно поддерживая сигнал CNTR1 в состоянии лог. 0.

## 1.12. Порт ввода—вывода с защитой от электрических перегрузок [12]

N—разрядный порт ввода—вывода (рис. 1.27) подключается к магистрали микроЭВМ и содержит следующие программно-доступные элементы: регистр управления (триггеры D1, D101), регистр выходных данных (триггеры D3, D103), линии входных данных (выходы компараторов D4, D104). Этим

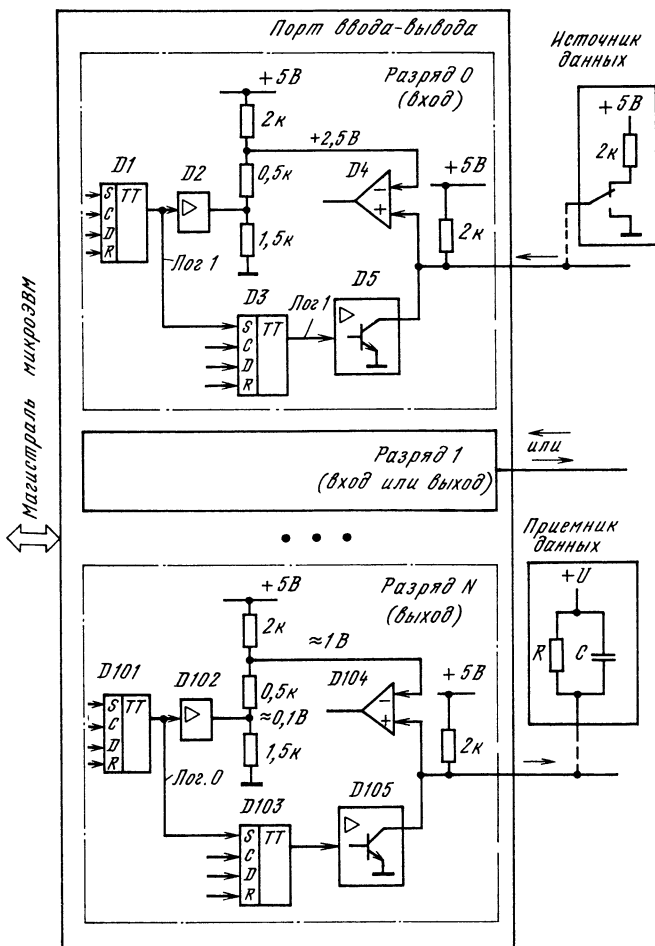


Рис. 1.27. Порт ввода—вывода микроЭВМ

элементам присвоены соответствующие адреса (аппаратура дешифрации адресов не показана), поэтому процессор при работе с портом оперирует ими, как обычными адресами ячеек памяти.

Запись кода в регистр управления позволяет назначить тот или иной разряд порта входным или выходным. В примере, показанном на рисунке, разряд 0—входной, разряд N—выходной, так как триггеры D1 и D101 регистра управления установлены соответственно в состояния лог. 1 и лог. 0. В качестве источника данных для разряда 0 показан ключ, приемником данных из разряда N является некоторое устройство, эквивалентная схема входной цепи которого представлена RC-цепью.

В разряде 0 сигнал лог. 1 с выхода триггера D1 поступает на вход S установки единицы триггера D3 и постоянно поддерживает его в состоянии лог. 1. Поэтому запись лог. 0 в триггер D3 со стороны процессора невозможна (это справедливо для всех разрядов порта, назначенных входными). Выходной транзистор усилителя D5 выключен и не препятствует приему входных данных (от ключа) на вход «+» компаратора D4.

Усилитель D2 с открытым коллектором также выключен, поэтому на входе «—» компаратора присутствует напряжение, близкое 2,5 В, которое задает порог его срабатывания, равноудаленный от обоих входных электрических уровней сигнала, поступающего с ключа (0 и 5 В). В зависимости от состояния ключа на выходе компаратора D4 формируется соответствующий сигнал, который считывается процессором по магистрали при выполнении программы.

В разряде N сигнал лог. 0 с выхода триггера D101 не препятствует установке триггера D103 в то или иное состояние, задаваемое процессором при записи кодов в регистр выходных данных порта. На выходе усилителя D102 с открытым коллектором сформирован сигнал лог. 0—напряжение, близкое 0,1 В, поэтому на входе «—» компаратора D104 постоянно присутствует пороговое напряжение, близкое 1 В. В зависимости от передаваемых данных выходной транзистор усилителя D105 выключен или насыщен, напряжение на его коллекторе в установившемся режиме близко  $+U$  (лог. 1) или не превышает 0,7 В (лог. 0). Если при передаче сигнала лог. 0 это напряжение превысило установленный порог, то это означает, что выходной транзистор перегружен—через него протекает слишком большой ток.

Чтобы защитить порт от таких перегрузок, процессор после каждой смены данных в выходном регистре (или, помимо этого, например, через каждые 0,1 с) проверяет состояния соответствующих компараторов и в необходимых случаях программно выключает перегруженный выходной усилитель. При этом оператору выдается сообщение о возникшей ситуации либо выполняются иные действия, предусмотренные программой.

Так как непосредственно после включения выходного транзистора усилителя D105 ток через него может быть достаточно большим из-за накопления заряда на емкости нагрузки, то процессор, следуя программе, опрашивает состояние компаратора D104 с некоторой задержкой, достаточной для окончания переходных процессов во внешней цепи.

### **1.13. Поиск источника прерывания преобразованием суммарного потенциального запросного сигнала в последовательность импульсных сигналов [13]**

Если микроЭВМ имеет единственный вход прерывания, а источников запросных сигналов несколько, причем все эти сигналы логически суммируются, то микроЭВМ, получив результирующий запросный сигнал, должна определить, какие источники требуют обслуживания. Вход прерывания микроЭВМ обычно выполняется динамическим, т. е. микроЭВМ реагирует не на логический уровень сигнала, а на его изменение, например на переход из состояния лог. 1 в состояние лог. 0.

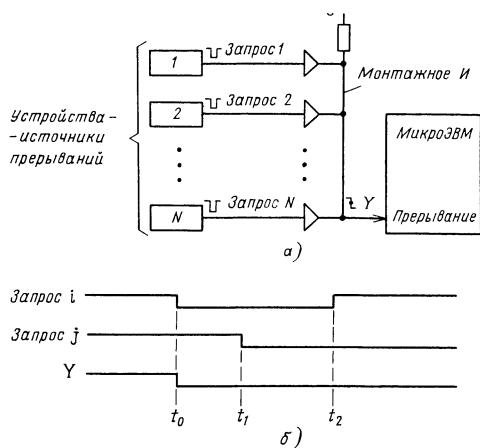


Рис. 1.28. Схема передачи запросных сигналов на вход прерывания микроЭВМ (а) и временная диаграмма (б), соответствующая наложению запросных сигналов. Воспринимается только запрос  $i$ , запрос  $j$  и следующие за ним теряются

Проблема состоит в следующем. Пусть имеется ряд источников прерываний (рис. 1.28), которые посылают в микроЭВМ запросные сигналы (лог. 0). Эти сигналы суммируются в линии типа Открытый коллектор. Если запросные сигналы не накладываются друг на друга по времени, то микроЭВМ реагирует на каждый из них, точнее, на каждый отрицательный фронт (лог. 1  $\rightarrow$  лог. 0) результирующего сигнала. Если же запросные сигналы от разных источников накладываются, то воспринимается только отрицательный фронт первого по времени запросного сигнала, другие запросы не обслуживаются, как показано на рис. 1.28, б.

В данном примере в моменты  $t_0$  и  $t_1$  формируются запросы от устройств с номерами  $i$  и  $j$ . МикроЭВМ воспринимает отрицательный фронт результирующего сигнала  $Y$ , т. е. реагирует на запрос от устройства с номером  $i$ . В момент  $t_2$  обработанный запрос  $i$  снимается, микроЭВМ готова принять и обработать запрос  $j$ , однако сигнал  $Y$  остается в состоянии лог. 0, так что информация о запросе  $j$  теряется. Дальнейшая работа системы оказывается невозможной из-за того, что устройства ждут обслуживания, поддерживая в состоянии лог. 0 свои запросные сигналы, а микроЭВМ не знает об этом, так как на ее входе прерывания нет отрицательных фронтов сигнала  $Y$ .

Чтобы избежать подобных ситуаций, можно преобразовать суммарный запросный сигнал  $Y$  в последовательность импульсных сигналов. При этом на входе прерывания микроЭВМ формируется нужное число отрицательных фронтов сигнала по числу стоящих на очереди запросов. В качестве примера рассмотрим схему разделения запросов, приведенную на рис. 1.29 [13], которая является составной частью системы управления электродвигателем.

Система содержит электродвигатель 1 переменного тока, симметричный тиристор (симистор) 2, датчик 3 скорости вращения вала электродвигателя 1, детектор 4 нулевого напряжения в сети переменного тока 220 В, 50 Гц, триггеры 5, 6, элементы И—НЕ 7, 8, усилитель 9, микроЭВМ 10, а также дополнительный элемент И 11, который пока не принимаем во внимание.

Задача заключается в поддержании постоянной (заданной) скорости вращения вала электродвигателя 1. На выходе датчика 3 формируются короткие импульсы В, частота следования которых пропорциональна скорости вращения вала. МикроЭВМ с помощью внутреннего таймера (на рисунке не показан)

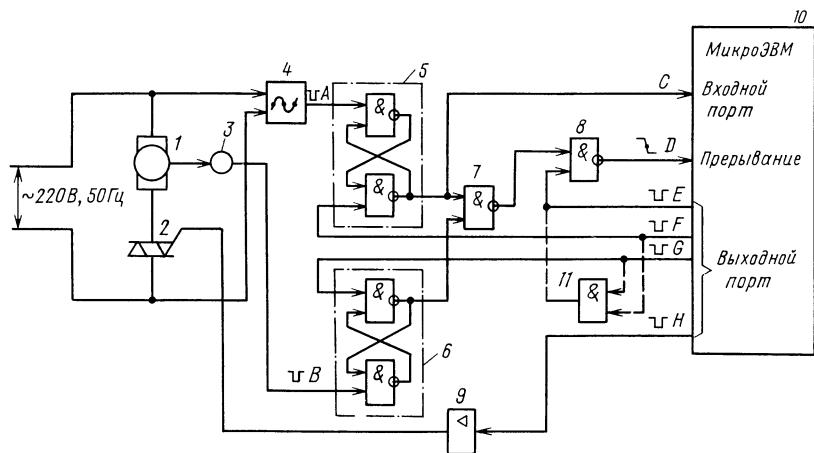


Рис. 1.29. Система управления электродвигателем.  
Запросные сигналы суммируются элементом 7, но разделяются на отдельные импульсы элементом 8

измеряет текущий период следования импульсов В, по результатам измерения вычисляет скорость вращения вала электродвигателя, сравнивает ее с заданной и в необходимых случаях уменьшает или увеличивает среднюю мощность, развиваемую электродвигателем.

Управление мощностью осуществляется в каждом полупериоде переменного напряжения (220 В, 50 Гц) выбором момента (фазового угла) включения симистора; его выключение происходит автоматически в конце каждого полупериода, когда ток через симистор уменьшается до некоторого порогового значения. Для включения симистора микроЭВМ в нужный момент времени (в каждом полупериоде переменного напряжения) выдает на вход усилителя 9 кратковременный импульсный сигнал Н (схемотехнические подробности построения цепи управления симистором опущены для упрощения рисунка). Чтобы правильно задавать величину фазового угла включения симистора, микроЭВМ должна иметь информацию о начальных точках отсчета времени, т. е. о моментах прохождения синусоиды питающего переменного напряжения через 0. Эта информация в виде коротких импульсов А поступает с детектора 4.

Таким образом, в данном примере микроЭВМ воспринимает информацию от двух независимых источников прерываний — детектора 4 нулевого напряжения, и датчика 3 скорости вращения вала. Импульсные сигналы от этих источников запоминаются в триггерах 5 и 6 и хранятся до тех пор, пока микроЭВМ не воспримет соответствующий запрос, после чего триггеры устанавливаются в исходное состояние сигналами F и G.

При работе системы триггеры 5 и 6 могут устанавливаться сигналами А и В в произвольной последовательности, в частности запросные сигналы с выходов этих триггеров могут перекрываться во времени, как было показано на рис. 1.28, б. Однако благодаря схеме разделения запросов (на основе элемента И — НЕ 8) суммарный сигнал D на входе прерывания микроЭВМ не фиксируется в состоянии лог. 0, а дробится сигналом E на два импульса, так что оба запроса, перекрывающихся во времени, обрабатываются последовательно. Рассмотрим работу системы подробнее.

В установившемся режиме работы системы существуют интервалы времени, когда запросы от датчика 3, детектора 4 и внутреннего таймера отсутствуют. В эти интервалы времени микроЭВМ выполняет фоновую программу, возможно, не связанную с управлением электродвигателем 1. На входах

элемента И—НЕ 7 присутствуют сигналы лог. 1, соответствующие отсутствию запросов от обоих источников прерываний, на верхнем входе элемента И—НЕ 8 сформирован сигнал лог. 0,  $D=1$ ,  $E=F=G=H=1$ ,  $C=1$ .

При поступлении сигнала А или В (или обоих сигналов одновременно) на одном или двух входах элемента И—НЕ 7 формируются сигналы лог. 0, на вход прерывания микроЭВМ поступает отрицательный фронт сигнала D. МикроЭВМ временно откладывает текущее (фоновое) задание, запоминает параметры возврата к нему и приступает к выполнению прерывающей программы.

Прерывающая программа опрашивает входной порт. Если  $C=0$ , то выполняется первая ветвь прерывающей программы, соответствующая появлению очередной начальной точки отсчета для задания нужного фазового угла включения симистора. Если  $C=1$ , то управление передается второй ветви, соответствующей оценке текущей скорости вращения вала электродвигателя. Первая ветвь завершается командами одновременной выдачи кратковременных сигналов  $E=0$  и  $F=0$  в выходной порт микроЭВМ для установки триггера 5 в исходное состояние и безусловного перевода D в состояние лог. 1 даже при наличии конкурирующего запроса, перекрывающегося во времени с только что обработанным. Сигнал  $A=0$  к моменту установки триггера 5 в исходное состояние заведомо кончился, так что триггер 5 переходит в состояние  $C=1$ . Аналогично вторая ветвь завершается командами одновременной выдачи сигналов  $E=0$  и  $G=0$ , триггер 6 возвращается в исходное состояние, элемент И—НЕ 8 временно закрывается по нижнему входу.

Далее в обеих ветвях прерывающей программы размещены команды выхода из прерывания. При отсутствии конкурирующего запроса окончание сигнала  $E=0$  не сопровождается формированием отрицательного фронта сигнала D, поэтому осуществляется возврат к выполнению фоновой программы. Однако если конкурирующий запрос имеется, то в момент снятия сигнала  $E=0$  на верхнем входе элемента И—НЕ 8 присутствует лог. 1 (обобщенный запросный сигнал), поэтому сигнал D переходит из 1 в 0, микроЭВМ вновь прерывается, анализирует состояние сигнала C и описанные ранее процессы повторяются.

Число разрядов выходного порта микроЭВМ можно уменьшить, если нижний вход элемента И—НЕ 8 отключить от микроЭВМ и соединить с выходом дополнительного элемента И11, на входы которого поданы сигналы F и G (см. штриховые линии на рис. 1.29). В этом случае сигнал лог. 0 поступает на нижний вход элемента И—НЕ 8 всякий раз, когда осуществляется установка в исходное состояние триггера 5 или 6, что не противоречит приведенному ранее описанию работы системы.

### 1.14. МикроЭВМ с распределенным арбитром запросов прерываний [14]

В микроЭВМ (рис. 1.30) ЦП взаимодействует с контроллерами K1—K4 с использованием «обычной» магистрали, содержащей шины адреса А, данных D и управления С (ОЗУ, ПЗУ и иные блоки, не существенные для описания идеи построения распределенного арбитра, не показаны). Контроллеры K1—K4 устанавливаются в соединители, размещенные на основной плате. Каждый контроллер обслуживает одно или несколько периферийных устройств и имеет ряд программно-доступных регистров, что позволяет процессору записывать и считывать из них данные (эти устройства и регистры также не показаны).

Чтобы избавить ЦП от необходимости периодического опроса регистров контроллеров с целью выявления запросов на обслуживание, обычно используют режим прерывания. При необходимости «привлечь к себе внимание» процессора, который исходно работает на фоновой программе, контроллер формирует сигнал запроса прерывания. Получив этот сигнал, ЦП временно

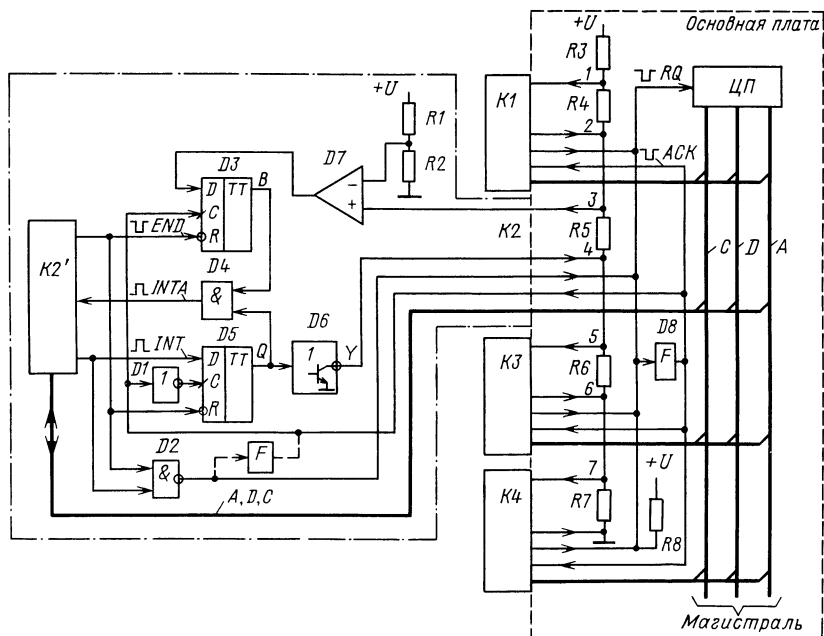


Рис. 1.30. Схема сопряжения контроллеров K1—K4 с процессором. Раскрыта схема контроллера K2; K2'—«собственно контроллер», остальные элементы составляют часть распределенного арбитра

прекращает выполнение фоновой программы, запоминает в ОЗУ параметры возврата к ней и переходит к прерывающей программе.

В данном решении предполагается, что прерывающая программа начинается с команды «Кто Вы?», которую ЦП адресует одновременно всем контроллерам. (Это обычная команда чтения по некоторому адресу, который распознаётся дешифраторами, размещенными во всех контроллерах.) Если только один контроллер просил обслуживания, то он, получив эту команду, устанавливает в шине данных свой номер, который принимается процессором; остальные контроллеры пассивны и не реагируют на приглашение к диалогу.

Если несколько контроллеров одновременно или почти одновременно установили сигналы запросов прерывания в общей линии, то в результате срабатывания распределенного приоритетного арбитра выбирается только один контроллер, который вступает в диалог с процессором.

По окончании работы с выбранным контроллером (работа состоит из ряда обращений к программно-доступным элементам контроллера) ЦП формирует команду «Конец обслуживания», которая так же, как и команда «Кто Вы?», адресуется одновременно всем контроллерам. Если все запросы удовлетворены, то ЦП выходит из прерывающей программы и возвращается к фоновому заданию, в противном случае он начинает обмен данными с очередным контроллером, который в новом цикле работы арбитра получил право обслуживания. Рассмотрим работу микроЭВМ подробнее.

В исходном состоянии ЦП работает по фоновой программе, контроллеры K1—K4 не требуют обслуживания. В каждом из них триггеры D3 и D5 установлены в 0 отрицательным импульсом, ранее поступившим на их R-входы, END=1 (напряжение высокого уровня), INT=0, INTA=0, выходной

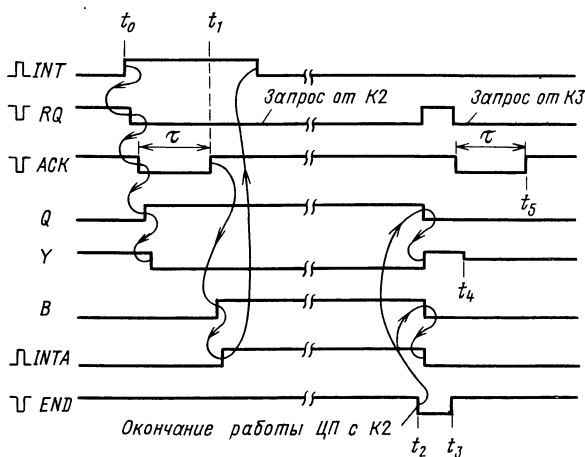


Рис. 1.31. Временные диаграммы работы распределенного арбитра при одновременном формировании запросов контроллерами K2 и K3 (K1 и K4 остаются пассивными)

транзистор инвертора D6 выключен, элемент И—НЕ D2 с открытым коллектором выключен, в общей линии запроса прерывания поддерживается сигнал  $RQ=1$  (запроса нет), формирователь импульса D8 выключен, на его выходе присутствует сигнал  $ACK=1$ .

Компаратор D7 с высоким входным сопротивлением сравнивает опорное напряжение, снимаемое со средней точки делителя напряжения R1—R2, с напряжением, поступающим с соответствующей точки делителя напряжения R3—R7. Опорные напряжения одинаковы для всех контроллеров и выбираются несколько меньшими падения напряжения на резисторе R7 в исходном состоянии, когда делитель напряжения R3—R7 не шунтирован сигналами лог. 0 с выходов инверторов D6 разных контроллеров. Поэтому при отсутствии запросов со стороны контроллеров в каждом из них напряжение на входе «+» компаратора D7 превышает напряжение на входе «-», следовательно, на выходах компараторов D7 всех контроллеров сформированы сигналы лог. 1. Формирователь F импульса, размещенный в плате контроллера, пока не принимает во внимание.

Предположим, что в момент  $t_0$  (рис. 1.31) с одного из выходов «собственно контроллера» K2' (т. е. контроллера K2, из которого для наглядности «выделены» элементы, принадлежащие распределенному арбитра) поступает сигнал  $INT=1$  — запрос на обслуживание. Этот сигнал проходит через элемент И—НЕ D2 и в виде сигнала  $RQ=0$  поступает в общую линию прерывания. Отрицательный фронт сигнала  $RQ=0$  воспринимается процессором как сигнал прерывания, поэтому он прекращает выполнение фоновой программы, запоминает параметры возврата к ней и переходит к выполнению первой команды прерывающей программы (эти процессы на рис. 1.31 не отражены; сейчас мы рассматриваем только самый начальный этап процесса обслуживания запроса).

По отрицательному фронту сигнала  $RQ=0$  срабатывает формирователь импульса D8. На его выходе вырабатывается отрицательный импульс  $ACK=0$  длительностью  $\tau$ , достаточной для окончания переходных процессов в распределенном арбитра. Пока развивались описанные процессы, могли появиться новые запросы от других контроллеров. Для определенности предположим, что одновременно с запросом от K2 или чуть позже сформирован запрос от K3 (на временных диаграммах этот факт не отражен, запрос от K3

«проявляется» только в момент  $t_3$ ); K1 и K4 пока остаются в пассивном состоянии.

Отрицательный фронт сигнала  $ACK=0$  инвертируется элементами D1 во всех контроллерах, в результате их триггеры D5 принимают сигналы с D-входов. В данном примере триггеры D5 контроллеров K2 и K3 устанавливаются в единицу. Эти же триггеры контроллеров K1 и K4 остаются в нулевом состоянии. Сигнал  $Q=1$  в контроллере K2 проходит через инвертор D6, выходной транзистор этого инвертора насыщается, напряжение в точке 4 снижается примерно до 0,2 В, поэтому резисторы R6 и R7 делителя напряжения R3—R7 «обесточиваются» (возможные контроллеры — конкуренты с низкими приоритетами исключаются из рассмотрения).

К моменту окончания отрицательного импульса  $ACK=0$  выходы компараторов D7 контроллеров K1—K4 находятся в следующих состояниях: 1, 1, 0, 0. Поэтому в момент  $t_1$  окончания этого импульса триггеры D3 контроллеров K1 и K2 устанавливаются в единицу, а в контроллерах K3 и K4 эти триггеры остаются в нулевом состоянии. В контроллере K2 срабатывает элемент И D4, на его выходе формируется сигнал  $INTA=1$  подтверждения перехода ЦП к обслуживанию этого контроллера, сигнал  $INT=1$  снимается, так как он сыграл свою роль. В контроллере K1 элемент И D4 закрыт по нижнему входу (не было запроса от этого контроллера), в K3 — по верхнему (запрос был, но с компаратора D7 поступил сигнал лог. 0), в K4 — по обоим входам.

Получив сигнал  $INTA=1$ , контроллер K2 «знает», что только он имеет право отреагировать на команду «Кто Вы?» выдачей в шину D собственного номера. Процессор принимает этот номер и использует его для выбора соответствующей программы обслуживания. По окончании сеанса связи с K2 (момент  $t_2$ ) ЦП формирует команду «Конец обслуживания» (обычная команда чтения или записи по некоторому адресу, который распознается одновременно всеми контроллерами), в результате во всех контроллерах формируется отрицательный импульс  $END=0$ , триггеры D3 и D5 всех контроллеров устанавливаются в 0 (или подтверждают старые нулевые состояния), элементы И—НЕ D2 на время действия этого импульса закрываются по верхнему входу, так что сигнал RQ временно переходит в состояние лог. 1.

Делитель напряжения R3—R7 возвращается в «нормальное» состояние, так как нет шунтирующего действия сигналов с выходов инверторов D6. В момент  $t_3$  заканчивается импульс  $END=0$ . Не обслуженный запрос от K3' вызывает срабатывание элемента И—НЕ D2 в K3, поэтому формируется новый сигнал  $RQ=0$  (начинается новый цикл арбитража). Процессор в это время завершает выполнение команды «Конец обслуживания» и проверяет состояние внутреннего триггера прерывания.

В данном примере этот триггер установлен в состояние, соответствующее наличию запроса прерывания, поэтому ЦП не возвращается к фоновой программе, а приступает к выдаче команды «Кто Вы?», и т. д. (параметры возврата к фоновой программе уже сохранены в памяти при первом вхождении в прерывающую программу). По отрицательному фронту сигнала  $ACK=0$  в контроллере K3 устанавливается в единицу триггер D5, поэтому в точке 6 формируется напряжение низкого уровня, напряжение в точке 4 несколько уменьшается (момент  $t_4$ ). В момент  $t_5$  заканчивается импульс  $ACK=0$ , контроллер K3 готов к восприятию команды «Кто Вы?», которая в скором времени поступает из ЦП, и т. д.

Формирователь импульса D8 и общую управляющую линию для передачи сигнала  $ACK$  можно исключить, если в каждый контроллер ввести «собственный» формирователь F, как показано на рисунке штриховыми линиями.

Приоритеты контроллеров убывают в направлении сверху вниз. В отличие от традиционных схем с последовательно убывающими приоритетами (дейзипечь), в предлагаемой микроЭВМ не обязательно устанавливать платы контроллеров «вплотную» друг к другу, чтобы не нарушить целостность последовательной цепи опроса, так как ее просто нет.

## 1.15. Уменьшение аппаратуры обслуживания запросов прерываний [15]

В традиционных схемах обслуживания запросов прерываний используют индивидуальные или общие линии связи между контроллерами  $K_i$  и ЦП (рис. 1.32). В первом примере (рис. 1.32, а) запросы прерываний  $RQ1$  и  $RQ2$  от контроллеров  $K1$  и  $K2$  (для упрощения рисунка показаны только два контроллера) поступают в ЦП, который анализирует их, выбирает наиболее приоритетный и посылает соответствующему контроллеру ответный сигнал  $ACK1$  или  $ACK2$ , подтверждающий, что запрос принят к обслуживанию.

Во втором примере (рис. 1.32, б) запросы от контроллеров суммируются по схеме Монтажное ИЛИ в общей линии, поэтому процессор, получив суммарный сигнал, не может сразу определить, сколько имеется источников запросов и где они находятся. Для выявления источника запроса процессор формирует ответный сигнал  $ACK1/2$ , который распространяется по последовательной цепи из контроллеров до первого из них, требующего обслуживания. Этот контроллер передает в процессор вектор прерывания  $V1/2$ , используя шину данных; процессор по вектору однозначно идентифицирует контроллер—источник прерывания.

Общий недостаток обеих схем—сложность: первая содержит радиальные линии связи процессора с контроллерами, вторая—последовательную цепь распространения ответных сигналов. Кроме того, во второй схеме приоритеты контроллеров убывают по мере удаления от процессора, что не позволяет оперативно их изменять.

Предлагаемое решение (рис. 1.33) позволяет уменьшить число линий магистрали, выделенных для установления связи контроллера с процессором, с сохранением возможности программного управления приоритетами запросов прерываний. Фактически в шине управления магистрали оставлена только одна линия, «обязанная своим существованием» исключительно процедуре прерывания,—по ней из контроллеров  $K1—K_N$  в процессор передается суммированный по ИЛИ сигнал  $RQ$  запроса, представленный напряжением низкого уровня (линия типа «открытый коллектор», нагрузочные резисторы не показаны).

Две другие линии шины управления, в явном виде отображенные на рисунке, передают «обычные» сигналы:  $WR$ —запись,  $RD$ —чтение. Иными словами, эти линии присутствуют в микроЭВМ независимо от того, предусмотрена ли в ней работа по прерыванию. Преимущества предлагаемого решения, однако, достигаются благодаря тому, что, как предполагается, все контроллеры  $K1—K_N$  способны осуществлять прямой доступ в память, т. е. обладают достаточным внутренним «интеллектом».

В адресном пространстве микроЭВМ (рис. 1.34) размещены адреса ОЗУ и контроллеров  $K1—K_N$  (иные устройства не представляют для нас интереса).

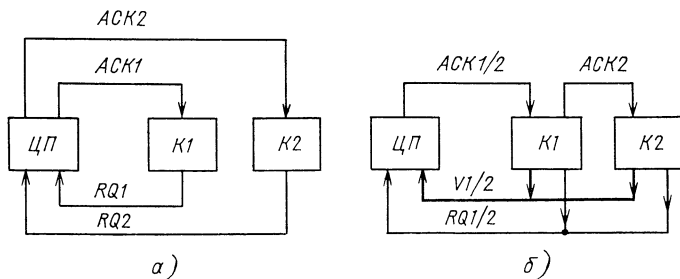


Рис. 1.32. Типовые схемы передачи запросных и ответных сигналов прерываний между контроллерами  $K1, K2$  и центральным процессором:

а—с индивидуальными линиями запросных ( $RQ1, RQ2$ ) и ответных ( $ACK1, ACK2$ ) сигналов; б—с общими линиями запросных сигналов ( $RQ1/2$ ), векторов ( $V1/2$ ) и деши-цепью распространения ответных сигналов ( $ACK1/2, ACK2$ )

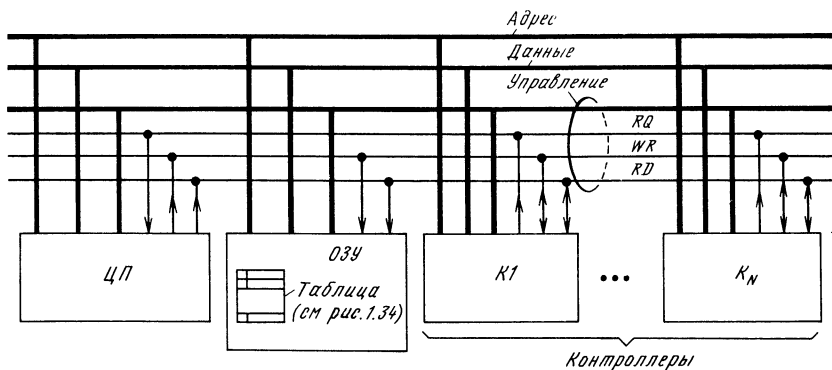


Рис. 1.33. Структура микроЭВМ

В ОЗУ выделена область для хранения таблицы состояний контроллеров. Каждому контроллеру соответствует один байт (строка таблицы); в его старшем разряде хранится признак F, единичное значение которого отражает наличие запроса на обслуживание данного контроллера центральным процессором. Оставшиеся семь разрядов могут содержать дополнительную информацию, например, о причине запроса.

Контроллер  $K_i$  (рис. 1.35) предназначен для сопряжения микроЭВМ с некоторым внешним устройством (не показано) и построен на основе периферийного процессора, способного (помимо прочего) записывать данные в соответствующую строку таблицы состояний контроллеров (см. рис. 1.34).

В исходном состоянии ЦП работает по некоторой фоновой программе, запросов прерываний от контроллеров  $K_1 \dots K_N$  нет, все признаки F в таблице состояний установлены в 0. При необходимости получения обслуживания со стороны центрального процессора микроЭВМ периферийный процессор контроллера  $K_i$  формирует импульсный сигнал  $IRQ=1$ , устанавливая триггер D6 в состояние лог. 1. Затем периферийный процессор захватывает магистраль микроЭВМ (с разрешения арбитра, не показанного на рисунках) и записывает в отведенную ему в ОЗУ ячейку таблицы состояний некоторый код, содержащий признак F=1.

Адрес этой ячейки, код данных и управляющий сигнал записи передаются из периферийного процессора в магистраль микроЭВМ через элементы D1, D3 и D12, работа которых разрешена сигналом  $EN=1$ . Дешифратор D7 реагирует на адрес табличной ячейки, выделенной данному контроллеру, формированием сигнала  $Y=1$ . Поэтому одновременно с записью запросного кода в таблицу срабатывает элемент И D8,

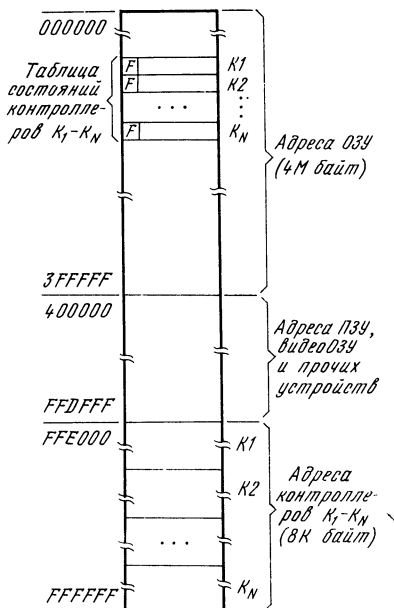


Рис. 1.34 Распределение адресного пространства микроЭВМ (рис. 1.33)

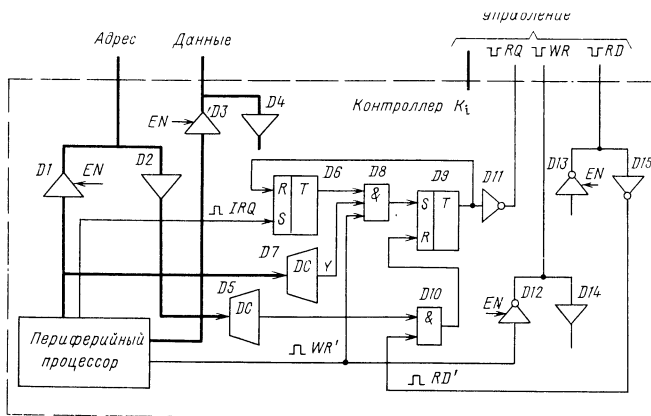


Рис. 1.35. Фрагмент схемы контроллера (рис. 1.33)

триггер D9 устанавливается в единицу, триггер D6 возвращается в исходное состояние (лог. 0), элемент D11 с открытым коллектором формирует на выходе сигнал лог. 0 — признак запроса прерывания.

Центральный процессор микроЭВМ, получив сигнал  $RQ=0$ , временно прекращает выполнение фоновго задания, запоминает в стеке параметры возврата к нему и переходит к прерывающей программе. Эта программа предусматривает поиск признаков F в таблице состояний контроллеров. Отметим, что к моменту опроса и в его процессе в таблице могут появиться изменения, связанные с формированием новых запросов, что, однако, не приводит к каким-либо нежелательным последствиям. Обнаружив первый встретившийся признак  $F=1$ , ЦП приступает к обслуживанию соответствующего контроллера.

В процессе обслуживания ЦП, в частности, считывает данные из программно-доступных элементов контроллера, обращаясь в соответствующую часть области FFE000—FFFFFF адресного пространства. Первое такое обращение вызывает формирование сигнала лог. 1 на выходе дешифратора D5, при этом срабатывает элемент И D10, триггер D9 устанавливается в 0. Сигнал  $RQ=0$  в линии магистрали снимается, если нет запросов прерываний от других контроллеров. Если запросы есть, то сигнал  $RQ=0$  остается неизменным, ЦП либо вновь прерывается, либо откладывает реакцию на запросы до окончания работы по исходной прерывающей программе — все зависит от структуры алгоритма обработки прерываний, в котором предусмотрено их маскирование в нужных ситуациях. По окончании обслуживания контроллера (контроллеров) ЦП микроЭВМ возвращается к прерванному заданию.

Таким образом, запрос прерывания передается из контроллера в ЦП обычным путем — по общей линии, а ответный сигнал снятия запроса формируется без использования дополнительных линий магистрали. Приоритеты контроллеров задаются программно — очередностью опроса строк таблицы состояний контроллеров.

## 1.16. Обработка прерываний в многопроцессорной системе с индивидуальными блоками кэш-памяти [16]

В многопроцессорной системе (рис. 1.36) запросы прерываний от контроллеров внешних устройств разделены на три уровня (число уровней можно увеличить). Запросы уровня I поступают от контроллеров накопителей на

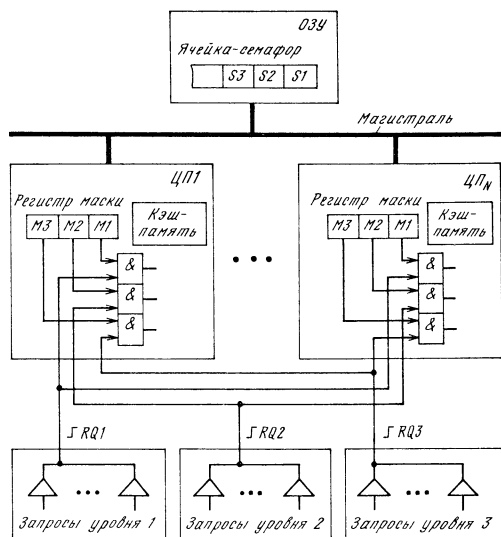


Рис. 1.36. Фрагмент многопроцессорной системы

магнитных дисках и лентах, уровень 2—от клавиатур, пультов управления, уровень 3—от устройств вывода данных (принтеров и т. п.). Сигналы RQ1—RQ3 отображают обобщенные запросы соответствующих уровней 1—3. Например, если есть хотя бы один запрос уровня 1, то RQ1=1 и т. д.

С точки зрения пользователя не имеет значения, какой именно процессор ЦП1—ЦП<sub>n</sub> воспримет запрос прерывания—важно, чтобы он был обработан, т. е. устройство было обслужено процессором.

Однако если в системе имеются процессоры, не занятые выполнением заданий в данном интервале времени (например, только что завершившие их), то им и следовало бы адресовать поток запросов прерываний, чтобы не отвлекать от работы занятые процессоры. Прежде чем показать, как эта задача решается, рассмотрим некоторые существенные элементы системы.

В общем для всех процессоров ОЗУ одна ячейка использована в качестве программного семафора, ее разряды S1—S3 отражают следующие ситуации. Если S<sub>i</sub>=1, то для обработки запросов i-го уровня выделен некоторый процессор; если S<sub>i</sub>=0, то такой процессор еще предстоит выделить. В исходном состоянии S1=S2=S3=0.

Каждый процессор соединен с собственным ОЗУ (на рисунке не показано) и имеет быстродействующую кэш-память, в которой автоматически накапливаются копии команд и данных, считываемых по магистрали из общего ОЗУ. Благодаря этому процессор может в быстром темпе многократно считывать накопленные в кэш-памяти копии, не обращаясь к «медленному» общему ОЗУ.

Обобщенные сигналы RQ1—RQ3 запросов прерываний поступают одновременно во все процессоры. Каждый процессор содержит программно-доступный регистр маски, в котором разряды M1—M3 соответствуют запросам уровней 1—3. При M1=0 процессор не воспринимает запросы уровня 1, при M1=1—реагирует на них. То же относится к разрядам M2 и M3. В исходном состоянии во всех процессорах M1=M2=M3=1. Контроллеры внешних устройств (на рисунке не показаны), от которых поступают запросы прерываний, подключены к магистрали; каждый из них может обслуживаться любым процессором.

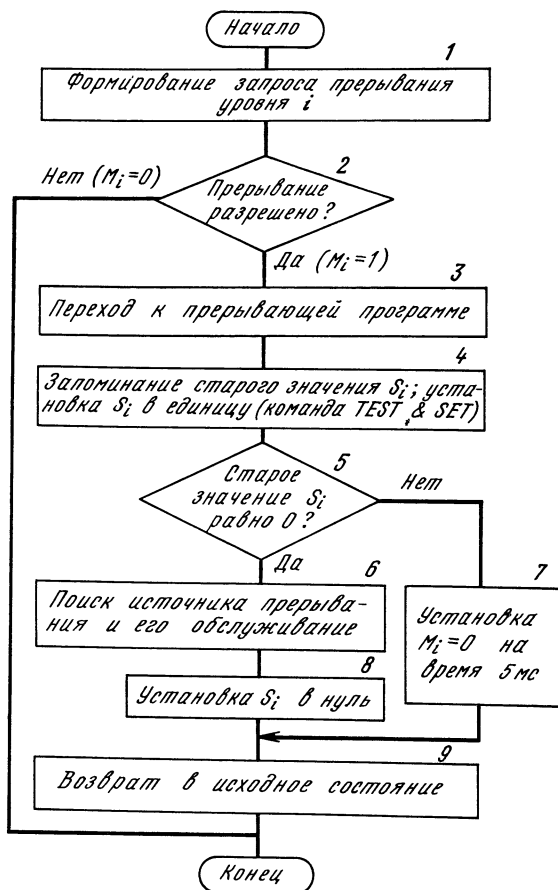


Рис. 1.37. Блок-схема алгоритма обработки запроса прерывания уровня  $i$

Предположим, что в некоторый момент на вход системы поступает запрос прерывания уровня  $i$  (рис. 1.37, блок 1). Сигнал  $RQ_i=1$  передается одновременно во все процессоры; каждый из них в зависимости от состояния разряда  $M_i$  своего регистра маски либо не воспринимает этот сигнал, либо запоминает параметры возврата к текущему состоянию и переходит к прерывающей программе (блоки 2, 3).

В соответствии с первой командой (TEST&SET, блок 4) прерывающей программы каждый процессор (из числа воспринявших сигнал прерывания) проверяет разряд  $S_i$  ячейки-семафора и затем безусловно устанавливает его в 1. Результат проверки хранится в регистре состояния процессора и в дальнейшем используется в качестве условия ветвления программы (блок 5). При выполнении команды TEST&SET процессор постоянно занимает магистраль, чтобы никто из конкурентов не мог «увидеть» в разряде  $S_i$  ячейки-семафора лог. 0—признак допуска процессора к обслуживанию устройства (если этот признак есть). Таким образом, процессор, который раньше других выполнил команду TEST&SET, считывает нулевое значение  $S_i$ , а остальные—единичные.

Возвращаясь к общему описанию системы, напомним, что в установившемся режиме в ней имеются как занятые, так и свободные процессоры. Чтобы создать последним льготные условия в конкурентной борьбе за право обслуживания источника прерывания, используется следующее правило. При переходе из занятого состояния в свободное процессор помещает в свою кэш-память команду TEST&SET, которую предстоит выполнить, когда начнется конкурентная борьба (см. блок 4). Точнее, в кэш-память записываются три команды, ориентированные на проверку и установку в 1 соответствующих разрядов S1, S2 и S3, но мы сейчас рассматриваем только один, i-й уровень запросов и соответствующий ему разряд  $S_i$  ячейки-семафора.

Загрузка команды TEST&SET в кэш-память автоматически сопровождает «холостую» пересылку содержимого соответствующей ячейки общего ОЗУ в один из регистров общего назначения процессора перед его переходом из занятого в свободное состояние (такая пересылка заложена в программе, вызываемой в связи с освобождением процессора). Занятые процессоры, вероятнее всего, не содержат в кэш-памяти интересующую нас команду проверки и закрытия семафора, так как в процессе выполнения текущего задания через кэш-память прошел поток новой информации (команд и данных), который, возможно, уничтожил эту команду. Таким образом, если в конкурентную борьбу включились занятый и свободный процессоры, то последний, вероятнее всего, быстрее приступит к выполнению команды TEST&SET и, следовательно, возьмет на себя инициативу поиска и обслуживания источника прерывания (блок 6).

Если в процессе обслуживания источника прерывания после гашения его запроса обнаруживается, что есть другие запросы уровня  $i$ , то данный процессор не оставляет их без внимания и безусловно обрабатывает, сохраняя признак  $S_i$  равным 1, чтобы исключить «помощь» других процессоров. Он переходит к открытию семафора (блок 8) только после того, как полностью ликвидирована очередь из запросов прерываний данного уровня.

После этого процессор возвращается в исходное состояние (блок 9). Примечательно, что в данный момент в его кэш-памяти находится информация, связанная с недавней обработкой запроса прерывания уровня  $i$ , так что «ему бы — карты в руки» при обработке последующих запросов этого уровня, ведь его инициатива должна быть наказуема!

Руководствуясь подобными аргументами, процессоры, которые приняли участие в конкурентной борьбе, но проиграли ее, решают в ближайшие 5 мс вообще в ней не участвовать (блок 7)—они временно переводят разряды  $M_i$  своих регистров маски в 0 и возвращаются в исходные состояния (блок 9).

Благодаря использованию данного алгоритма занятые процессоры по возможности не отвлекаются от выполняемых заданий. Прерывания обрабатываются свободными процессорами, а если их нет, то из участвующих в соревновании занятых преимущества имеют те, у которых в кэш-памяти полностью или частично сохранилась информация, связанная с обслуживанием запросов.

## **1.17. Упрощение аппаратуры слежения за временем выполнения заданий однопоточными устройствами [17]**

В микроЭВМ (рис. 1.38) контроллер накопителей на магнитных дисках управляет вводом и выводом данных. В частности, он следит за тем, чтобы накопители во время «отчитывались» о завершении ранее полученных заданий. Например, следуя такому заданию, накопитель должен за приемлемое время найти на магнитном диске дорожку и сектор с заданными номерами. Если он не смог выполнить это задание из-за искажений информации на диске или по каким-либо иным причинам, то контроллер, не дожидаясь ответа в течение допустимого интервала времени, отменяет задание и вновь повторяет его выдачу. После нескольких неудачных попыток поиска нужной дорожки и сектора контроллер оповещает об этом операционную систему, которая решает, что делать дальше.

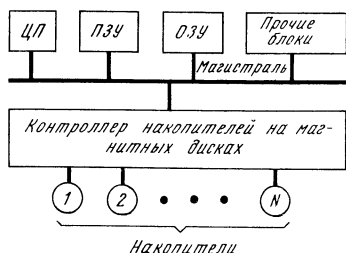


Рис. 1.38. Структура микроЭВМ

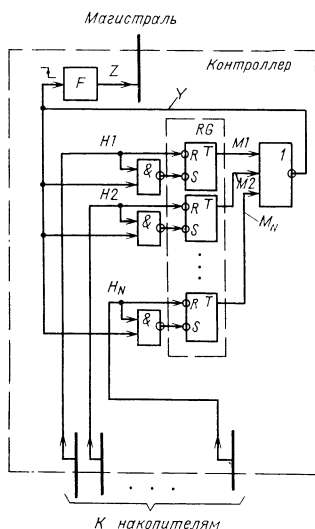


Рис. 1.39. Фрагмент схемы контроллера (рис. 1.38), первый вариант

Нас будет интересовать только решение задачи слежения за периодами отработки заданий накопителями, поэтому варианты построения контроллера рассматриваются именно с этих позиций. В схеме, приведенной на рис. 1.39, сигнал  $H_i$  принимает значение, равное лог. 1 на протяжении выполнения задания накопителем  $i$ .

В исходном состоянии, до момента  $t_1$  (рис. 1.40), накопители  $1-N$  не заняты выполнением заданий,  $H1=H2=\dots=H_N=0$ , поэтому все разряды регистра  $RG$  установлены в 0,  $M1=M2=\dots=M_N=0$ , формирователь импульса  $F$  установлен в 0 сигналом  $Y=1$ , сигнал прерывания  $Z$  центрального процессора отсутствует ( $Z=0$ ).

В момент  $t_1$  накопитель 3 приступает к выполнению задания и устанавливает сигнал  $H3=1$ . Так как  $Y=1$ , то соответствующий триггер  $T$  регистра  $RG$  устанавливается в 1 по входу  $S$ , сигнал  $M3=1$  проходит через элемент ИЛИ—НЕ и переводит сигнал  $Y$  из 1 в 0, закрывая регистр  $RG$  по входам установки 1. По отрицательному фронту сигнала  $Y$  формирователь (таймер)  $F$  начинает отсчет времени. (Если входной сигнал  $Y$  остается в состоянии лог. 0 слишком долго, см. интервал  $t_4-t_8$ , то формируется сигнал  $Z=1$ .)

В момент  $t_2$  начинается работа накопителя 4. При этом, однако, сигнал  $H4=1$  не проходит на соответствующий выход  $M4$  регистра  $RG$ , так как этот регистр закрыт по входам  $S$  сигналом  $Y=0$ . В момент  $t_3$  вступает в действие накопитель 5, формируется сигнал  $H5=1$ , который также не проходит в регистр  $RG$ .

В момент  $t_4$  накопитель 3 успешно завершает выполнение задания (нужная дорожка и сектор найдены) и снимает сигнал  $H3=1$ . Это вызывает переход сигнала  $M3$  из 1 в 0, на выходе элемента ИЛИ—НЕ формируется сигнал  $Y=1$ , таймер  $F$  устанавливается в 0, регистр  $RG$  открывается по входам установки в 1 и принимает новую информацию, сигналы  $M4=1$  и  $M5=1$  с его выходов проходят через элемент ИЛИ—НЕ, в результате регистр вновь закрывается по входам установки в 1, а формирователь  $F$  начинает отсчет нового периода.

В момент  $t_5$  накопитель 5 успешно завершает работу, сигнал  $H5=0$  проходит на выход соответствующего разряда регистра. В данном примере накопитель 4 не справился с заданием и поддерживает сигнал  $H4$  в состоянии

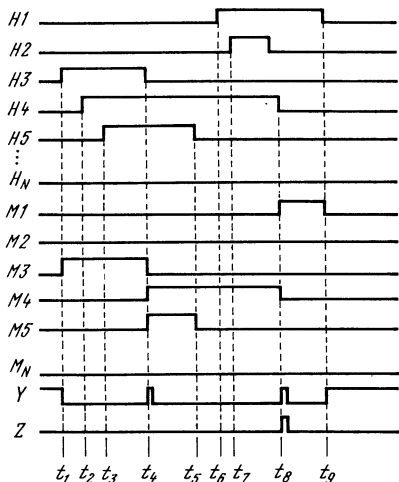


Рис. 1.40. Временные диаграммы работы схемы (рис. 1.39)

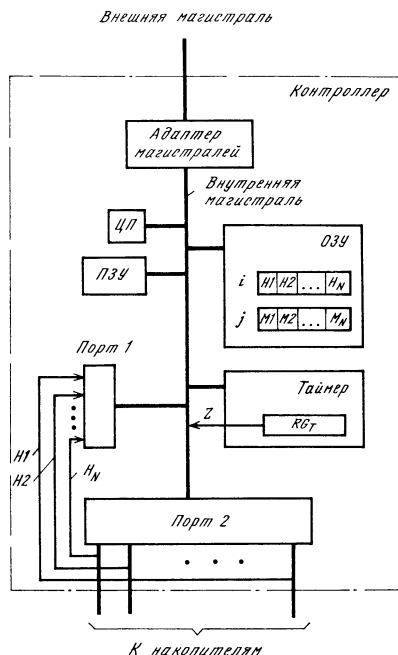


Рис. 1.41. Фрагмент схемы контроллера (рис. 1.38), второй вариант

лог. 1. В моменты  $t_6$  и  $t_7$  в действие вступают накопители 1 и 2, однако сигналы  $H1=1$  и  $H2=1$  не фиксируются в регистре, так как  $Y=0$ .

В момент  $t_8$  на выходе формирователя  $F$  вырабатывается сигнал  $Z=1$ , свидетельствующий о том, что максимально допустимое время ожидания окончания операции исчерпано. Этот сигнал поступает, например, в ЦП микроЭВМ, который считывает по магистрали содержимое регистра  $RG$  (этот регистр выполнен программно-доступным) и определяет номер накопителя, не справившегося с заданием.

Процессор переводит этот накопитель в пассивное состояние, по сигналу  $H4=0$  формируются сигналы  $M4=0$ ,  $Y=1$ , в регистре фиксируется сигнал  $M1=1$ , так как накопитель 1 продолжает, а накопитель 2 уже закончил выполнение задания. По отрицательному фронту сигнала  $Y$  начинается новый отсчет времени, однако в момент  $t_9$  в регистре  $RG$  — нулевой код, сигнал  $Y=1$  поддерживает формирователь  $F$  в пассивном состоянии,  $Z=0$ .

В рассмотренной схеме нет индивидуальных таймеров в каждом канале — в этом, собственно, и заключены ее преимущества. Иными словами, с помощью одного таймера ( $F$ ) удастся контролировать процессы во всех каналах и выявлять среди них «зависшие» (не справившиеся с порученными им заданиями).

Предложенную идею использования общего для всех каналов таймера можно реализовать и на программном уровне. В схеме контроллера, приведенной на рис. 1.41, сигналы  $H1—H_N$  поступают в программно-доступный порт (порт 1), который периодически опрашивается ЦП, результаты опроса отображаются в ячейке  $i$  ОЗУ. Ячейка  $j$  выполняет функцию регистра  $RG$  в схеме на рис. 1.39. Микросхема таймера содержит регистр  $RG_T$ , в который

программно загружается число, определяющее максимально допустимое время ожидания окончания операции. Число в регистре  $RG_T$  с течением времени уменьшается, и, когда оно становится равным 0, формируется сигнал  $Z$  прерывания процессора.

В исходном состоянии в ячейках  $i$  и  $j$  ОЗУ записаны нулевые коды, таймер выключен. Сигналы  $H_k=1$ , обнаруженные при очередном опросе порта 1, переписываются в ячейку  $i$ , а затем в ячейку  $j$ , после чего запись новых единиц в ячейку  $j$  программно запрещается до момента перехода всех ее разрядов в 0. Таймер начинает отсчет времени. По окончании отработки заданий сигналы  $H_k$ , ранее установленные в 1, гасятся, соответствующие разряды ячеек  $i$  и  $j$  безусловно переводятся в 0.

При правильной работе системы нулевой код в ячейке  $j$  устанавливается до того, как сработает таймер. Затем ненулевой код из ячейки  $i$  переписывается в ячейку  $j$ , таймер перезапускается, запись новых единиц в ячейку  $j$  запрещается, и т. д. При «зависании» в одном или нескольких каналах таймер формирует сигнал  $Z$ , процессор считывает содержимое ячейки  $j$  и определяет номер канала, не выполнившего задание.

### **1.18. Подключение пользовательских контроллеров к ПЭВМ типа IBM PC (PS) с выделением им «не освоенных» фирмой IBM областей адресного пространства ввода—вывода [18]**

Персональная ЭВМ (ПЭВМ) типа IBM PC или IBM PS содержит системную плату, на которой размещены основные устройства (ЦП, сопроцессор, память, контроллеры каналов прямого доступа к памяти (ПДП), контроллеры прерываний и пр.), а также соединители для установки дополнительных плат контроллеров, адаптеров, блоков памяти и т. п.

Несмотря на широкий ассортимент дополнительных плат, разработанных разными фирмами, многие пользователи предпочитают создавать свои контроллеры. Такое стремление может быть обусловлено состоянием рынка (дешевле разработать свой контроллер, чем воспользоваться покупным), невозможностью или неэффективностью применения покупных контроллеров и т. п. Всякий раз, когда пользователь ПЭВМ приступает к разработке контроллера, перед ним возникает задача отыскания не занятой области адресов в адресном пространстве ввода—вывода, где предстоит разместить адреса регистров, триггеров и иных программно-доступных элементов, вносимых в систему.

Исторически сложилось так, что адресное пространство ввода—вывода ПЭВМ фирмы IBM имеет весьма ограниченный объем  $2^{10}=1K$  байт, так как для адресации регистров контроллеров используются только 10 разрядов  $A_9-A_0$  ( $A_0$ —младший разряд) адресной шины. Практически вся первая четверть адресного пространства ввода—вывода (область 000—0FF, за исключением нескольких ячеек) зарезервирована фирмой IBM для адресов регистров основных устройств, размещенных на системной плате,—сопроцессора, контроллеров каналов ПДП, контроллеров прерываний, контроллера клавиатуры, таймера и др. Поэтому она практически непригодна для отображения дополнительно вводимых адресов программно-доступных элементов пользовательских контроллеров.

Оставшаяся часть адресного пространства ввода—вывода (область 100—3FF) занята не столь плотно, хотя и здесь «стандарт де-факто» оккупирован все новые районы. В этой области и пришлось бы размещать адреса регистров нашего «самодельного» контроллера (а куда деться?), если бы не рассмотренное далее решение, которое предоставляет пользователю исключительное право владения «личными» областями адресного пространства ввода—вывода достаточного объема, где заведомо нет и не предвидится

Разъемы системной платы ПЭВМ IBM PC (PS)

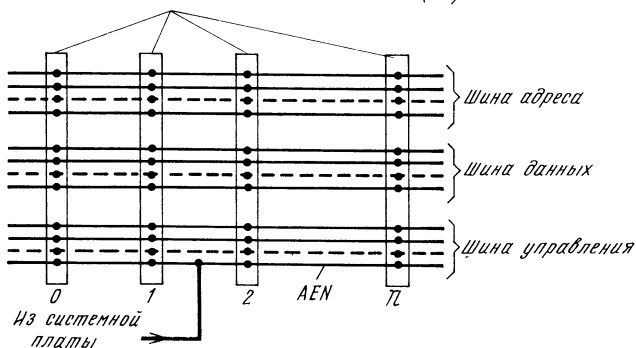


Рис. 1.42. Магистраль типа XT или AT ПЭВМ фирмы IBM

появление конкурирующих адресов «чужих» контроллеров. Прежде чем перейти к этому решению, рассмотрим некоторые существенные особенности магистралей типа AT и XT, используемых в ПЭВМ фирмы IBM.

Магистраль (рис. 1.42) содержит шины адреса, данных и управления. Пользовательские контроллеры устанавливаются в разъемы системной платы. Особый интерес в связи с рассмотренным далее усовершенствованием представляет линия шины управления, по которой передается сигнал AEN (Address Enable — разрешение использования адреса). При нормальной работе ЦП с памятью или устройствами ввода—вывода AEN=0 (напряжение низкого уровня). Условие AEN=1 означает, что магистраль передана в распоряжение канала ПДП, который может работать в режимах чтения и записи данных в ячейки памяти.

В режиме чтения выполняется ряд циклов, в каждом из которых в шине адреса устанавливается адрес очередной ячейки памяти, а в шине управления — сигнал —MEMR (чтение из памяти, знак «—» означает активный низкий уровень сигнала); после выдачи из памяти в шину данных считанного кода формируется сигнал —IOW (запись в устройство ввода—вывода), по которому в контроллер, осуществляющий обмен, записывается код, считанный из памяти. Остальные устройства (контроллеры ввода—вывода) не должны реагировать на сигнал —IOW.

В режиме записи каждый цикл сопровождается сигналами —IOR, —MEMW (чтение из устройства ввода—вывода, запись в память) и выдачей адреса ячейки памяти и кода данных в соответствующие шины магистральной. По сигналу —IOR контроллер—источник информации выдает в шину данных очередной код для записи в память, остальные контроллеры не должны реагировать на этот сигнал.

Таким образом, при работе канала ПДП AEN=1, сигналы —IOW и —IOR не имеют отношения к передаваемому адресу (он принадлежит адресному пространству памяти, изолированному от адресного пространства ввода—вывода), в отличие от обычного режима, когда AEN=0 и процессор записывает или считывает содержимое выбранного по адресу регистра. В последней ситуации адрес принадлежит адресному пространству ввода—вывода (000—3FF). Сигнал AEN=1 позволяет исключить ложное опознание собственного адреса контроллерами ввода—вывода, в том числе размещенными непосредственно на системной плате. Фактически этот сигнал можно рассматривать как признак запрета дешифрации адреса контроллерами.

Предлагаемая доработка системной платы ПЭВМ (рис. 1.43) связана с разделением общей управляющей линии передачи сигнала AEN на отдельные

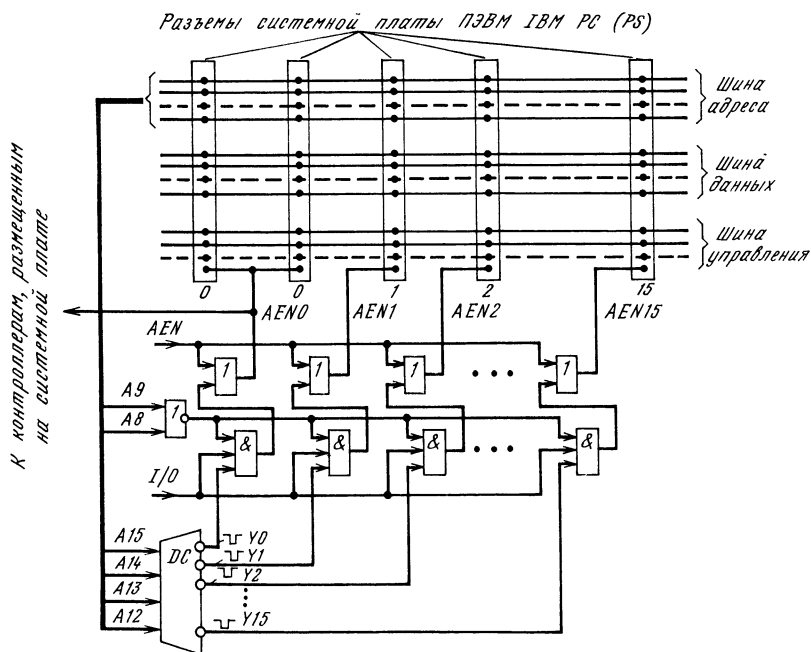


Рис. 1.43. Усовершенствованная системная плата ПЭВМ фирмы IBM с магистралью типа XT или AT.

Пользователю предоставлен ряд «экологических ниш» в адресном пространстве ввода—вывода с гарантированным отсутствием конкурентов

части, каждая из которых охватывает один или несколько разъемов, по усмотрению пользователя. Выход источника сигнала AEN (того же, что и в предыдущей схеме) отключен от соответствующей линии управления и соединен с верхними входами двухвходовых элементов ИЛИ. В схеме использован сигнал I/O, который равен лог. 1 при обращении процессора к адресному пространству ввода—вывода и лог. 0 во всех остальных ситуациях. Контроллеры, размещенные непосредственно на системной плате, вместо сигнала AEN теперь отслеживают сигнал AEN0 (при AEN0=1 адресное обращение к ним запрещено).

В схеме задействованы разряды A15—A12 адреса, хотя ранее отмечалось, что фирма IBM по историческим причинам продолжает ограничиваться десятью разрядами A9—A0, чтобы сохранить преемственность моделей ПЭВМ и гарантировать совместимость с ранее разработанными контроллерами. Однако благодаря «прозорливости» разработчиков микропроцессоров 8086, 80286, 80386 предусмотрена возможность использования 16-разрядного адреса при выполнении команд ввода—вывода. В предлагаемом решении используется расширенное адресное пространство ввода—вывода при сохранении совместимости доработанной ПЭВМ с ранее разработанными контроллерами, воспринимающими 10-разрядный адрес.

Схема (рис. 1.43) формирует сигналы AEN0—AEN15 согласно следующим правилам.

1. При работе канала ПДП AEN0=AEN1=...=AEN15=1. Это правило в полной мере отвечает существующему стандарту фирмы IBM и, как было показано, позволяет устранить ложные опознания адреса контроллерами.

Сигнал  $AEN=1$  проходит через элементы ИЛИ и поступает на все разъемы системной платы, а также к контроллерам, непосредственно размещенным на этой плате.

2. При обращении процессора в адресное пространство памяти  $AEN_0=AEN_1=\dots=AEN_{15}=0$ . Это правило также отвечает существующему стандарту. В данном случае  $AEN=0$ ,  $I/O=0$ , элементы И закрыты, на входах элементов ИЛИ присутствуют сигналы лог. 0, поэтому на их выходах также сформированы сигналы лог. 0.

3. При обращении процессора во вторую, третью и четвертую четверти адресного пространства ввода—вывода (по адресам 100—3FF, соответствующим ненулевым комбинациям сигналов в разрядах  $A_9$  и  $A_8$ )  $AEN_0=AEN_1=\dots=AEN_{15}=0$ . И это правило полностью отвечает стандарту фирмы IBM (еще немного терпения, и мы отклонимся с проложенного фирмой пути). В данной ситуации на выходе элемента ИЛИ—НЕ присутствует сигнал лог. 0, элементы И закрыты по верхним входам,  $AEN=0$ , поэтому на выходах элементов ИЛИ сформированы сигналы лог. 0. (Адрес передается по шине в прямом коде.)

4. При обращении процессора в первую четверть адресного пространства ввода—вывода (область 000—0FF) один из сигналов  $AEN_0—AEN_{15}$  принимает нулевое значение, остальные—единичные. Положение нулевого сигнала определяется результатом дешифрации кода  $A_{15}—A_{12}$ .

Если этот код нулевой, что соответствует работе обычной ПЭВМ фирмы IBM, то  $Y_0=0$ ,  $Y_1=Y_2=\dots=Y_{15}=1$ , поэтому с учетом того, что  $I/O=1$ ,  $A_9=A_8=0$ , получим  $AEN_0=0$ ,  $AEN_1=AEN_2=\dots=AEN_{15}=1$ . Таким образом, два разъема системной платы, на которые поступает сигнал  $AEN_0$ , могут использоваться как для установки «фирменных» устройств со стандартными адресами из диапазона 000—0FF (если эти устройства не размещены непосредственно на системной плате), так и для подключения любых разработанных ранее устройств с адресами из диапазона 100—3FF. Иными словами, эти разъемы как бы заимствованы из недоработанной ПЭВМ.

Оставшиеся разъемы (1—15) также гарантируют совместимость с ранее разработанными контроллерами, чьи собственные 10-разрядные адреса принадлежат области 100—3FF (не зря же столь тщательно соблюдались правила, установленные фирмой, см. пп. 1—3). Однако в эти разъемы можно устанавливать и новые (несовместимые с обычной ПЭВМ) контроллеры, реагирующие на 10-разрядный адрес  $A_9—A_0$ . Несовместимость состоит в том, что новые контроллеры занимают одну и ту же область адресов (независимо друг от друга), выделенную фирмой IBM для устройств системной платы: 000—0FF. Другими словами, при разработке каждого такого контроллера предполагалось, что ему (и только ему!) в адресном пространстве ввода—вывода выделена область 000—0FF, содержащая 256 ячеек, и этими ячейками можно свободно распоряжаться, не беспокоясь о конфликтах с конкурирующими адресами других устройств.

Противоречия здесь нет. Действительно, если процессор обращается к адресному пространству ввода—вывода с ненулевым кодом в разрядах  $A_{15}—A_{12}$  и нулевым—в разрядах  $A_9$  и  $A_8$  (при этом  $AEN=0$ ,  $I/O=1$ ), то сигналом  $AEN_i=0$  ( $i=1, 2, \dots, 15$ ) выбирается только один— $i$ -й разъем, а на остальных присутствуют сигналы  $AEN_j=1$ , т. е. соответствующим устройствам (в том числе размещенным непосредственно на системной плате и реагирующим на сигнал блокировки  $AEN_0=1$ ) запрещается анализировать адрес и, следовательно, участвовать в обмене данными. Таким образом, «системная» четверть адресного пространства ввода—вывода (область 000—0FF) отбирается в пользу контроллера, установленного в  $i$ -й разъем, на время обращения к нему со стороны процессора.

В данном примере в ПЭВМ допускается установить до 15 нестандартных контроллеров, каждый из которых может содержать до 256 индивидуально адресуемых программно-доступных элементов—регистров, триггеров, внутренних точек схемы и т. п. Число этих элементов можно увеличить в четыре

раза, если воспользоваться разрядами A11 и A10 адреса. Тогда контроллеру будут отведены следующие четыре области по 256 ячеек каждая: 000—0FF; 400—4FF; 800—8FF; C00—CFF (напомним, что нестандартный контроллер реагирует только на те коды, в которых  $A9=A8=0$ ).

Если сигнал  $AEN_i$  ( $i=1, 2, \dots, 15$ ) поступает на несколько разъемов, то соответствующие контроллеры не должны иметь одинаковых адресов в выделенном им диапазоне (256 или 1K ячеек).

## Глава 2

# Контроль, диагностика, восстановление, поиск информации

## 2.1. Схемы выходных каскадов мажоритарно-резервированных устройств [19]

Мажоритарное резервирование используют для повышения надежности устройств. Устройство имеет три или более идентичных канала. Результаты вычислений в каждом канале сравниваются между собой. С помощью мажоритарного логического элемента путем голосования выбирается наиболее достоверный результат. В простейшем случае этот результат представлен единственным сигналом, который используется, например, для включения — выключения электродвигателя (рис. 2.1).

Проблема заключается в том, что мажоритарный элемент и выходной транзистор (представленные единственным экземплярами) с точки зрения надежности составляют наиболее слабое звено схемы — их отказы приводят к потере работоспособности всей системы. Вероятнее всего, при таком отказе электродвигатель будет либо постоянно выключен, либо постоянно включен. В зависимости от требований к системе наиболее нежелательно либо ложное (несанкционированное) включение электродвигателя, либо его ложное выключение. В первой ситуации можно применить дублированный выходной каскад с последовательно соединенными выходными транзисторами VT1 и VT2 (рис. 2.2), во второй — аналогичную схему с параллельно включенными транзисторами.

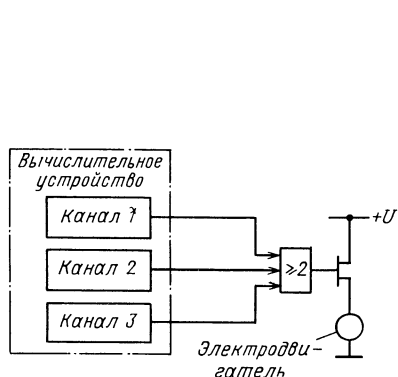


Рис. 2.1. Резервированная система (первый вариант)

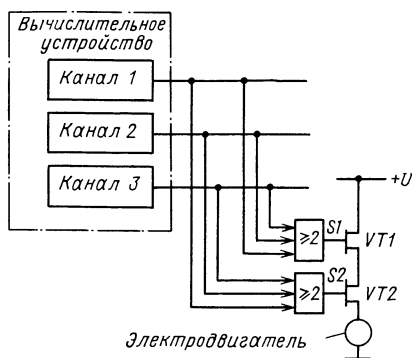


Рис. 2.2. Резервированная система (второй вариант)

Система (рис. 2.2) сохраняет работоспособность при ошибках, эквивалентных короткому замыканию одного из транзисторов, так как оставшийся исправный выходной канал (мажоритарный элемент — транзистор) обеспечивает правильное управление электродвигателем. Если в системе возникла неисправность, эквивалентная постоянному размыканию одного из выходных транзисторов, то электродвигатель не сможет включиться, когда это будет предписано вычислительным устройством.

Это, как предполагалось, не столь существенный отказ, во всяком случае, он не приводит к нежелательным последствиям для пользователя. То же справедливо и при неисправностях, эквивалентных постоянному размыканию обоих транзисторов. Напротив, при одновременном ложном замыкании обоих транзисторов электродвигатель постоянно включен «вопреки указанию» вычислительного устройства, и пользователь (или система защиты более высокого уровня) вынужден принимать экстренные меры по выключению электродвигателя во избежание нежелательных последствий.

Вероятность отказа, эквивалентного одновременному короткому замыканию обоих транзисторов, сравнительно мала. Например, если вероятность отказа, эквивалентного короткому замыканию транзистора в одном канале за некоторый период времени, равна  $10^{-2}$ , то вероятность одновременного отказа в двух каналах в течение того же периода времени составляет  $10^{-4}$ . Если увеличивать число последовательно соединенных транзисторов (и соответствующих мажоритарных элементов), то вероятность ложного включения электродвигателя будет быстро уменьшаться. Аналогично параллельное включение транзисторов, управляющих подачей энергии на электродвигатель, уменьшает вероятность ложного выключения, если именно оно представляется пользователю особо нежелательным.

В схеме, показанной на рис. 2.3, использованы два резервированных вычислительных устройства, работающих синхронно. В отличие от предыдущей схемы, при отказе, эквивалентном постоянному замыканию или размыканию любого выходного транзистора, система сохраняет работоспособность. Кроме того, при обнаружении отказа одного из устройств оно может быть на время ремонта удалено из системы без потери ее работоспособности.

Схема, показанная на рис. 2.4, близка к рассмотренной ранее (см. рис. 2.2), однако дополнительно позволяет контролировать работу выходных транзисторов VT1 и VT2. Сигналы S1 и S2 поступают с выходов мажоритарных элементов, входы которых соединены с выходами 3-канального резервированного вычислительного устройства.

Предположим сначала, что низкочастотный генератор G коротких импульсов выключен, на его выходах сформированы сигналы  $A=B=1$ . В этом

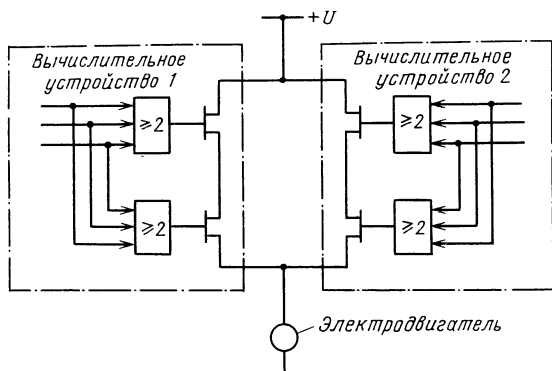


Рис. 2.3. Резервированная система (третий вариант)

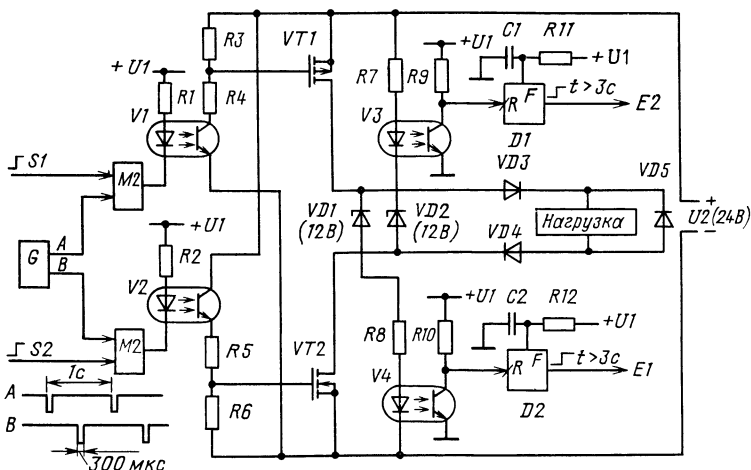


Рис. 2.4. Выходной каскад резервированной системы с контролем работоспособности его элементов

случае элементы Иключающее ИЛИ (M2) выполняют функции инверторов по отношению к сигналам S1 и S2. При правильной работе системы сигналы S1 и S2 всегда принимают одинаковые значения. Если  $S1=S2=0$ , то оптроны V1 и V2 выключены, напряжения исток—затвор транзисторов VT1 и VT2 равны 0, эти транзисторы закрыты, ток через нагрузку практически отсутствует. При  $S1=S2=1$  оптроны V1, V2 и транзисторы VT1, VT2 включены, через нагрузку протекает рабочий ток.

Состояния оптронов V3 и V4 определяются состояниями транзисторов VT2 и VT1: если включен транзистор VT2 (VT1), то включен оптрон V3 (V4); при выключении транзисторов VT2 и VT1 оптроны V3 и V4 также выключаются. Таймер D1 (D2) устанавливается в 0 по положительному фронту сигнала на его динамическом входе R и начинает отсчет времени в соответствии с параметрами внешней RC-цепи R11C1 (R12C2). Если в течение 3 с на вход R не поступил очередной положительный фронт сигнала, то на выходе таймера формируется сигнал лог. 1.

Диоды VD3 и VD4 предотвращают возможное протекание тока через входные цепи оптронов V3 и V4 при выключенных транзисторах VT1 и VT2. Действительно, в отсутствие диодов VD3 и VD4 (если бы они были заменены перемычками) при выключенных транзисторах VT1, VT2 и неблагоприятном сочетании параметров ток мог бы протекать по цепи  $(+U2) - R7 - V3 - VD2 -$  нагрузка с параллельно включенным диодом  $VD5 - VD1 - R8 - V4 - (-U2)$ . Диод VD5 служит для гашения импульса напряжения обратной полярности в период прекращения тока через нагрузку, если она обладает заметной индуктивностью.

Рассмотрим теперь работу схемы с учетом действия тестирующих импульсных сигналов A и B от генератора G. При постоянных сигналах S1 и S2 с поступлением отрицательного импульса A сигнал на выходе верхнего элемента Иключающее ИЛИ меняет значение, поэтому состояния оптрона V1 и транзистора VT1 меняются на противоположные. С окончанием импульса A эти элементы возвращаются в прежние состояния. В результате на динамическом входе таймера D2 формируется короткий импульс той или иной полярности, по положительному фронту которого таймер устанавливается

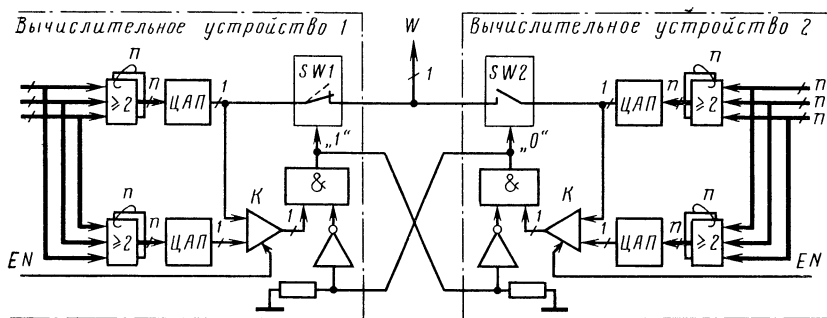


Рис. 2.5. Выходной каскад резервированной системы с отключением неисправного канала

в 0 или подтверждает установленное ранее нулевое состояние. В дальнейшем процесс повторяется с периодом 1 с.

Другой канал V2—VT2—V3—D1 работает аналогично, но со сдвигом во времени на 0,5 с. Таким образом, при нормальной работе схемы на выходах таймеров D1 и D2 поддерживаются сигналы  $E1=E2=0$ . При отказе транзисторов VT1, VT2 или иных элементов, когда импульсы установки 0 не поступают на входы таймеров в течение периода времени, превышающего 3 с, эти таймеры формируют единичные сигналы ошибок E1 и E2.

Отметим, что тестирующие импульсы при нормальной работе схемы вызывают кратковременные отключения нагрузки. Если в качестве нагрузки использован электродвигатель, реле или иное инерционное устройство, то короткие импульсы отключения остаются «незамеченными». Это же относится и к коротким импульсам включения, когда один из транзисторов VT1 или VT2 постоянно открыт в результате неисправности, а второй периодически открывается при ежесекундном тестировании.

Система, показанная на рис. 2.5, содержит два резервированных вычислительных устройства, работающих синхронно. В отличие от предыдущих схем, выходные сигналы вычислительных устройств 1 и 2 представлены в аналоговой форме. В зависимости от положения аналоговых ключей SW1 и SW2, показанных на рисунке в виде механических выключателей, выходной аналоговый сигнал W снимается с верхнего цифрово-аналогового преобразователя (ЦАП) первого или второго вычислительного устройства.

Аналоговые ключи SW1 и SW2 управляются сигналами с выходов RS-триггера, построенного на двух элементах И и двух элементах НЕ. Если система работает правильно, то RS-триггер находится в произвольном (устойчивом) состоянии. Когда аналоговые компараторы К закрыты ( $EN=0$ ), с их выходов снимаются сигналы лог. 1. После выполнения вычислений и преобразования мажоритарно-восстановленных  $n$ -разрядных кодов в аналоговые сигналы последние сравниваются компараторами, результаты сравнения выдаются на установочные входы триггера в моменты поступления стробирующих сигналов  $EN=1$ .

Если сравниваемые аналоговые сигналы достаточно близки друг другу, то компаратор формирует сигнал лог. 1, в противном случае (при неудовлетворительной работе устройства)—сигнал лог. 0, который воздействует на RS-триггер. В результате триггер переходит в новое устойчивое состояние (или подтверждает старое), при котором сигнал W снимается с выхода правильно работающего вычислительного устройства, а отказавшее устройство логически отключается.

Отказавшее вычислительное устройство можно, не выключая напряжения питания, извлечь из разряда и передать в ремонт. При этом оставшееся

исправное устройство продолжает работу и «не замечает» удаления соседа, так как сигнал лог. 0 с выхода противоположного плеча триггера «плавно» подменяется собственным сигналом лог. 0, присутствующим на входе элемента НЕ благодаря его привязке к шине нулевого потенциала (лог. 0) через резистор.

## 2.2. Самоконтролируемый выходной каскад мажоритарно-резервированного устройства [20]

Мажоритарно-резервированное устройство содержит три идентичных канала, к которым подключен выходной каскад на элементах D1—D7 (рис. 2.6). Мажоритарный элемент D1 формирует на выходе сигнал  $V=1$ , если на его входах две или три единицы, и сигнал  $V=0$ , если на входах два или три нуля (голосование по большинству). Компаратор D2 вырабатывает сигнал  $U=1$  при взаимном равенстве входных сигналов, т. е. при  $Q1=Q2=Q3=0$  или  $Q1=Q2=Q3=1$ .

Сумматор по модулю два D3 формирует сигнал  $W=1$ , если число единиц, поданных на его входы, нечетно, и  $W=0$  при четном числе единиц. Последовательная цепь из элементов Иключающее ИЛИ D4—D6 суммирует по модулю два сигналы  $V$ ,  $U$ ,  $W$  и тестовый сигнал  $T$ , элемент НЕ D7 инвертирует выходной сигнал. Результирующие сигналы  $T'$  и  $U$  отображают состояние устройства (нет ошибки—есть ошибка), причем обнаруживаются неисправности элементов D1—D7. Напомним, что трехканальное резервирование позволяет исправлять все одиночные и некоторые двойные ошибки.

При отсутствии неисправностей аппаратуры и безошибочных вычислениях сигналы  $Q1—Q3$  одинаковы (см. строки 1—4 табл. 2.1). Тестовый сигнал  $T$ , подаваемый на контрольный вход устройства, суммируется по модулю два с нечетным числом единиц, поэтому с учетом инвертирования результата получим, что  $T'=T$ ,  $U=1$ . Внешняя система контроля (на рисунке не показана) в каждом такте работы (при  $T=0$  и  $T=1$ , см. рис. 2.6, б) принимает к сведению, что ошибок на выходах каналов 1—3 нет ( $U=1$ ), а сам выходной каскад (элементы D1—D7) исправен ( $T'=T$ ).

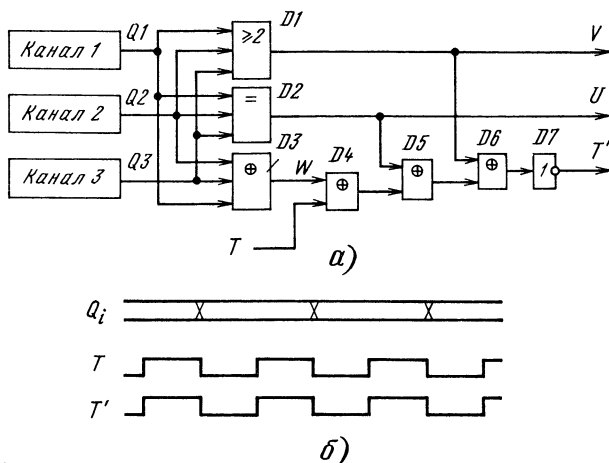


Рис. 2.6. Самоконтролируемый выходной каскад (а) мажоритарно-резервированного устройства и временные диаграммы (б) его работы в отсутствие ошибок

Таблица 2.1

Номер строки	Входные сигналы				Выходные сигналы						
	Q1	Q2	Q3	T	V	U	W	T'			
1	0	0	0	0	0	1	0	0	} Ошибок нет		
2	0	0	0	1	0	1	0	1			
3	1	1	1	0	1	1	1	0			
4	1	1	1	1	1	1	1	1			
5	0	0	1	0	0	0	1	0	} Ошибка типа Ложная 1		
6	0	0	1	1	0	0	1	1			
7	0	1	0	0	0	0	1	0			
8	0	1	0	1	0	0	1	1			
9	1	0	0	0	0	0	1	0	} Ошибка типа Ложный 0		
10	1	0	0	1	0	0	1	1			
11	1	1	0	0	1	0	0	0			
12	1	1	0	1	1	0	0	1			
13	1	0	1	0	1	0	0	0	} Ошибка типа Ложный 0		
14	1	0	1	1	1	0	0	1			
15	0	1	1	0	1	0	0	0			
16	0	1	1	1	1	0	0	1			

Если на выходах каналов 1—3 присутствует одиночная ошибка (строки 5—16 таблицы) типа Ложная 1 или Ложный 0, то  $U=0$ , внешняя система контроля воспринимает этот сигнал и предпринимает соответствующие действия, например включает индикатор на приборной панели автомобиля, где применено данное резервированное устройство. Водитель получает информацию о том, что устройство хотя и работоспособно, но запас надежности уменьшился; для его восстановления нужен ремонт отказавшего канала.

При наличии одиночных ошибок в каналах (строки 5—16 таблицы) число единиц на выходах  $V$ ,  $U$  и  $W$  нечетно, как и в случае правильной работы устройства (строки 1—4). Поэтому  $T'=T$  и, вероятно, возникают сомнения в необходимости формирования этого сигнала (ведь он не отражает присутствие ошибки!). Чтобы убедиться в целесообразности введения элементов D3—D7, рассмотрим наиболее вероятные ситуации, связанные с отказами элементов выходного каскада.

1. В результате отказа входной цепи элемента D1, D2 или D3 один из сигналов Q1—Q3 принимает постоянное значение (0 или 1). Эта ситуация, по существу, эквивалентна рассмотренной ранее, когда предполагалось, что ошибка присутствует в одном из каналов.

2. Вследствие отказа выхода элемента D1 или входа D6 сигнал  $V$  постоянно равен 0 или 1. Если  $V=0$ , то в тех ситуациях, когда этот сигнал ожидается единичным (см. строки 3, 4, 11—16 таблицы), его отсутствие проявляется в том, что нарушается нечетность числа единиц в группе  $V$ ,  $U$ ,  $W$ , поэтому тестовый сигнал  $T$  преобразуется в сигнал  $T'$  с изменением фазы, что регистрируется внешней системой контроля. То же наблюдается при  $V=1$  (см. строки 1, 2, 5—10).

3. Ошибка типа Ложный 0 или 1 присутствует на выходе D2 из-за отказа этого элемента или входа D5. Признак  $U=0$  воспринимается внешней системой контроля как отказ одного из каналов, поэтому ошибка не останется незамеченной. Более того, в отсутствие ошибок в каналах 1—3 (строки 1—4 таблицы) нарушается общая нечетность числа единиц в группе сигналов  $V$ ,  $U$ ,  $W$ , поэтому сигнал  $T'$  не совпадает с  $T$ .

При  $U=1$  и отсутствии иных неисправностей (см. строки 1—4 таблицы) ошибка остается незамеченной, так как она совпадает с ожидаемым значением сигнала. Однако при отказе одного из каналов (строки 5—16), когда

4. В случае отказа выхода D3 или входа D4  $W \equiv 0$  или 1. Обе ситуации обнаруживаются благодаря нарушению условия нечетности числа единиц в группе V, U, W, когда ожидаемые значения W не совпадают с фактическими. Вследствие этого сигнал T, проходя через цепь элементов D4—D7, инвертируется, что регистрируется внешней системой контроля.

Таким образом, исключается неконтролируемое накопление ошибок в устройстве. Если бы не было цепи D3—D7, то, например, при  $U \equiv 1$  (из-за отказа выхода элемента D2) внешняя система контроля оставалась бы в неведении при отказе канала 1 и т. п.

В интеллектуальной системе электронного зажигания двигателя внутреннего сгорания высоковольтный импульс подается на свечу в строго заданное время, исчисляемое по определенному алгоритму от момента прохождения маховиком некоторого положения. Этот момент фиксируется датчиком положения, который для надежности дублируется и вырабатывает на двух независимых выходах синхронные импульсы. Если один из выходов дублированного датчика отказал, то система электронного зажигания пользуется информацией с другого, исправного выхода и предупреждает человека, что остался только один исправный канал.

[illegible]

53

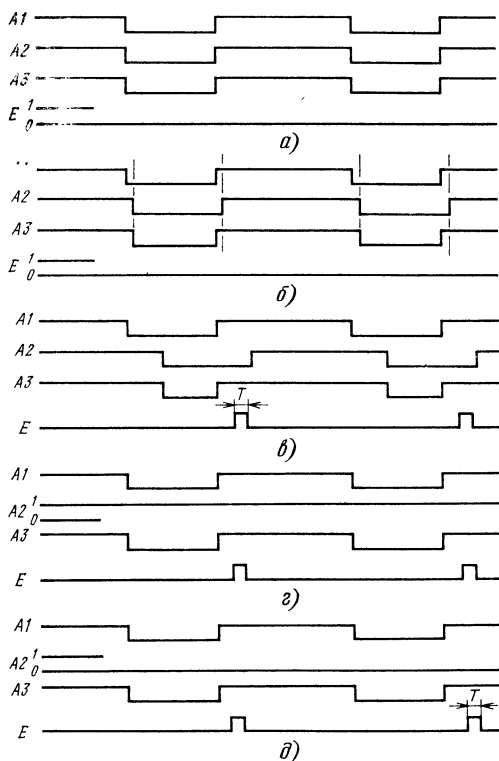


Рис. 2.8. Временные диаграммы работы схемы на рис. 2.7: *a*—при отсутствии сдвига между сигналами *A1* и *A2*; *б*—при незначительном сдвиге; *в*—при недопустимо большом сдвиге; *г*, *д*—при отказе второго канала ( $A2=1$ ,  $A2=0$ )

результрующий сигнал  $A3$ . Если сигналы  $A1$  и  $A2$  одинаковы (рис. 2.8, *a*), то результирующий сигнал  $A3$  совпадает с каждым из них с точностью до задержек элементов  $D1$ ,  $D3$  (скважность отрицательных импульсов на рисунке для наглядности уменьшена). Сигнал ошибки  $E$  не вырабатывается, т. е. равен 0. При незначительном смещении импульсов  $A1$  и  $A2$  друг относительно друга (рис. 2.8, *б*) сигнал ошибки также не вырабатывается, результирующий импульс  $A3$  более узкий, чем каждый из входных, так как они логически суммируются:  $A3 = A1 + A2$ .

С увеличением смещения импульсов  $A1$  и  $A2$  (рис. 2.8, *в*) результирующий сигнал  $A3$  продолжает сужаться, при этом на выходе ошибки формируется сигнал  $E$ —последовательность положительных импульсов длительностью, меньшей или равной периоду  $T$  сигнала тактового генератора  $G$ .

Если сигнал  $A2$  ( $A1$ ) постоянно равен 1 (рис. 2.8, *г*) или 0 (рис. 2.8, *д*), то на результирующий выход передается сигнал  $A3 = A1$  ( $A3 = A2$ ), на выходе ошибки формируется последовательность импульсов  $E$  длительно-стью  $T$ .

Схема приводится в исходное состояние при воздействии кратковременного сигнала Сброс=0. Все триггеры, за исключением триггера  $D6.1$ , устанавливаются в 0, триггер  $D6.1$ —в 1 (на его нулевом плече сигнал лог. 0). При поступлении отрицательных импульсов  $A1$  и  $A2$  на выходах элементов И—НЕ  $D1.1$  и  $D1.2$  формируются сигналы лог. 1,  $A3=0$ . Триггеры  $D3.1$  и  $D3.2$  устанавливаются в 1, так как на их  $D$ -входы поданы сигналы лог. 1 (см. знак «+» на рис. 2.7).

При установке в 1 триггера D3.1 или D3.2 срабатывает элемент ИЛИ—НЕ D4.1, на его выходе формируется сигнал лог. 0. Триггеры D6.1 и D6.2 осуществляют привязку этого сигнала к синхроимпульсу от генератора G и задержку на один период T. К моменту формирования сигнала лог. 1 на выходе триггера D6.2 на выходе элемента Иключающее ИЛИ (M2) D5.1 сформирован сигнал лог. 0 (при правильной работе) или лог. 1 (при недопустимо большом сдвиге входных сигналов или при отказе одного из каналов). Поэтому при неудовлетворительном состоянии входных сигналов срабатывает элемент И D7.3, на его выходе формируется сигнал лог. 1.

В следующем такте сигнал лог. 1 с выхода триггера D6.2 переписывается в триггер D6.3, сигнал лог. 0 с его нулевого выхода проходит через элемент И D7.2 и приводит триггеры D3.1, D3.2, D6 в исходное состояние (триггеры D3.3 и D3.4 установлены в 0 в начале цикла сигналом лог. 0 с выхода элемента D4.1). В моменты окончания отрицательных импульсов A1 и A2 триггеры D3.3 и D3.4 устанавливаются в 1,  $A3=1$  по первому из этих событий.

При отказе одного из входных каналов сигнал A1 или A2 перестает меняться, поэтому соответствующий триггер D3.3 или D3.4 постоянно находится в состоянии 0 (сигнал установки 0 периодически поступает на его вход). Таким образом, запирается элемент И—НЕ D1.1 или D1.2 и предотвращается влияние отказавшего канала на выходной сигнал A3.

## 2.4. Обнаружение ошибок при последовательной передаче информации между двумя устройствами [22]

При последовательной передаче информации между двумя устройствами в линии связи могут возникать ошибки. Для их регистрации обычно используют такие способы: дублирование аппаратуры канала передачи информации со сравнением данных из разных каналов; введение в информационную посылку контрольных разрядов (кодов), в простейшем случае разряда, дополняющего число передаваемых единиц до четного или нечетного.

Оба способа могут оказаться неудовлетворительными, первый — в силу больших аппаратных затрат, второй — из-за ограниченной способности обнаружения ошибок при контроле по четности (нечетности) или из-за сложности схем шифрации и дешифрации, когда применяют более совершенные методы контрольного суммирования. Кроме того, при использовании контрольного суммирования ошибка фиксируется не в момент ее появления, а только после приема и анализа всей переданной информационной посылки.

В предлагаемой системе передачи информации (рис. 2.9) ошибка фиксируется в момент ее появления; дублирования канала связи не требуется, схемы

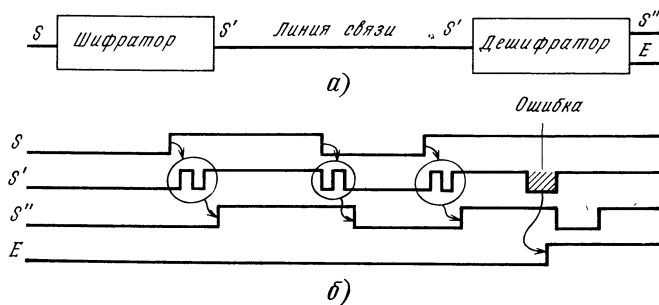


Рис. 2.9. Система передачи информации по линии связи (а) и временные диаграммы сигналов (б).

Шифратор заменяет каждый фронт исходного сигнала S тремя фронтами сигнала S' (обведены овалами), передаваемого по линии связи, так что дешифратор может отличить правильный сигнал от ошибочного (заштрихован) и зафиксировать момент появления ошибки (E=1)

шифрации и дешифрации относительно просты. Обнаружение ошибок достигается за счет уменьшения пропускной способности канала связи. Сигналы, выдаваемые в линию связи, преднамеренно «искажаются» шифратором — каждый фронт исходного сигнала  $S$  преобразуется в три фронта сигнала  $S'$ . Иными словами, прежде чем перевести сигнал  $S'$  в новое устойчивое состояние, шифратор формирует короткий импульс, предупреждающий о предстоящем переводе. Дешифратор, в свою очередь, следит за тем, чтобы не было «простых» переходов сигнала  $S'$  из 0 в 1 и наоборот; каждый переход должен сопровождаться коротким «лишним» импульсом. Сигнал ошибки, наведенный в линии связи (на рис. 2.9, б этот сигнал заштрихован), надо полагать, не будет столь удачным по форме, чтобы в полной мере отвечать принятым в системе правилам кодирования. Поэтому его можно отличить от полезного сигнала и сформировать сигнал обнаружения ошибки  $E=1$ .

Система передачи информации (рис. 2.9) может быть синхронной или асинхронной в зависимости от того, подключены ли шифратор и дешифратор к общему или отдельным независимым источникам синхросигналов. Можно обойтись и без источника (источников) синхросигналов, если для отсчета временных интервалов использовать, например, одновибраторы с времязадающими RC-цепями. Если шифратор и дешифратор подключены к разным источникам синхросигналов, то для надежной работы системы частота сигналов синхронизации дешифратора выбирается достаточно высокой, чтобы короткие импульсные сигналы  $S'$  соответствовали по длительности, например, 8 или 16 периодам синхросигнала.

Рассмотрим синхронную систему, как наиболее простую. Схемы шифратора и дешифратора приведены на рис. 2.10 и 2.11, временные диаграммы работы системы — на рис. 2.12.

В исходном состоянии, до момента  $t_0$  (рис. 2.12), на входе шифратора присутствует сигнал  $S=0$ , на его выходе сформирован сигнал  $S'=0$ , дешифратор выдает на информационный выход сигнал  $S''=0$ , признак обнаружения ошибки  $E=0$  отображает правильную работу системы передачи информации.

В момент  $t_0$  на вход шифратора поступает сигнал  $S=1$ . Привязка этого сигнала к синхросигналу  $CLK$  (общему для шифратора и дешифратора) осуществляется D-триггером D1 (рис. 2.10) в моменты поступления положительных фронтов синхросигнала. В результате сигнал  $S'$  на выходе шифратора при правильной работе системы передачи информации также привязан к синхросигналу  $CLK$ . Триггеры D2 и D3 образуют сдвиговой регистр, элементы Иключающее ИЛИ (M2) D4 и D5 формируют сигнал  $S'$  нужной формы (см. рис. 2.9).

В момент  $t_1$  (рис. 2.12) формируется сигнал  $S'=1$ , который поступает в дешифратор и проходит через элемент И—ИЛИ D3 на информационный вход D-триггера D4. В момент  $t_2$  по положительному фронту синхросигнала  $CLK$  сигнал  $A1=1$  записывается в триггер D4 ( $A2=1$ ). Сигналы A5 и A6 переходят в состояние лог.1, инвертор D2 закрывает верхний элемент И схемы И—ИЛИ D3, при этом нижний элемент И этой схемы открывается,  $A1=A2=1$ . (Переходные процессы могут сопровождаться кратковременными импульсами, показанными в виде утолщенных отрезков на временных диаграммах сигналов A1, A7, A8. Однако эти импульсы формируются уже после перехода схемы в новое устойчивое состояние, поэтому они не могут вызвать появления каких-либо сбойных ситуаций.)

В моменты  $t_3$  и  $t_4$  сигнал  $A2=1$  распространяется через триггеры D6 и D10, на информационном выходе дешифратора формируется сигнал  $S''=1$ . На этом заканчивается начальный этап передачи сигнала лог.1 с входа на выход системы ( $S \rightarrow S''$ ). Из временных диаграмм следует, что сигнал A8 на данном этапе равен 0, если не считать кратковременных импульсов, связанных с переходными процессами. Эти импульсы, как отмечалось, не нарушают правильность работы схемы, триггер D12 остается в состоянии  $E=0$ , соответствующем отсутствию ошибки.

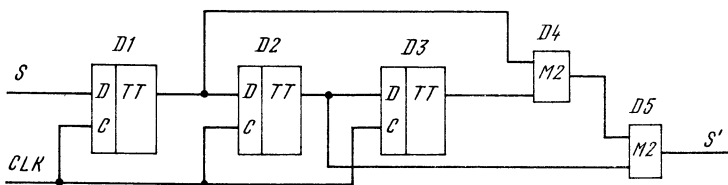


Рис. 2.10. Схема шифратора (рис. 2.9)

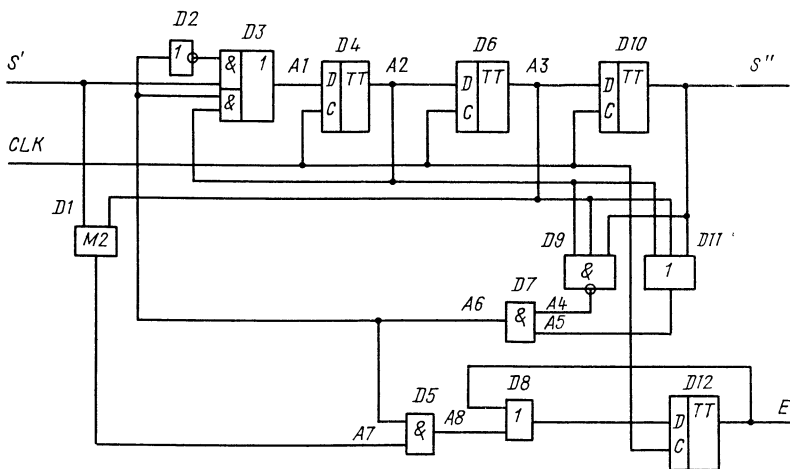


Рис. 2.11. Схема дешифратора (рис. 2.9)

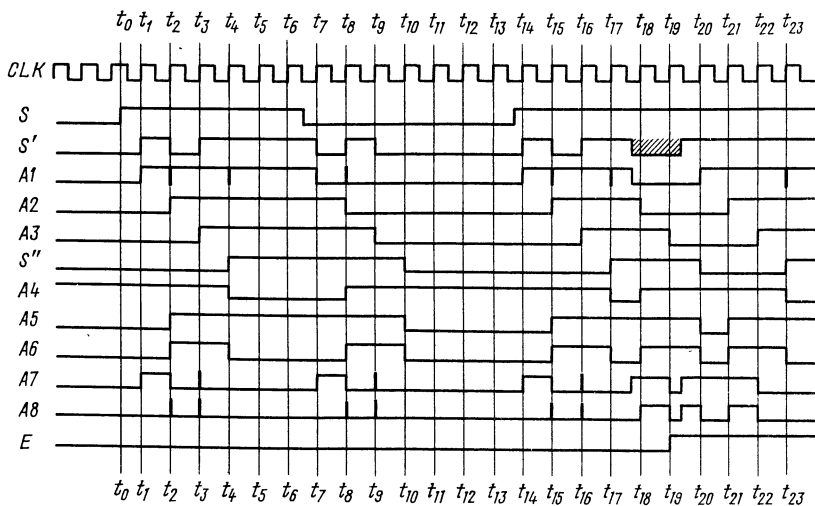


Рис. 2.12. Временные диаграммы сигналов в системе передачи информации при нормальной работе и при появлении в линии связи импульса ошибки (заштрихован)

Далее процессы протекают аналогично. Переход сигнала  $S$  из 1 в 0 сопровождается формированием сигнала  $S'=0$  с положительным «выбросом» в период  $t_8-t_9$ , на выходе дешифратора в момент  $t_{10}$  появляется сигнал  $S''=0$ , очищенный от «выброса», сигнал обнаружения ошибки не формируется и т. д.

При появлении ошибки в сигнале  $S'$ , передаваемом по линии связи (ошибочный импульс на рис. 2.12 заштрихован), дешифратор регистрирует ее и в момент  $t_{19}$  формирует сигнал обнаружения ошибки  $E=1$ . Этот сигнал фиксируется в триггере D12 до момента его установки в нуль внешним источником, принявшим ошибку к сведению (этот источник и цепь установки нуля триггера D12 на рисунке не показаны). После окончания импульса ошибки сигнал  $S''$  восстанавливает правильное значение, и далее, начиная с момента  $t_{23}$ , схема формирует правильную информацию.

Способность системы обнаруживать ошибки в линии связи можно повысить, если фронт исходного сигнала  $S$  преобразовать не в 3, а в 5 фронтов сигнала  $S'$ , т. е. если сформировать два «лишних» импульса вместо одного (см. рис. 2.9). Однако при этом продолжает снижаться пропускная способность системы и усложняется аппаратура шифрации — дешифрации.

## 2.5. Передача тестовой информации для проверки составных частей микроЭВМ в обход внутренней магистрали через последовательный канал связи [23]

При тестировании микроЭВМ, когда она по тем или иным причинам не может сама себя проверить, необходимо получить доступ к ее отдельным блокам (ПЗУ, ОЗУ, контроллерам и др.) со стороны внешнего стенда. Для такого доступа обычно используют внутреннюю магистраль микроЭВМ. Процессор проверяемой микроЭВМ переводится в пассивное состояние или извлекается из контактирующей колодки, и магистраль предоставляется в распоряжение внешнего стенда. Такое решение, однако, неэкономично, если магистраль содержит много линий, например, 32 и 24 линии для передачи данных и адреса.

Чтобы упростить внешнюю проверочную аппаратуру и уменьшить число проводов в соединительном кабеле, предлагается передавать тестовую ин-

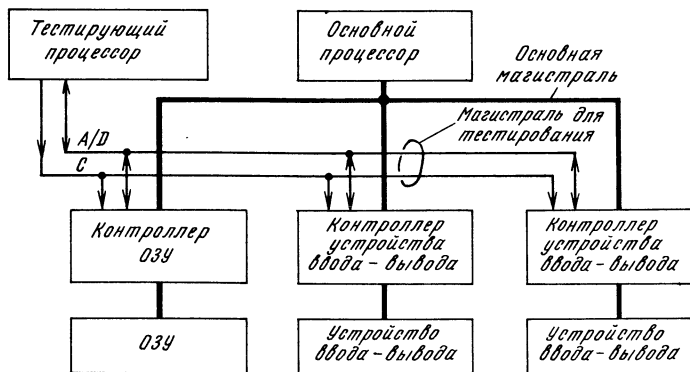


Рис. 2.13. Схема подключения тестирующего процессора к микроЭВМ. Для уменьшения аппаратных затрат тестирования вместо многоразрядной основной магистрали используется вспомогательная двухразрядная магистраль

формацию в обход основной магистрали (рис. 2.13). В качестве стенда используется тестирующий процессор, связанный с проверяемой микроЭВМ двумя сигнальными линиями и общей шиной нулевого потенциала (на рисунке не показана). По сигнальным линиям передаются адреса, данные (A/D) и синхриимпульсы (С).

Проверяемая микроЭВМ содержит основной процессор, ОЗУ, устройства ввода—вывода с соответствующими контроллерами и другие блоки (на рисунке не показаны). При нормальной работе микроЭВМ тестирующий процессор остановлен либо вообще не присоединен к магистрали. Основной процессор, как обычно, извлекает команды из памяти (ПЗУ, ОЗУ) и выполняет их, обращаясь к различным блокам, подключенным к магистрали, реагирует на сигналы прерываний и т. д.

В режиме тестирования или отладки управление передается тестирующему процессору, основной процессор не участвует в работе. Программы тестирующего процессора находятся в «стендовом» ОЗУ (ОЗУ на рисунке не показано). Возможна поочередная работа обоих процессоров, но этот вариант не рассматриваем, как не имеющий прямого отношения к существу предложенного решения.

Тестирующий процессор способен обмениваться информацией с адресуемыми элементами проверяемой микроЭВМ в режимах записи и чтения. Обмен представлен двумя фазами: в первой сигнал A/D отображает адрес ячейки памяти или регистра, во второй—данные, передаваемые из процессора (при записи) или в процессор (при чтении).

В контроллерах применены однотипные схемные решения. Для определенности рассмотрим контроллер ОЗУ (рис. 2.14). При нормальной работе микроЭВМ по основной магистрали передаются адрес, данные и сигналы управления. Адрес с шины 2 запоминается в регистре D8 и поступает в ОЗУ по шине 5. С помощью дешифратора D3 проверяется правильность обращения,

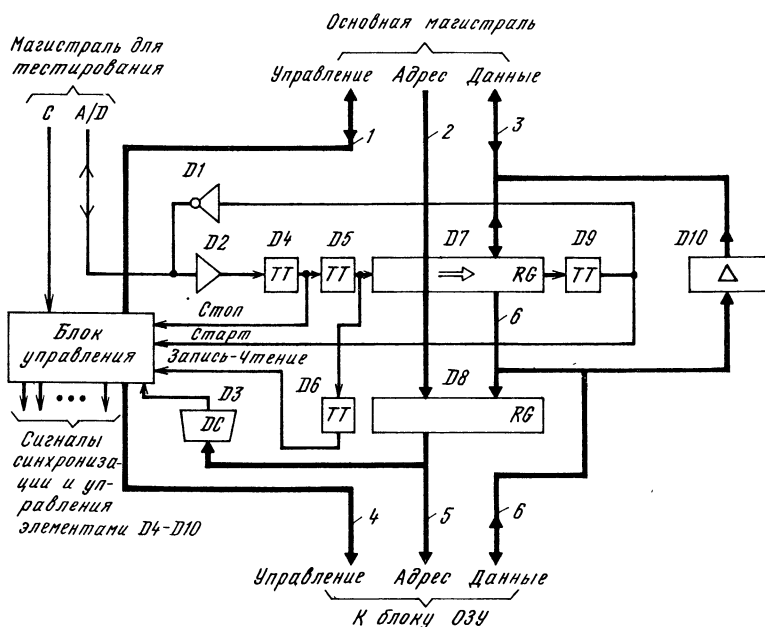


Рис. 2.14. Схема контроллера ОЗУ (рис. 2.13)

т. е. попадание адреса в область адресного пространства, выделенную для ОЗУ. При выходе адреса за заданный диапазон контроллер остается в пассивном состоянии. При правильной адресации блок управления формирует последовательность сигналов, зависящую от выполняемого действия (запись или чтение).

Если выполняется запись, то данные с шины 3 основной магистрали запоминаются в регистре D7 и поступают в ОЗУ по шине 6. Блок управления формирует в линиях шины 4 сигналы, в соответствии с которыми слово данных запоминается в ячейке памяти. Если выполняется чтение, то поступившее из ОЗУ по шине 6 слово проходит через усилители D10, выдается в шину 3 и попутно запоминается в регистре D7 (выдача информации из этого регистра в шину 6 в данном случае запрещена). После этого регистр D7 выдает слово в шину 3, подтверждая уже имеющуюся там информацию, а блок ОЗУ освобождается и переходит в пассивное состояние. Выдача слова из регистра D7 в шину 3 прекращается по завершении процессором операции чтения.

В режиме тестирования алгоритм работы контроллера ОЗУ иной (рис. 2.15). Напомним, что основной процессор выключен, а тестирующий включен. После воздействия сигнала общего сброса (от клавиши либо от основного процессора на заключительной стадии его работы) блок управления контроллера устанавливает в 0 триггеры D4, D5, D9 и регистр D7 (блок 100). Элементы D4, D5, D7 и D9 образуют единый сдвиговый регистр, в котором поступающая через неинвертирующий усилитель D2 информация сдвигается вправо при переходе сигнала С из 1 в 0. Выход инвертора D1 выполнен по схеме с открытым коллектором (нагрузочный резистор не показан, он установлен в тестирующем процессоре), поэтому до тех пор, пока триггер D9 находится в состоянии лог.0 (Старт=0), этот инвертор выключен и не препятствует поступлению информации в сдвиговый регистр.

Как отмечалось, в первой фазе обмена сигнал А/Д (последовательный код) отображает адрес ячейки, к которой предстоит обращение. Точнее, после выполнения ряда сдвигов этот адрес размещается в регистре D7. В то же время служебные биты Старт и Стоп, равные единице, попадают в триггеры D9 и D4, а признак, определяющий режим обращения (запись или чтение), заносится в триггер D5. Блок управления выявляет выполнение условия Старт & Стоп=1 (блок 101) и при переходе сигнала С в состояние 1 (блок 102) пересылает по шине 6 адрес из регистра D7 в регистр D8, а признак записи—чтения—из триггера D5 в триггер D6 (см. также временные диаграммы, приведенные на рис. 2.16).

Так как на входе инвертора D1 присутствует сигнал Старт=1, выходной транзистор этого инвертора насыщен. Подобная же ситуация наблюдается и в остальных контроллерах. Тестирующий процессор после выдачи на вход элемента D2 Стоп-бита переключается на прием сигнала А/Д и ожидает его перехода в состояние лог.1. Когда самый «медленный» контроллер устанавливает в 0 триггеры D4, D5, D9 и регистр D7 (блок 104), инверторы D1 всех контроллеров оказываются выключенными и сигнал А/Д переходит из 0 в 1. Этот факт регистрируется тестирующим процессором и рассматривается им как готовность системы контроллеров к переходу ко второй фазе обмена, связанной с передачей данных.

Вторая фаза начинается с анализа содержимого триггера D6 (блок 105). Если блок управления получил признак записи, то он ожидает заполнения сдвигового регистра D4—D5—D7—D9 данными, когда вновь выполнится условие Старт & Стоп=1 (блок 109), как и при передаче адреса. (После заполнения сдвигового регистра данными триггер D5 не содержит полезной информации.) Далее при переходе сигнала С из 0 в 1 (блок 110) и правильном обращении (блок 111) слово данных с шины 6 записывается в ячейку ОЗУ (блок 112), как и при обычной работе с основным процессором.

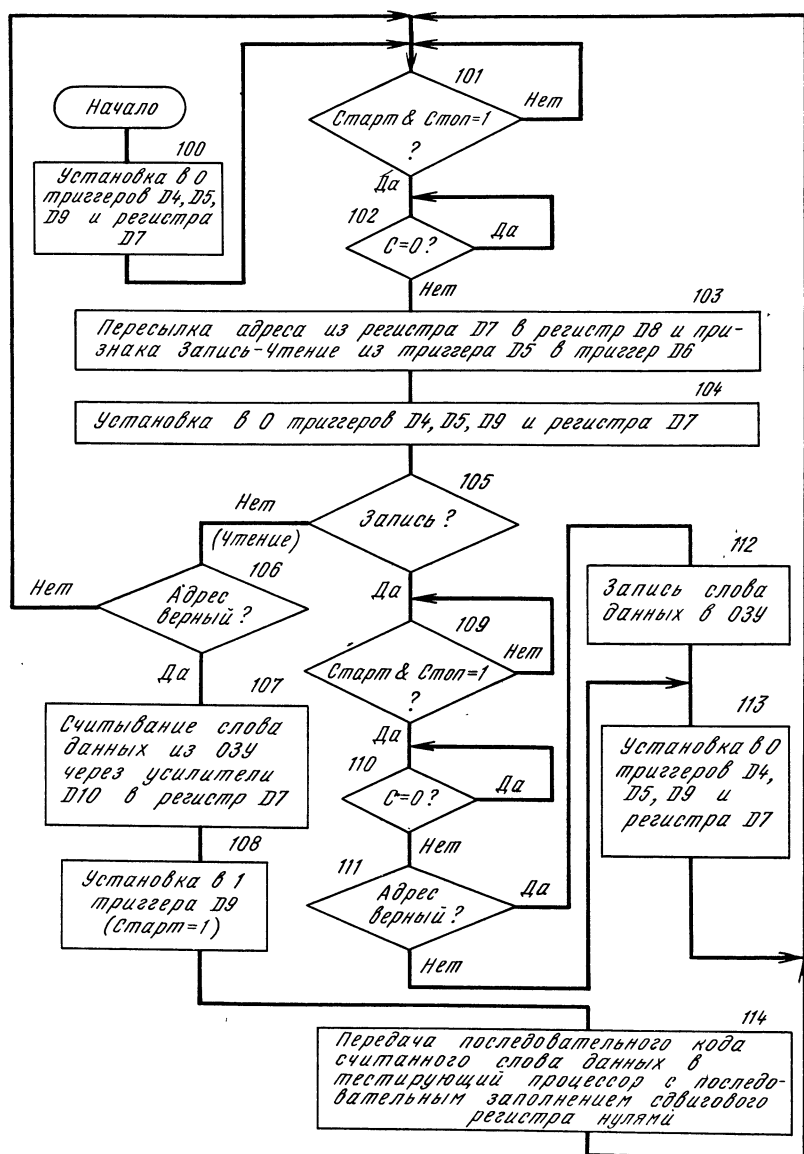


Рис. 2.15. Блок-схема алгоритма работы контроллера ОЗУ в режиме тестирования

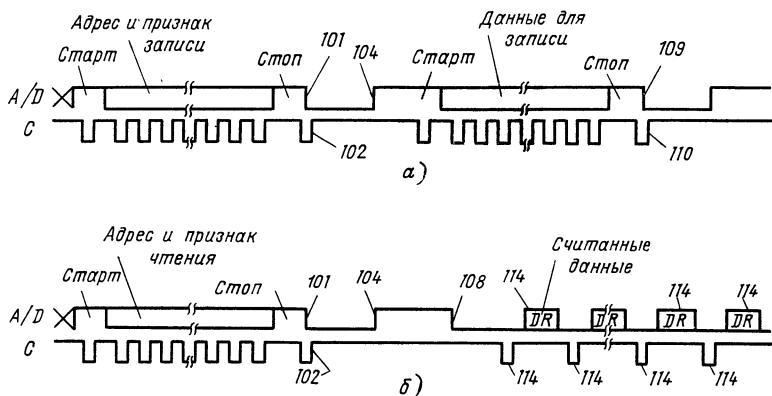


Рис. 2.16. Временные диаграммы сигналов A/D и C в магистрали для тестирования (рис. 2.13) при выполнении записи (а) и считывания (б) данных.

Числами обозначены выполняемые в данный момент алгоритмические блоки в соответствии с принятой на рис. 2.15 нумерацией

Тестирующий процессор после передачи на вход элемента D2 Стоп-бита, сопровождающего слово данных, переключается на прием сигнала A/D, ожидая его переход из 0 в 1. Контроллеры, не распознавшие свой адрес (блок 111), сразу же устанавливают в 0 триггеры D4, D5, D9 и регистр D7, их инверторы D1 выключаются. В данном случае самым «медленным» оказывается контроллер, распознавший свой адрес. Он выключает свой инвертор D1 только после выполнения записи кода по этому адресу. Тестирующий процессор принимает к сведению факт завершения операции (сигнал A/D перешел из 0 в 1) и формирует новое обращение, если это необходимо, а контроллеры готовы к очередному сеансу связи и вновь выполняют проверку условия Старт & Стоп=1 (блок 101).

Возвращаясь к моменту проверки сигнала Запись—Чтение (блок 105), рассмотрим процесс считывания информации из ОЗУ и ее передачи в тестирующий процессор. Если полученный адрес не имеет отношения к данному контроллеру (блок 106), то этот контроллер переходит к ожиданию поступления нового адреса (блок 101). Если адрес попал в заданный диапазон, то считанное из ОЗУ по шине 6 слово данных передается через усилители D10 и с шины 3 принимается в регистр D7 (блок 107).

Тестирующий процессор в это время проверяет состояние сигнала A/D. Так как во всех контроллерах сигнал Старт=0, их инверторы D1 выключены, поэтому A/D=1. После считывания слова данных из ОЗУ в регистр D7 блок управления устанавливает в 1 триггер D9 (блок 108). При этом срабатывает инвертор D1, тестирующий процессор обнаруживает переход сигнала A/D из 1 в 0 и расценивает этот факт как готовность данных, размещенных в регистре D7, к считыванию по последовательному каналу связи.

Далее (блок 114) выполняется серия сдвигов слова данных в цепи D4—D5—D7—D9, в результате это слово в последовательном коде передается в тестирующий процессор. По мере вытеснения данных из сдвигового регистра освободившиеся его разряды заполняются нулями. Для этого тестирующий процессор к моментам перевода сигнала C из 1 в 0 формирует сигналы A/D=0, а перед считыванием очередного разряда DR слова прекращает выдачу этого сигнала, освобождая магистраль, и переключается на прием. В результате по окончании считывания слова данных элементы D4, D5, D7 и D9 оказываются в состоянии лог. 0,

контроллер возвращается в исходное состояние и ожидает следующего сеанса связи (блок 101).

Если адрес не опознан ни одним контроллером, то при чтении тестирующий процессор в течение некоторого предельно допустимого интервала времени не получает сигнал готовности данных (который сформировался бы блоком 108 при правильном обращении), а при записи, напротив, подтверждение окончания операции приходит слишком рано (блок 112 обойден, управление от блока 111 передается непосредственно блоку 113). Поэтому обе ошибочные ситуации распознаются тестирующим процессором программно.

## 2.6. Схема контроля частоты сигналов в различных точках цифрового устройства [24]

При правильной работе цифрового устройства (рис. 2.17) в некоторых его контрольных точках должны присутствовать импульсные сигналы Н1—Н8, минимальная частота которых известна и составляет, например, 80 Гц. Такими точками в микроЭВМ могут служить выходы регистров, контакты разъемов магистралей, портов и т. п. Для слежения за сигналами в заданных точках предлагается использовать схему контроля, которая последовательно анализирует их. Если по крайней мере один из проверяемых сигналов не изменяется в течение максимально допустимого интервала времени, то это означает, что программа выполняется неправильно.

В схеме контроля (рис. 2.18) использован генератор G непрерывной последовательности импульсов частотой 100 Гц, которые поступают на двоичный счетчик СТ. Сигналы F1—F5 с выходов счетчика (рис. 2.19) участвуют в формировании импульсов CL и S, а также управляют мультиплексором MS, который последовательно передает входные сигналы Н1—Н8 на выход M для последующего анализа.

В исходном состоянии после воздействия сигнала Сброс счетчик СТ и регистр RG установлены в 0, индикаторы (светодиодные или иные) выключены. Так как F1=F2=0, S=1, триггер ТТ поддерживается в состоянии Q=1, мультиплексор MS передает на выход M сигнал Н1 (M=Н1).

При поступлении первого импульса от генератора G счетчик переходит в состояние 00001 (младший разряд F1 справа), сигнал S=1 снимается, триггер ТТ переходит к ожиданию первого положительного фронта сигнала M, с поступлением которого он переходит в состояние Q=0, так как на его D-входе постоянно присутствует сигнал лог. 0. Период ожидания фронта сигнала M соответствует состояниям 00001 и 00010 счетчика. Когда в счетчике накапливается код 00011, то формируется сигнал CL=1, его положительный фронт вызывает прием кода в регистр RG.

При правильной работе устройства к моменту формирования положительного фронта сигнала CL триггер ТТ установлен в 0, поэтому признак ошибки не вырабатывается. В данном примере обнаружено отсутствие изменения сигнала M=Н6 в течение максимально допустимого времени ожидания (точнее, отсутствие положительного фронта этого сигнала), составляющего 20 мс. В этой ситуации формируется сигнал Ошибка, элемент И «навсегда» закрывается по верхнему входу, чтобы не потерялась зафиксированная в регистре RG информация. Индикаторы отображают признак ошибки и код F'3—F'5, равный в данном случае 101 и соответствующий неправильноному сигналу Н6.

Схему можно усовершенствовать, чтобы сигнал ошибки вызывал прерывание процессора, его перезапуск или иные действия. Можно применить несколько подобных схем, охватывающих разные группы контрольных точек и настроенных на разные периоды ожидания смены значений проверяемых сигналов.

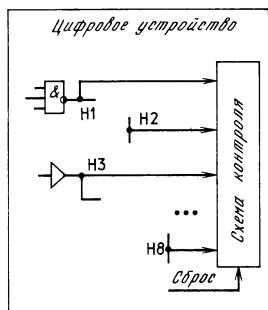


Рис. 2.17. Цифровое устройство со встроенной схемой контроля динамики сигналов Н1—Н8 в выбранных точках

Рис. 2.18. Структура схемы контроля.  
При обнаружении «плохой» динамики сигналов Н1—Н8 формируется признак Ошибка и сопровождающий его код номера канала ( $F'3—F'5$ )

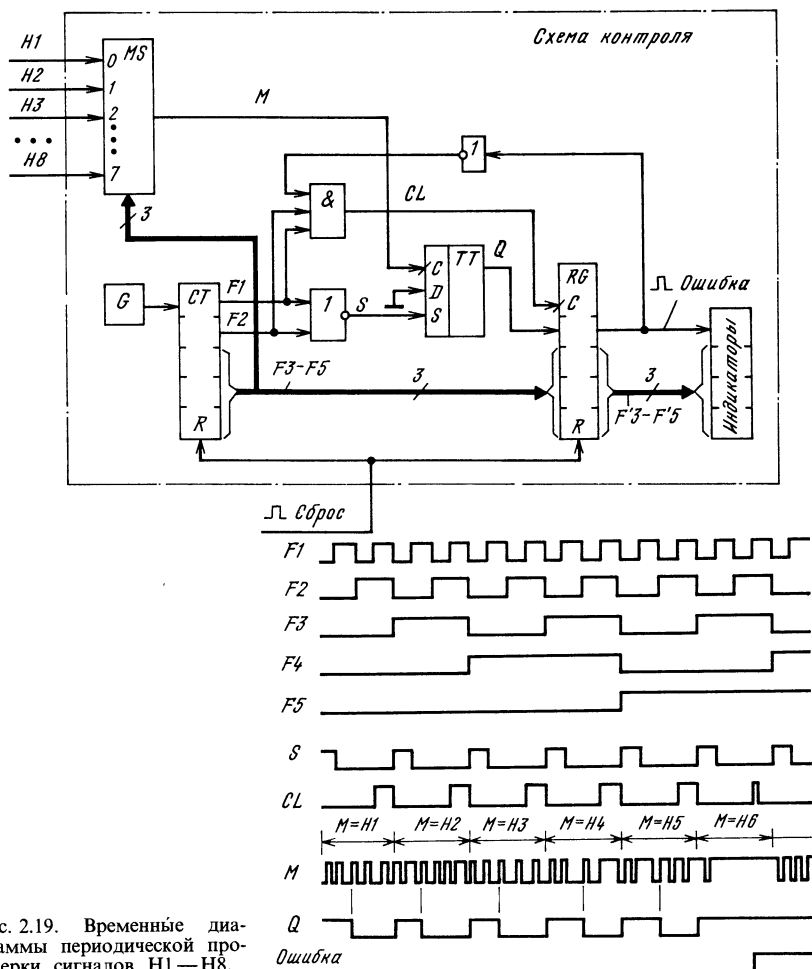


Рис. 2.19. Временные диаграммы периодической проверки сигналов Н1—Н8. Обнаружена «плохая» динамика сигнала Н6

## 2.7. Ускоренное формирование контрольных разрядов четности результата суммирования двух чисел [25]

В схеме рис. 2.20 16-разрядные числа  $A$  и  $B$  складываются сумматором  $ADD$  и одновременно с этим обрабатываются блоком  $F$  формирования контрольных разрядов четности результата  $S$ . Обозначения  $AL$ ,  $BL$ ,  $SL$  и  $AH$ ,  $BH$ ,  $SH$  соответствуют младшим и старшим байтам операндов  $A$ ,  $B$  и суммы  $S$ ;  $PL$  и  $PH$  — контрольные разряды четности байтов  $SL$  и  $SH$ . Сумматор обычно имеет дополнительные выходы состояния: Нулевой результат, Отрицательный результат, Переполнение, Перенос из знакового разряда и др., однако они не представляют для нас интереса. При правильной работе сумматора и блока  $F$  контрольный разряд результата дополняет число единиц в соответствующем байте до четного.

Блок  $F$  (рис. 2.21) содержит два 8-разрядных сумматора  $ADD$ , два 8-входовых и один двухвходовый сумматоры по модулю два ( $M2$ ), элемент  $И$  и логический блок  $L$ . Контрольный разряд  $PL$  младшего байта результата формируется сложением по модулю два единичных битов в коде суммы младших байтов чисел  $A$  и  $B$ ; при наличии переноса из младшего байта формируется сигнал  $C=1$ , однако он не поступает в сумматор старших байтов.

Если  $C=0$ , то элемент  $И$  закрыт, сигнал  $PH=PH'$  формируется в результате суммирования по модулю два единичных битов в коде  $E$  на выходах сумматора старших байтов чисел  $A$  и  $B$ . Иными словами, в отсутствие переноса из младшего байта контрольный разряд  $PH$  формируется «автономно».

Перенос  $C=1$  из младшего байта в старший может изменить или оставить без изменения значение контрольного разряда  $PH$  в зависимости от кода  $E$ . Например, при  $E=00000000$   $PH'=0$ , но если  $C=1$ , то фактический код в старшем байте  $SH$  суммы (см. рис. 2.20) равен  $00000001$  и его контрольный разряд должен быть равен 1. Поэтому в данной ситуации необходимо откорректировать сигнал  $PH'=0$  (путем его инвертирования), а для этого нужно, чтобы на выходе логического блока  $L$  присутствовал сигнал  $Y=1$ .

Другой пример:  $E=00000111$ ,  $PH'=1$ ,  $C=1$ . В этом случае фактический результат сложения  $SH=00001000$  содержит нечетное число единиц, как и код  $E$ , т. е. коррекции сигнала  $PH'=1$  не требуется ( $PH'=PH$ ). Поэтому на выходе блока  $L$  должен присутствовать сигнал  $Y=0$ .

Из этих примеров следует, что логический блок  $L$  введен для выработки признака коррекции сигнала  $PH'$  при его преобразовании в сигнал  $PH$ , когда это необходимо. Работа блока  $L$  описывается табл. 2.2.

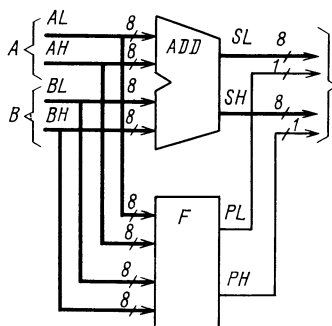


Рис. 2.20. Схема суммирования чисел  $A$  и  $B$  с формированием контрольных разрядов  $PL$  и  $PH$  четности младшего  $SL$  и старшего  $SH$  байтов суммы

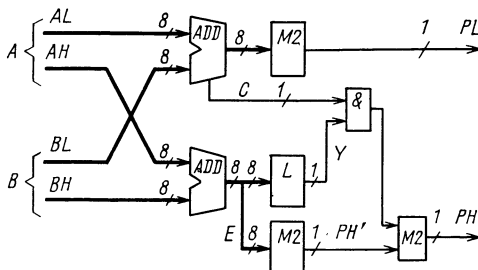


Рис. 2.21. Структура блока  $F$

Таблица 2.2

Код E	Код SH	Y
× × × × × × × 0	× × × × × × × 1	1
× × × × × × 0 1	× × × × × × 1 0	0
× × × × × 0 1 1	× × × × × 1 0 0	1
× × × × 0 1 1 1	× × × × 1 0 0 0	0
× × × 0 1 1 1 1	× × × 1 0 0 0 0	1
× × 0 1 1 1 1 1	× × 1 0 0 0 0 0	0
× 0 1 1 1 1 1 1	× 1 0 0 0 0 0 0	1
0 1 1 1 1 1 1 1	1 0 0 0 0 0 0 0	0
1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	0

Примечание × — произвольное значение (0 или 1).

Данное решение позволяет вычислять предполагаемое значение РН' контрольного разряда четности старшего байта, не дожидаясь окончания распространения сигнала переноса через полный 16-разрядный сумматор (поэтому в блоке F 16-разрядный сумматор заменен двумя независимыми 8-разрядными). В необходимых случаях при появлении сигнала C=1 значение контрольного разряда корректируется. Это приводит к меньшим потерям времени, так как не нужно начинать вычисления «с нуля».

## 2.8. Дублированное ОЗУ с коррекцией одиночных ошибок [26]

Дублированное ОЗУ (рис. 2.22) содержит два одинаковых блока памяти (1 и 2), объединенных системной и внутренней магистралями. В каждом блоке памяти имеется накопитель, выполненный на микросхемах динамических ОЗУ (например, типа K565PY7) с соответствующим блоком управления (F11, F21). Блок управления формирует 9-разрядный мультиплексированный адрес MA, сигналы выбора строки и столбца (RAS и CAS), а также признак записи WR. (Описания этих сигналов имеются в справочниках по микросхемам динамической памяти, поэтому здесь не приводятся.)

В каждой ячейке накопителя хранятся байт данных и соответствующий ему контрольный разряд, дополняющий общее число единиц в 9-разрядном коде до нечетного. Контрольный разряд P формируется блоком F12 (F22) при записи данных в накопитель и в дальнейшем считывается из накопителя одновременно с данными. Если нарушено условие нечетного числа единичных битов в считанной группе (9 разрядов), то формируется сигнал ошибки ERR1=0 (ERR2=0). Направление передачи данных через блоки F12 и F22 задается сигналами DIR.

Блоки памяти 1 и 2 работают синхронно под управлением сигналов CL, принимаемых с одной из линий системной магистрали. Регенерация данных в накопителях 1 и 2 также проводится синхронно благодаря объединению блоков F11 и F21 линией передачи сигнала взаимной синхронизации REF. Интерфейсные блоки F13 и F23 обеспечивают обмен сигналами между блоками памяти 1 и 2, а также между одним из этих блоков и активным устройством — процессором, каналом ПДП. Сигналы START начала операции вырабатываются блоками F13 и F23 одновременно. С помощью ключей SA1 и SA2 один из блоков памяти (в данном примере — блок 1) назначается основным, другой — дублирующим.

Данное решение гарантирует правильную работу ОЗУ при наличии одиночных ошибок в накопителях 1 и 2. Точнее, по каждому адресу

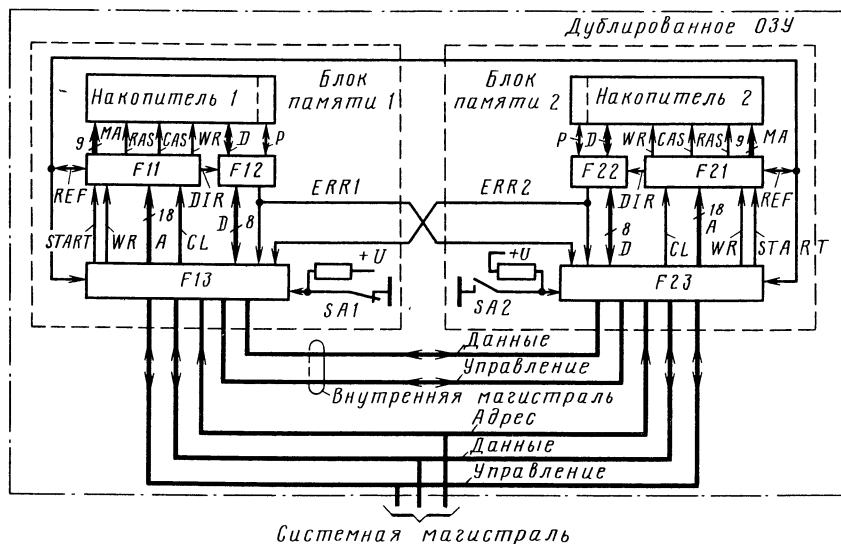


Рис. 2.22. Структура дублированного ОЗУ

допустима одиночная ошибка в одном из накопителей; правильные данные из исправной ячейки поступают в системную магистраль и одновременно с этим переписываются в ошибочную ячейку соседнего накопителя. Попытка перезаписи (восстановления) данных имеет смысл в том случае, когда ошибка не связана с отказом элементов БИС, а вызвана сбоем вследствие пролета  $\alpha$ -частицы через запоминающий элемент, действия импульса помехи по цепи питания и т. п.

В отсутствие ошибок (при  $ERR1=ERR2=1$ ) первый блок памяти — основной, второй — вспомогательный. Записываемые данные из системной магистрали поступают в одноименные ячейки обоих накопителей и дополняются контрольными разрядами  $P$ . Считанные из накопителей данные проходят через блоки  $F12$ ,  $F22$  и проверяются на соответствие контрольным разрядам  $P$ . После успешной проверки данные передаются блоком  $F13$  в системную магистраль; при этом блок  $F23$  отключен от шины данных системной магистрали (как дублирующий). Таким образом, данные записываются одновременно в оба накопителя, а считываются в системную магистраль только из блока памяти 1.

При обнаружении ошибки в накопителе 1 сигнал  $ERR1=0$  запрещает выдачу в системную магистраль считанных данных из блока памяти 1 и разрешает их выдачу из блока памяти 2. Одновременно с этим по внутренней магистрали из блока  $F23$  в блок  $F13$  передаются правильные данные и соответствующие сигналы управления. Блок  $F13$  принимает сигналы с внутренней магистрали и приступает к перезаписи данных по адресу последнего обращения. Благодаря введению в устройство внутренней магистрали процесс перезаписи идет «скрыто», в то время как системная магистраль уже освободилась. Однако прием новой команды записи или чтения возможен только после окончания этого процесса.

При обнаружении ошибки в накопителе 2 события развиваются аналогично, но блоки памяти 1 и 2 меняются ролями — данные передаются в системную магистраль из блока 1, как и в отсутствие ошибок, а затем делается попытка коррекции ошибки в накопителе 2.

## 2.9. Повышение корректирующих возможностей системы исправления ошибок, использующей код Хэмминга [27]

Для исправления одиночных ошибок в информационных словах, хранимых в ячейках динамического ОЗУ, обычно используют код Хэмминга. В схеме, показанной на рис. 2.23, в дополнение к обычному корректору по Хэммингу применен буфер—набор двунаправленных ретрансляторов. В зависимости от значений управляющих сигналов Направление передачи и Инвертирование данных (блок управления ОЗУ не показан) буфер передает информацию сверху вниз или снизу вверх с инвертированием всех разрядов или без инвертирования. В исходном состоянии информация буфером не инвертируется, ОЗУ работает в соответствии с общепринятым алгоритмом исправления одиночных и обнаружения двойных (и большей кратности) ошибок.

Напомним кратко, как работает обычное ОЗУ с коррекцией информации по Хэммингу. При записи 64-разрядного слова в ОЗУ корректор транслирует его из магистрали в буфер, который, в свою очередь, передает его в неизменном виде в накопитель. Одновременно с этим корректор формирует 8-разрядный код Хэмминга, который также передается в накопитель. В результате в адресуемой 72-разрядной ячейке ОЗУ запоминается полезная информация (64 бита) и вспомогательная (8 бит). При считывании слова из ячейки памяти оно в неизменном виде проходит через буфер и поступает в корректор; туда же передается код Хэмминга, считанный из той же ячейки.

Корректор анализирует полученную информацию. Если ошибок нет, то поступившее из накопителя 64-разрядное слово без изменений передается в магистраль. Если имеется одиночная ошибка в одном из 64 считанных из накопителя разрядов, то эта ошибка исправляется «на проходе» (но не в самом накопителе) и восстановленное слово передается в магистраль. Одновременно с этим формируется сигнал *Корректируемая ошибка*, который может вос-

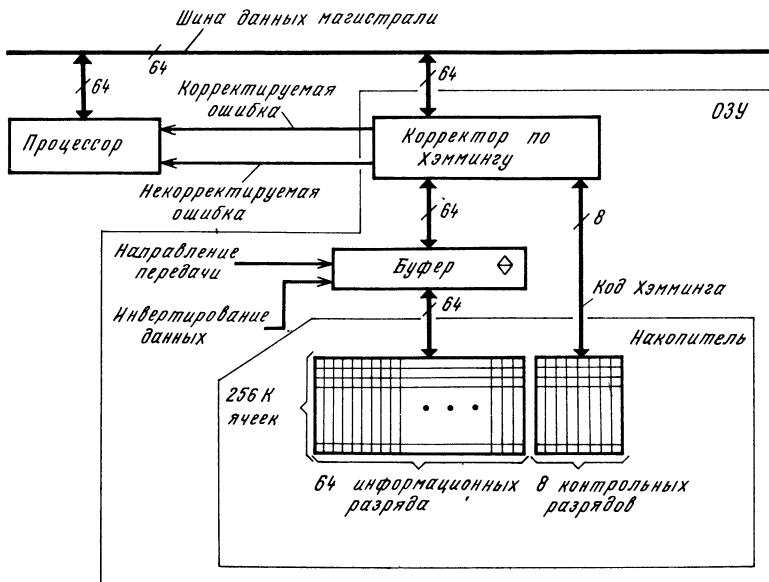


Рис. 2.23. ЭВМ с памятью, защищенной от ошибок.

Данные хранятся в накопителе в прямом или обратном коде, контрольные разряды—только в прямом

приниматься или игнорироваться процессором в зависимости от особенностей построения конкретной ЭВМ. Допустима также одиночная ошибка в считанных контрольных разрядах, если нет ошибок в информационных. Если в считанном 72-разрядном слове две или более ошибок, то оно не исправляется, в процессор посылается сигнал Некорректируемая ошибка. Отметим, что корректор способен определять положение одиночной ошибки—если она в информационных разрядах, то ее нет в контрольных, и наоборот.

В микросхемах динамической памяти обычно наблюдаются ошибки двух видов—постоянные и типа «сбой». Постоянные ошибки характеризуются тем, что содержимое одной, нескольких или всех адресуемых ячеек БИС заранее определено—из них всегда считывается одна и та же информация. Например, из ячеек с адресами, кратными некоторому числу, может считываться лог. 1, и попытка записи лог. 0 в эти ячейки не дает желаемых результатов. Ошибки типа «сбой» часто связывают с пролетом  $\alpha$ -частиц через кристалл БИС, что вызывает разряд конденсаторов динамических элементов памяти. Это, конечно, не единственно возможная причина сбоев—они могут быть, например, следствием действия кратковременных импульсов случайных помех по цепям питания или линиям управления и т. п. Ошибку типа «сбой» можно устранить повторной записью в ту же ячейку считанного и откорректированного кода.

Накопитель (рис. 2.23) можно построить из 72 БИС K568РУ7 со структурой  $256K \times 1$  бит. Каждая БИС соответствует одному столбцу матрицы накопителя. При постоянном отказе БИС информация искажается в целом столбце матрицы или его части. Благодаря коррекции по Хэммингу искаженные слова восстанавливаются, однако при появлении новых ошибок (вероятнее всего, типа «сбой») ОЗУ теряет работоспособность, так как двойные ошибки не исправляются.

Задача заключается в том, чтобы значительно уменьшить вероятность отказа ОЗУ при появлении одиночных сбоев в ячейках, уже имеющих постоянные одиночные ошибки. Эта задача в данном случае решается при условии, что информация после ее загрузки в накопитель не меняется или, во всяком случае, меняется редко. Идея состоит в следующем. Постоянная ошибка в одном из 64 информационных разрядов ячейки может «прятаться», совпадая с полезной информацией. Так, ошибка типа постоянной 0 не мешает хранению «настоящего» нуля, и наоборот. Чтобы «подстроиться» под имеющуюся ошибку, информационное слово при необходимости инвертируют, так что хранимый бит не противоречит ошибке.

Этот известный прием («маскировки» постоянных ошибок использовался и в ранее разработанных схемах, однако для того, чтобы система коррекции знала, представлено ли хранимое информационное слово в прямом или обратном коде, предусматривался дополнительный разряд накопителя. Этот разряд, в свою очередь, должен быть защищен от одиночных ошибок, например, мажоритарным резервированием. В предлагаемом решении (рис. 2.23) нет необходимости введения дополнительного разряда—в этом и заключается его преимущество. Рассмотрим работу ЭВМ подробнее.

После загрузки массива информации в накопитель производится «маскировка» постоянных одиночных ошибок, по крайней мере—подавляющей их части. Этот процесс протекает с участием или без участия процессора, в зависимости от принятых архитектурных и схемных решений конкретной ЭВМ. При маскировке постоянных ошибок выполняются следующие действия (рис. 2.24).

1. Считывается содержимое очередной ячейки ОЗУ (блок 1).

2. Если ошибка корректором не зарегистрирована (блок 2), то проверяется условие окончания обработки массива (блок 11). При обнаружении ошибки в зависимости от того, корректируемая она или нет (блок 3), производится переход к блоку 5 или 4, в котором предусмотрена выдача человеку—оператору сообщения о невозможности работы в связи с отказом ОЗУ.

3. Выполняется эксперимент по определению типа ошибки (блок 5). Как отмечалось, ошибку типа «сбой» удастся исправить перезаписью откорректированного слова. Если при последующем повторном считывании ошибка

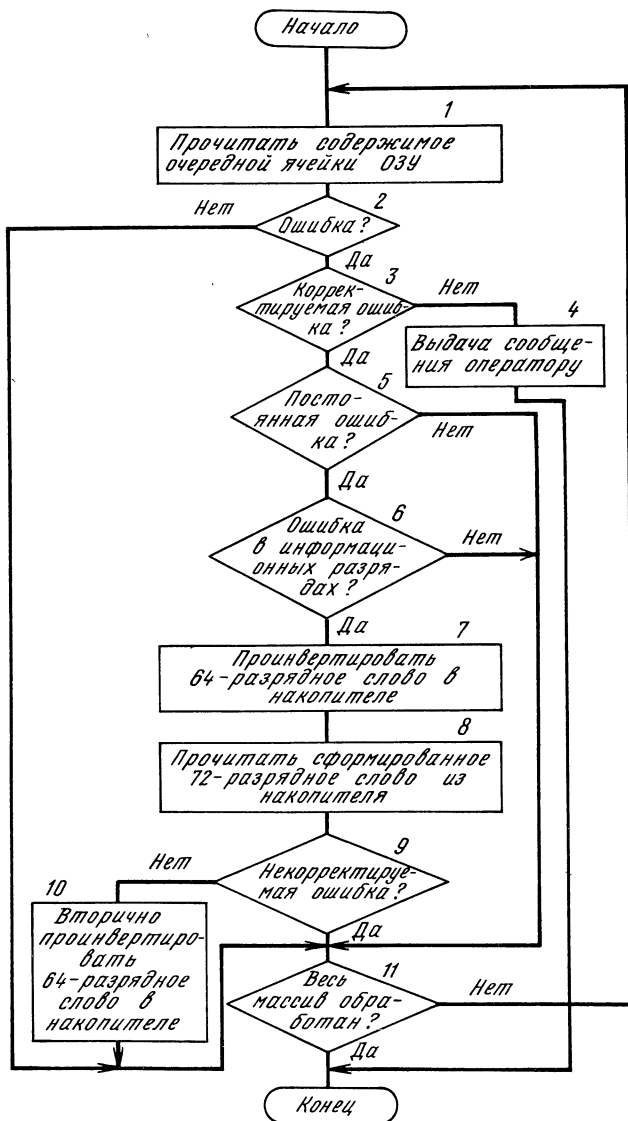


Рис. 2.24. Блок-схема алгоритма обработки массива данных с целью сокрытия постоянных одиночных ошибок

не проявляется, то сбой устранен. Если ошибка по-прежнему проявляется, то она расценивается как постоянная. В зависимости от результата эксперимента осуществляется переход к блоку 6 или 11.

4. Проверяется положение постоянной одиночной ошибки в 72-разрядном слове (блок 6). Ошибка в контрольных разрядах не обрабатывается (осуществляется переход к блоку 11). Если ошибка присутствует в информационной части слова (64 разряда), то далее (блок 7) производится попытка ее «маскировки» инвертированием. Разряды кода Хэмминга при этом не инвертируются.

5. Инвертированное 64-разрядное слово данных и неинвертированный код Хэмминга поступают из накопителя в корректор (блок 8). В подавляющем большинстве кодовых ситуаций корректор регистрирует некорректируемую ошибку — а это и есть своеобразная метка, сопровождающая упорядоченную инвертированием ошибку. Поэтому при обнаружении некорректируемой ошибки слово считается успешно обработанным, производится переход к блоку 11. Если ожидаемой (запланированной) некорректируемой ошибки нет, то осуществляется переход к блоку 10 для восстановления первоначальной формы представления данных (прямой код данных совместно с прямым кодом Хэмминга), так как попытка упрятывания привела к ложным, но «безопасным» или «корректируемым» данным.

6. Проверяется условие окончания обработки массива (блок 11), и при выполнении этого условия процедура завершается. Таким образом, информация в основном (а возможно, и полностью) «подогнана» под имеющиеся постоянные ошибки. Эта информация готова для последующего использования.

Считывание информации поясняется рис. 2.25. Из накопителя извлекается 72-разрядное слово (блок 1) и в неизменном виде поступает в корректор. Если ошибок нет, то процедура завершается. При обнаружении

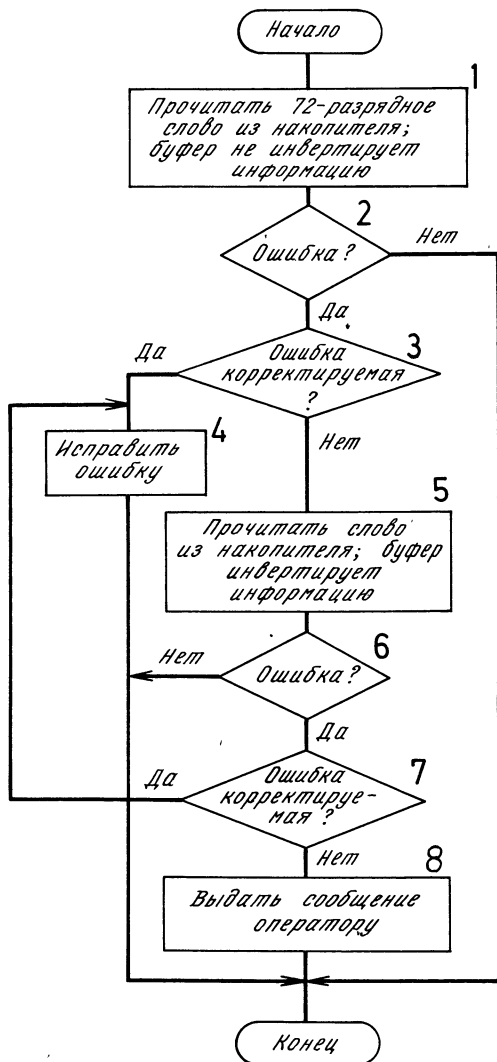


Рис. 2.25. Блок-схема алгоритма считывания слова

корректируемой ошибки она исправляется (блоки 2—4). При обнаружении некорректируемой ошибки 64-разрядное слово данных вновь считывается из накопителя, но с инвертированием (блок 5) и вновь проверяется на отсутствие ошибок (блок 6). Если ошибок нет, то слово передается в процессор. Если ошибка корректируемая (блок 7), то она исправляется (блок 4), если ошибка некорректируемая, то ОЗУ признаётся неисправным (блок 8).

Таким образом, без излишних затрат аппаратуры удастся скрыть большинство постоянных одиночных ошибок и на их фоне исправлять новые ошибки (типа «сбой»), пользуясь «неизрасходованными» корректирующими возможностями кода Хэмминга. Отметим, что при конкретной реализации предлагаемой системы с выбранной модификацией кода Хэмминга должно быть теоретически или с помощью машинного моделирования подтверждено выполнение следующего требования. Инвертированный код данных, удовлетворяющий условию, определяемому блоком 9 рис. 2.24, при внесении в любой его разряд одиночной ошибки и сопоставлении с неинвертированным кодом Хэмминга должен вызывать формирование признака некорректируемой ошибки. В противном случае инвертированное слово данных с упятой в одном из разрядов постоянной ошибкой и ошибкой типа «сбой» в одном из оставшихся разрядов при считывании будет расценено как обычное неинвертированное слово (безошибочное или содержащее корректируемую ошибку). То же относится и к внесению любой одиночной ошибки («сбоя») в код Хэмминга, сопровождающий инвертированные данные.

## **2.10. Контроль ошибки динамического ОЗУ, совмещенный с выполнением циклов его регенерации [28]**

Для поддержания целостности хранимой в динамическом ОЗУ информации необходима периодическая ее регенерация. Чтобы в какой-то мере оправдать бесполезные (с точки зрения пользователя) затраты времени на регенерацию, предлагается попутно вести чтение содержимого ячеек ОЗУ (или его части) с контролем считанной информации. Ячейки перебираются последовательно, циклически, в полном диапазоне адресов, занимаемых ОЗУ, или в части этого диапазона; при этом, однако, регенерируются все ячейки ОЗУ. При обнаружении ошибки производится попытка ее коррекции путем перезагрузки правильной информации с внешнего устройства в соответствующую ячейку или область ОЗУ.

В примере, приведенном на рис. 2.26, ОЗУ содержит 256К 16-разрядных ячеек и выполнено на 16 БИС, каждая из которых имеет структуру  $256\text{К} \times 1$  бит. Регенерация информации в накопительной матрице БИС (512 строк, 512 столбцов) должна осуществляться путем обращения к каждой из 512 строк не реже чем через 8 мс. Это требование выполняется при сканировании ОЗУ, т. е. при считывании информации из ячеек, лежащих в последовательных строках матрицы БИС, с частотой 64 кГц (период равен 15,625 мкс). Цикл перебора всех 256К ячеек ОЗУ составляет  $256 \times 1024 / 64000 = 4,096$  с, однако можно сканировать не всю область ОЗУ, а только ту часть, в которой хранится полезная (для выполнения данного задания) информация. Минимальная сканируемая область составляет 512 последовательных ячеек, отображаемых на разные строки накопительной матрицы БИС, так что регенерируются все 256К ячеек, а проверяется только полезная информация.

В микроЭВМ выполнение программы процессором совмещено (чередуются) во времени со сканированием последовательных ячеек памяти (или ее части) со стороны блока управления каналом прямого доступа в память. Собственно канал прямого доступа в память (соответствующий данному блоку управления) отсутствует, поток считываемой при сканировании ОЗУ информации адресуется несуществующему внешнему устройству и нужен лишь для контроля правильности считанных кодов. Контроль осуществляется проверкой четности числа единиц в байтах 16-разрядного слова. К каждому

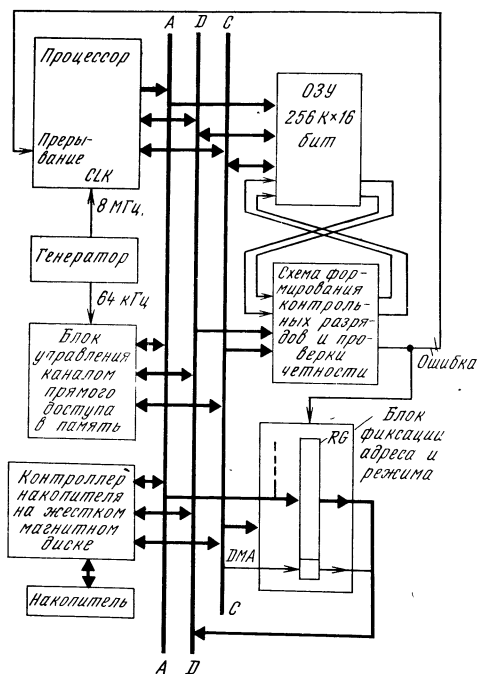


Рис. 2.26. МикроЭВМ с контролем динамического ОЗУ.

Регенерация сопровождается проверкой четности числа единиц в последовательных ячейках памяти, так что ошибка в слове может быть найдена и исправлена еще до того, как процессор этим словом воспользуется

байту добавлен контрольный разряд, дополняющий число единиц до четного или нечетного в зависимости от принятого схемного решения. Схема формирования контрольных разрядов и проверки четности передает в ОЗУ два контрольных разряда при записи слова, а при его считывании из ОЗУ вырабатывает признак ошибки, если нарушено условие четности (нечетности) числа единиц хотя бы в одном байте.

Ошибки в ОЗУ разделяют на постоянные и типа «сбой». Последние обычно обусловлены помехами по цепям питания, линиям управления, пролетом  $\alpha$ -частиц через кристалл БИС и другими факторами. Эти ошибки могут быть устранены (исправлены) записью правильной информации, если таковая имеется. Постоянные ошибки исправить таким путем не удастся. В предлагаемом решении правильная информация (программы, неизменяемые данные и пр.) хранится в накопителе на жестком магнитном диске. При обнаружении ошибки производится перегрузка правильной информации с диска в ОЗУ на место неправильной.

Если информация исправлена заблаговременно, до момента ее использования процессором, то проблем не возникает, процессор продолжает нормальное выполнение программы. Если же ошибка обнаружена при обращении процессора к ОЗУ, то ситуация усложняется. Чтобы скомпенсировать ошибку, процессор должен проанализировать сферу ее влияния, вернуться к некоторой опорной точке программы, перезагрузить с накопителя на жестком магнитном диске в ОЗУ правильную информацию и повторить выполнение исправленного фрагмента программы.

Такие действия, однако, не всегда осуществимы из-за практической неразрешимости задачи анализа некоторых ошибочных ситуаций. В таких ситуациях единственным приемлемым решением является формирование сигнала общего сброса микроЭВМ с последующей перезагрузкой и перезапуском решаемой задачи с самого начала. Однако такой подход может оказаться неэффективным из-за потери результатов вычислений, накопленных в ОЗУ к моменту обнаружения ошибки.

При обнаружении ошибки на вход прерывания процессора подается сигнал, по которому процессор временно откладывает выполнение текущего задания, запоминает параметры возврата к нему и переходит на начало прерывающей программы реакции на ошибку. Прерывающая программа предусматривает считывание программно-доступного регистра RG и анализ его содержимого. В этом регистре в момент формирования сигнала Ошибка фиксируется адрес ошибочной ячейки ОЗУ и одноразрядный признак DMA, по которому можно определить, возникла ли ошибка при обращении к ОЗУ со стороны процессора или при сканирующей регенерации (прочие каналы прямого доступа в память для простоты не рассматриваем).

Если ошибка не связана с обращением процессора в ОЗУ, то он способен корректно продолжать работу. В данном случае процессор анализирует адрес ошибки, оценивает возможность ее исправления и, если такая возможность имеется, осуществляет загрузку правильной информации в ОЗУ через контроллер. Далее процессор возвращается к прерванному заданию. (Произошла опережающая коррекция, процессор заблаговременно «расчистил дорогу» для беспрепятственного продвижения по программе в будущем.)

Если ошибка связана с недавним обращением процессора в ОЗУ (признак DMA в регистре RG установлен в соответствующее положение), то, как отмечалось, необходим сложный анализ ситуации или, когда результаты анализа неопределенны,—перезапуск микроЭВМ «с нуля».

После включения напряжения питания в ОЗУ присутствует случайная информация, блок управления каналом прямого доступа в память начинает работу по сканирующей регенерации ОЗУ в соответствии с параметрами, полученными от процессора. Чтобы не реагировать на ложные сигналы ошибок в этот период, процессор маскирует (запрещает) прием сигналов прерывания до момента формирования информации с правильной четностью во всей сканируемой области. Блок фиксации адреса и режима хранит параметры ошибки до тех пор, пока эти параметры не будут прочитаны процессором, затем блок переходит к ожиданию новой ошибки.

## **2.11. Уменьшение энергии, потребляемой статическим ОЗУ с контролем по четности от резервной батареи в отсутствие штатного напряжения питания [29]**

Для сохранения информации при отключении штатного напряжения питания микроЭВМ (например, 220 В, 50 Гц) используют статические ОЗУ с резервным питанием от батареи или аккумулятора. В схеме, показанной на рис. 2.27, восемь информационных и один контрольный разряды хранятся соответственно в ОЗУ1 и ОЗУ2. Особенность этой схемы состоит в том, что в отсутствие штатного напряжения питания энергия от резервной батареи поступает только в ОЗУ1, а ОЗУ2 остается обесточенным и, следовательно, информация в нем теряется. Это, однако, не создает каких-либо трудностей при дальнейшей работе микроЭВМ, так как потерянную информацию можно восстановить, как показано далее.

Рассмотрим состояние ОЗУ1 и ОЗУ2 к моменту выключения штатного напряжения питания. Предполагается, что в ОЗУ1 хранятся некоторые коды А, В, С, и т. д., причем код S в ячейке с адресом 16K—1 этого ОЗУ

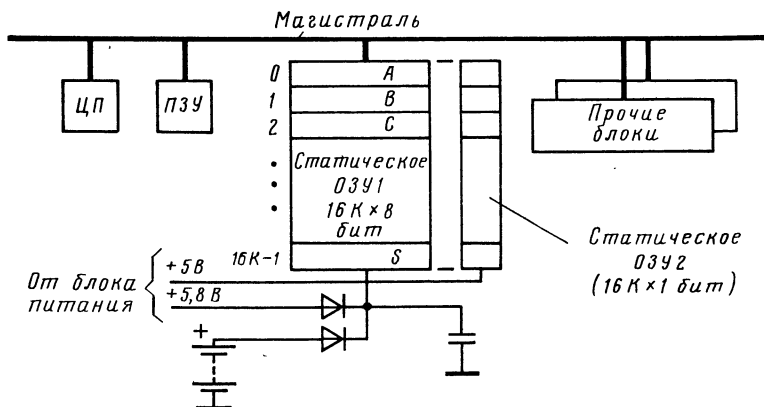


Рис. 2.27. МикроЭВМ с сохранением информации в ОЗУ1 в отсутствие штатного напряжения питания.

ОЗУ2 контрольных разрядов не потребляет энергию от резервной батареи, поэтому соответствующая информация теряется, но в дальнейшем она воспроизводится при перезаписи кодов в ОЗУ1

представляет собой контрольную сумму всех кодов, хранимых в ячейках с адресами 0—(16К—2). Контрольная сумма S образуется, например, арифметическим суммированием с учетом циклического переноса из старшего разряда в младший (возможны и иные способы получения контрольной суммы). В ОЗУ2 хранятся контрольные разряды, дополняющие число единиц в соответствующих ячейках ОЗУ1 до четного или нечетного (по договоренности).

При наличии штатного напряжения питания энергия от батареи не потребляется, так как нижний диод закрыт. После выключения штатного напряжения питания информация в ОЗУ1 сохраняется благодаря поступлению на это ОЗУ напряжения от батареи, при этом верхний диод закрыт. Информация в ОЗУ2, как отмечалось, теряется.

При последующем включении штатного напряжения питания центральный процессор приступает к выполнению программы начального пуска, хранимой в ПЗУ. В этой программе, в частности, предусмотрено вычисление новой контрольной суммы и сравнение ее со старой, хранимой в ячейке с адресом 16К—1. Если суммы совпали, то предполагается, что прошлая информация сохранена без искажений и может использоваться для дальнейшей работы. Если суммы не совпали, то оператору выдается сообщение об ошибке или выполняются иные действия, предписанные программой начального пуска.

При вычислении контрольной суммы коды, считываемые из ОЗУ1, сопровождаются случайной информацией, поступающей из соответствующих ячеек ОЗУ2, поэтому схема контроля четности (на рисунке эта схема не показана) регистрирует появление ошибочных ситуаций. Чтобы не реагировать на сигналы от схемы контроля, процессор оставляет закрытым (замаскированным) соответствующий вход прерывания до тех пор, пока в ОЗУ2 не будет сформирована правильная информация. (Исходно при включении штатного напряжения питания сигнал запрета прерывания устанавливается автоматически.)

Правильная информация формируется в ОЗУ2 при перезаписи кодов в ОЗУ1 следующим образом. После считывания очередного кода из ОЗУ1 в процессе формирования или проверки новой контрольной суммы этот код возвращается процессором в ту же ячейку ОЗУ1, из которой он был считан. При этом срабатывает схема формирования контрольного разряда (эта схема конструктивно совмещена со схемой проверки четности),

Для правильной работы микроЭВМ необходимо, чтобы к моменту выключения штатного питающего напряжения контрольная сумма S точно соответствовала массиву кодов в ячейках с адресами 0—(16K—1) ОЗУ1. Если информация в ОЗУ1 неизменна, то контрольная сумма подсчитывается сразу после загрузки этой информации в ОЗУ1 (или в процессе загрузки) и в неизменном виде хранится в последней ячейке этого ОЗУ. Если же информация в ОЗУ1 меняется, то необходимо оперативно (аппаратно или программно) корректировать значение S, чтобы выключение штатного напряжения питания не застало микроЭВМ «врасплох». Можно разбить хранимый массив кодов на несколько страниц, каждой из которых присваивается своя контрольная сумма. Это позволяет уменьшить объем работы в критических ситуациях, когда ожидается переключение с основного источника питания на резервный (см. [65], § 6.14). Число контрольных разрядов может быть увеличено, например, при использовании кода Хэминга, при этом микросхемы памяти для их хранения также не подключаются к резервной батарее.

Принятые данные проходят через приемники D1 и D2 и записываются в буферное ОЗУ D3 (аппаратура формирования адресных и управляющих сигналов не показана, хотя и ее нужно контролировать; однако это уже несколько иная сфера и ее не будем касаться). Каждый входной байт с контрольным разрядом проверяется на четность числа единиц с помощью элемента D4 суммирования по модулю два. Если общее число единиц четно (ошибок нет), то триггер D7, исходно установленный в 0, в момент поступления синхросигнала CL1 подтверждает исходное состояние.

Одинокная ошибка в 9-разрядном коде на выходах приемников D1 и D2 приводит к нарушению условия четности числа единиц, поэтому триггер D7 устанавливается в единицу, на выходе устройства формируется сигнал ERR A=1, свидетельствующий об ошибке данных на входах ОЗУ. Сигнал лог. 0 с нулевого плеча триггера D7 закрывает элементы И D11 и D12, чтобы не формировались признаки ошибок в более глубоких слоях структуры; это облегчает пользователю отыскание первопричины ошибки.

Если одинокная ошибка возникла на выходах ОЗУ (других ошибок нет — здесь мы рассматриваем только одиночные ошибки, двойные и большей кратности не обнаруживаются), то она выявляется элементами D6 и D8 при поступлении синхросигнала CL2. При этом формируется сигнал ошибки ОЗУ: ERR RAM=1. Элемент И D11 закрывается сигналом лог. 0 с выхода нулевого плеча триггера D8, чтобы у пользователя не возникало подозрений в нарушении работоспособности блоков последующих уровней.

Считанные из ОЗУ данные K0—K7 и контрольный разряд четности PK поступают на первую группу входов блока F, например сумматора. На вторую группу входов этого блока подается операнд G0—G7 с контрольным разрядом четности PG («происхождение» этого операнда и его контрольного разряда может быть аналогичным: шина—приемники—ОЗУ с соответствующими схемами контроля, хотя возможны и иные варианты—суть не в этом). На выходе блока F формируется 8-разрядный код данных E0—E7 (например, содержащий младшие 8 разрядов суммы) и контрольный разряд четности PE, полученный косвенными вычислениями, а не непосредственной «сверткой» кода E0—E7 (подробнее об этом — чуть позже, а пока продолжим рассмотрение работы устройства в целом).

Код E0—E7 поступает в шину В через передатчики D15. Контрольный разряд четности PAR В формируется элементом D9 суммирования по модулю два и проходит в шину В через передатчик D16. Если элементы D9, D15 или D16 работают неправильно, то в шине В нарушается условие общей четности числа единиц. Этот факт регистрируется проверкой четности принятого из шины В кода с помощью элемента D13 суммирования по модулю два. Сигнал ошибки в шине В (ERR В=1) в соответствующий момент фиксируется во внешнем триггере (не показан).

Элемент D10 суммирования по модулю два сравнивает сигналы PE и PAR В. При правильной работе устройства эти сигналы совпадают. Если одинокная ошибка вызвана неправильной работой блока F (D5), то нарушается условие общей четности числа единичных сигналов в группе E0—E7, PE. В этой ситуации контрольные разряды PE и PAR В не совпадают, сигнал лог. 1 с выхода элемента D10 проходит через элемент И D11, в результате формируется признак ошибки блока F (ERR F=1), который в соответствующий момент принимается внешним триггером (не показан). Таким образом, все устройство «охвачено» контролем по четности.

В заключение рассмотрим два примера построения блока F. В первом примере (рис. 2.29) блок представляет собой сумматор с последовательным переносом, выполненный с использованием одноразрядных сумматоров (рис. 2.30), содержащих дублированные схемы формирования переноса. Сигнал S (сумма) формируется сложением по модулю два сигналов X, Y и Z; сигналы S и S' (основной и дублированный переносы) вырабатываются при наличии на входах двух или трех единиц. Сигнал PE образуется суммированием по модулю два контрольных разрядов PG, PK и дублированных переносов

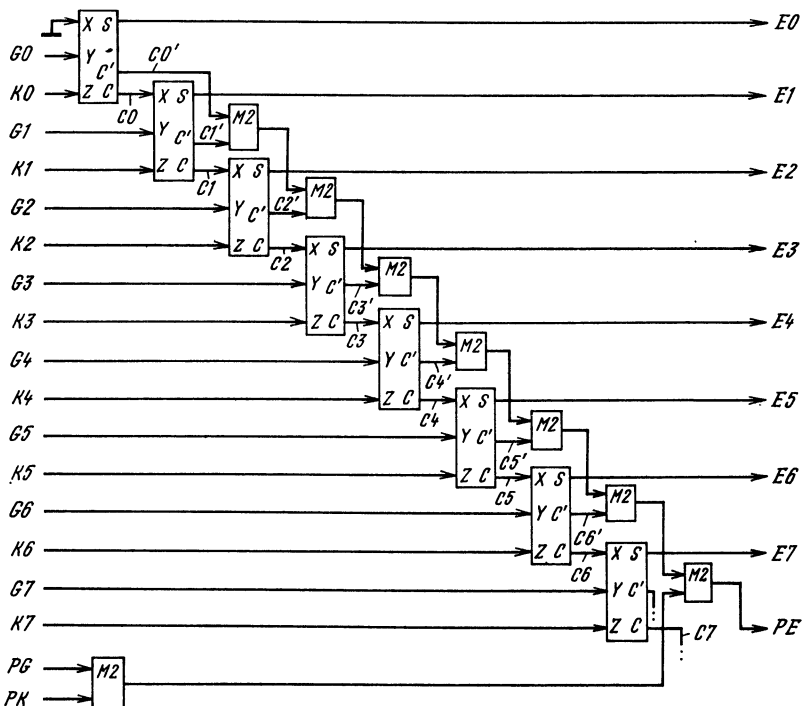


Рис. 2.29. Первый пример построения блока F

$C0'—C6'$ . Подтвердим правомерность такого решения

$$\begin{aligned}
 E0 &= G0 \oplus K0; & C0 &= G0K0; \\
 E1 &= G1 \oplus K1 \oplus C0; & C1 &= G1K1 + G1C0 + K1C0; \\
 E2 &= G2 \oplus K2 \oplus C1; & C2 &= G2K2 + G2C1 + K2C1; \\
 & \dots & & \dots \\
 E7 &= G7 \oplus K7 \oplus C6; & C7 &= G7K7 + G7C6 + K7C6;
 \end{aligned}$$

$$\begin{aligned}
 PE &= E0 \oplus E1 \oplus \dots \oplus E7 = G0 \oplus G1 \oplus \dots \oplus G7 \oplus K0 \oplus K1 \oplus \dots \oplus K7 \oplus \\
 &\oplus C0 \oplus C1 \oplus \dots \oplus C6 = PG \oplus PK \oplus C0 \oplus C1 \oplus \dots \oplus C6 = \\
 &= PG \oplus PK \oplus C0' \oplus C1' \oplus \dots \oplus C6'.
 \end{aligned}$$

Благодаря использованию дублированных схем формирования переносов исключаются «необнаруживаемые» ошибки, которые одновременно искажают сумму (код  $E0—E7$ ) и контрольный разряд  $PE$ . Таким образом, сигнал  $PE$  «предсказывает» четность суммы, а не является результатом ее свертки по модулю два, что позволяет контролировать правильность работы блока F.

1. Будем считать, что одиночных ошибок во входных данных нет, так как в противном случае сработали бы схемы контроля четности на входах блока F (на рис. 2.28 показана только схема  $D6—D8$  контроля канала K).

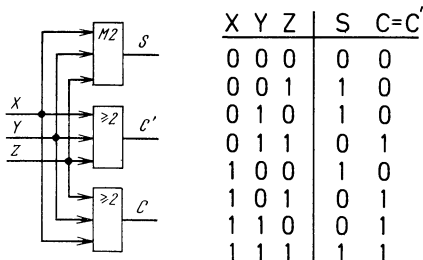


Рис. 2.30. Одноразрядный сумматор с дублированной схемой формирования переноса

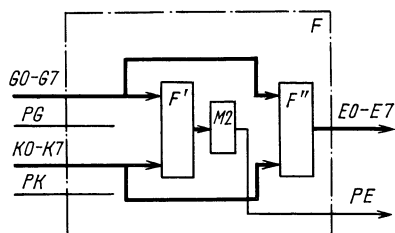


Рис. 2.31. Второй пример построения блока F

2. Если один из сигналов переносов  $C_i$  постоянно равен 0 (при этом сигнал  $C'_i$  остается правильным, так как рассматриваем только одиночные ошибки), то такая ошибка обнаруживается при сложении кодов, порождающих данный перенос. Пусть, например, сигнал  $C_4$  постоянно равен 0 и складываются следующие коды:

Номер разряда	7 6 5 4 3 2 1 0	
Код 67-60	0 0 0 1 0 0 0 0	$PG = 1$
Код 77-70	0 1 0 1 0 0 0 0	$PK = 0$
+-----		
Код E7-E0	$\left\{ \begin{array}{l} 0 1 1 0 0 0 0 0 \\ 0 1 0 0 0 0 0 0 \end{array} \right.$	← Правильный результат ← Неправильный результат
из-за отсутствия переноса из разряда 4 ( $C_4 = 0$ , $C'_4 = 1$ )		

$$PE = PG \oplus PK \oplus C'_4 = 0$$

«Предсказанный» контрольный разряд  $PE=0$  не соответствует четности числа единиц в неправильном результате, что обнаруживается элементом D10.

3. Если один из сигналов переносов  $C_i$  постоянно равен 1 (сигнал  $C'_i$  правильный), то такая ошибка обнаруживается при сложении кодов, в которых данный перенос отсутствует. Пусть, например, сигнал  $C_4$  постоянно равен 1 и складываются следующие коды:

Номер разряда	7 6 5 4 3 2 1 0	
Код 67-60	1 0 0 0 0 0 1 0	$PG = 0$
Код 77-70	1 0 0 0 0 1 1 0	$PK = 1$
+-----		
Код E7-E0	$\left\{ \begin{array}{l} 0 0 0 0 1 0 0 0 \\ 0 0 1 0 1 0 0 0 \end{array} \right.$	← Правильный результат ← Неправильный результат из-за
ложного переноса из разряда 4 ( $C_4 = 1$ , $C'_4 = 0$ )		

$$PE = PG \oplus PK \oplus C'_1 \oplus C'_2 = 1$$

Здесь также обнаруживается несоответствие вычисленного контрольного разряда  $PE=1$  фактической четности числа единиц в неправильном результате сложения.

4. Если ошибка типа Постоянный 0 или 1 присутствует на выходе S элемента суммирования по модулю два какого-либо одноразрядного сумматора (при этом сигналы C и C' правильные), то она обнаруживается при сложении кодов, сумма которых в соответствующем разряде равна 1 или 0. Иными словами, такая ошибка изменяет ожидаемую четность числа единиц в коде E0—E7, не влияя на правильность работы цепей переноса. Поэтому в данных ситуациях сигнал PE также не будет соответствовать четности числа единичных сигналов кода E0—E7.

В примере, приведенном на рис. 2.31, сигналы PG и PK не используются для формирования сигнала PE. Блок F состоит из двух одинаковых подблоков (комбинационных схем) F' и F'', первый из которых дублирующий и служит для контроля работы второго—основного. Ошибки в канале G или K, которые могли бы вызвать одновременную неправильную работу обоих подблоков, регистрируются соответствующими внешними схемами контроля в канале. Одиночные ошибки подблока F'' приводят к потере соответствия сигнала PE сигналам E0—E7, что и требуется.

### 2.13. Контролируемая схема сопряжения порта микроЭВМ с нагрузочными резисторами [31]

В схеме, показанной на рис. 2.32, микроЭВМ управляет подачей энергии в мощные нагрузочные резисторы  $R_n$  через ключевые блоки F1—F4. Сигнал  $C_i=0$  ( $i=1, 2, 3, 4$ ), выдаваемый из порта микроЭВМ, соответствует отключению нагрузочного резистора от источника напряжения питания +U; при  $C_i=1$  нагрузочный резистор подключается к этому источнику.

Если блоки F1—F4 исправны, то линии для передачи сигналов  $C_i$  можно рассматривать как однонаправленные, соединяющие выходы микроЭВМ с входами блоков. При обнаружении отказа блока на вход прерывания микроЭВМ поступает сигнал E=0. МикроЭВМ реагирует на него, переключается на прием сигналов C1—C4 и после их считывания определяет местоположение отказавшего блока. Рассмотрим эти процессы подробнее, пользуясь структурной схемой блока  $F_i$  (рис. 2.33).

При нормальной работе системы (когда все блоки  $F_i$  исправны) на выходе G порта присутствует сигнал лог. 1 (напряжение высокого уровня), поэтому транзистор VT1 закрыт и в качестве коллекторной нагрузки для транзисторов VT2 блоков  $F_i$  фактически используется только резистор R4, сопротивление которого относительно велико. Транзисторы VT2 всех блоков  $F_i$  выключены, поэтому E=1.

Компаратор D1 сравнивает напряжение  $U_3$ , поступающее с эмиттера транзистора VT2, с некоторым опорным напряжением  $U_A$ , выбранным достаточно большим. Так как транзистор VT2 закрыт,  $U_A > U_3$ , на выходе компаратора сформирован сигнал  $Y=1$ . Этот сигнал поддерживает ключ SW1 в состоянии, показанном на рисунке (изображен механический, а не электронный ключ, чтобы не усложнять рисунок несущественными схемотехническими деталями).

Сигнал  $C_i$ , принимаемый в блок  $F_i$  из порта микроЭВМ, проходит через ключ SW1 и поступает на вход триггера, выполненного на неинвертирующем усилителе D3 с резистором R6 в цепи обратной связи. Этот резистор достаточно высокоомный, поэтому он не препятствует изменению уровня напряжения на входе элемента D3 при изменении сигнала  $C_i$ . Забегая вперед, отметим, что триггер на элементе D3 используется для хранения бита данных в тех ситуациях, когда микроЭВМ переключилась на прием сигналов C1—C4.

Однако вернемся к рассмотрению нормальной работы системы. Сигнал  $C_i$  проходит через элемент D3 и поступает на управляющий вход ключа

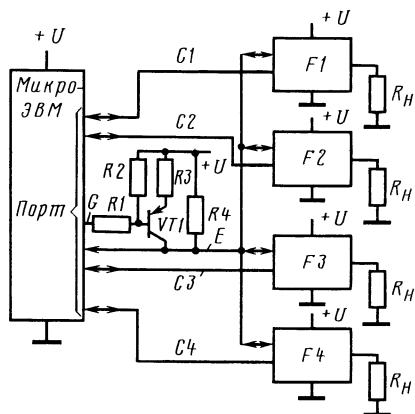


Рис. 2.32. Схема сопряжения порта микро-ЭВМ с нагрузочными резисторами  $R_H$ . При отказе одного из блоков F1—F4 микроЭВМ программно определяет его местоположение

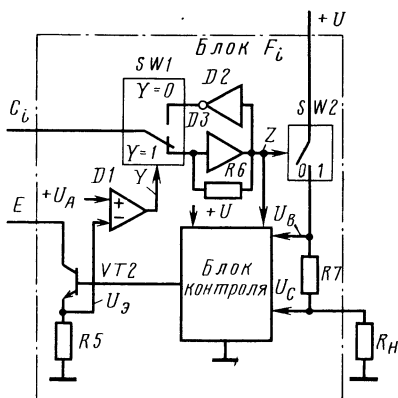


Рис. 2.33. Структура блока  $F_i$

SW2 (здесь также для простоты показан механический, а не электронный ключ). При  $Z=0$  этот ключ разомкнут, ток через нагрузочный резистор практически отсутствует; при  $Z=1$  ключ замкнут и к нагрузочному резистору  $R_H$  приложено напряжение, близкое  $+U$  ( $R7 \ll R_H$ ).

Блок контроля следит за правильностью реакции блока  $F_i$  на сигнал  $C_i$ . В установившемся режиме при правильной работе системы возможны две ситуации. В первой  $C_i = Z = 0$ , ключ SW2 разомкнут,  $U_B \approx U_C \approx 0$ . Во второй  $C_i = Z = 1$ , ключ SW2 замкнут,  $U_B \approx U_C \approx U$ , разность напряжений  $U_B$  и  $U_C$  не превышает заданного порога. Последнее условие позволяет контролировать перегрузки по току, когда сопротивление нагрузочного резистора  $R_H$  по каким-либо причинам становится ниже допустимого.

Если отклонений от нормальной работы нет, то блок контроля поддерживает транзистор VT2 в выключенном состоянии. В противном случае этот транзистор открывается. Так как  $R5 \ll R4$ , а транзистор VT1 выключен, то уровень сигнала Е снижается до стандартного уровня лог. 0, что вызывает прерывание выполнения фоновой программы процессором микроЭВМ и переход к прерывающей программе поиска отказавшего блока F<sub>1</sub>. Напряжение  $U_3$  в отказавшем блоке по-прежнему ниже, чем  $U_A$ ,  $Y=1$ .

При выполнении прерывающей программы процессор переключает линии порта на прием сигналов C1—C4. Это не приводит к потере старых данных, выдававшихся в блоки F<sub>i</sub>, благодаря триггерным схемам D3—R6. Затем процессор микроЭВМ формирует выходной сигнал G=0, транзистор VT1 включается, резистор R3, сопротивление которого соизмеримо с сопротивлением резистора R5, подключается в качестве дополнительной коллекторной нагрузки транзистора VT2 отказавшего блока. Ток через резистор R5 в отказавшем блоке существенно возрастает, так что выполняется условие  $U_3 > U_A$ , следовательно, Y=0. (В исправных блоках Y=1.) Ключ SW1 в отказавшем блоке переводится в верхнее положение, в действие вступает инвертор D2.

В результате подключения инвертора к линии отказавший блок формирует в ней сигнал  $C_1$ , противоположный тому, который ранее принимался из порта микроЭВМ. Исправные блоки формируют те же сигналы  $C_1$ , которые были приняты ранее, так как они снимаются непосредственно с триггеров D3—R6 (без использования инверторов D2).

Процессор микроЭВМ принимает код  $C1—C4$ , сопоставляет его с копией старого кода, хранимой в памяти, находит несовпадающий разряд и «узнаёт» о том, какой блок  $F_i$  неисправен. Дальнейшее поведение системы определяется ее программным обеспечением и не представляет для нас интереса.

Отметим, что при выполнении прерывающей программы эталонный код можно получить считыванием сигналов  $C1—C4$  через порт микроЭВМ в тот период, пока транзистор  $VT1$  еще не включен. В этом случае нет необходимости хранить в памяти копию старого кода.

## 2.14. Пример рационального использования разрядов порта ввода—вывода микроЭВМ в управляющей системе с самоконтролем [32]

Фрагмент системы, показанный на рис. 2.34, отражает возможность уменьшения числа разрядов порта ввода—вывода микроЭВМ, необходимых для ручного и автоматического управления объектом (креслом водителя автомобиля), а также для обеспечения самоконтроля системы.

Если ключ  $SW1$  разомкнут, то управление током через обмотку реле  $K1$  осуществляется от микроЭВМ. При  $D0=0$  (напряжение низкого уровня) транзистор  $VT1$  включен, через обмотку реле  $K1$  течет ток, при  $D0=1$  (передатчик с открытым коллектором выключен) напряжение на базе транзистора  $VT1$  близко к  $+U$ , этот транзистор выключен, ток через обмотку реле  $K1$  практически отсутствует.

По желанию оператора ключ  $SW1$  может быть замкнут, при этом независимо от сигнала, поступающего от передатчика (активный 0 или пассивная 1), напряжение на базе транзистора  $VT1$  равно 0, поэтому через обмотку реле  $K1$  протекает ток.

Факт замыкания ключа  $SW1$  может регистрироваться процессором микроЭВМ (если это событие существенно для него) следующим образом. Процессор периодически, например с частотой 50 Гц, на короткое время выключает передатчик, соединенный с ключом  $SW1$  (если этот передатчик был включен), и через приемник считывает сигнал  $D0$ . Если  $D0=0$ , то ключ  $SW1$  замкнут, если  $D0=1$ , то ключ  $SW1$  разомкнут. При разомкнутом ключе  $SW1$  и нулевом исходном состоянии сигнала  $D0$  кратковременное отключение передатчика не вызывает выключения реле  $K1$  из-за его инерционности.

Замыкание ключей  $SW2$  и  $SW3$  вызывает срабатывание реле  $K2$  и  $K3$ . Если ключ разомкнут, то управление состоянием реле осуществляется от микроЭВМ. Программный опрос с кратковременным отключением передатчиков (если они были включены) позволяет процессору определить текущее состояние ключей  $SW2$  и  $SW3$  так же, как было описано ранее. Передатчики выполнены по схеме с открытым коллектором.

Чтобы исключить потери машинного времени, связанные с периодическим программным опросом источника информации, обычно используют режим прерывания процессора. В данном примере в момент замыкания ключа  $SW2$  или  $SW3$  на входе

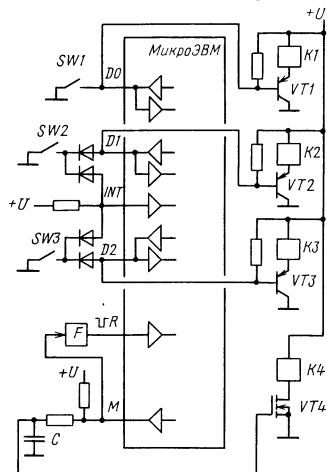


Рис. 2.34. МикроЭВМ в системе управления объектом

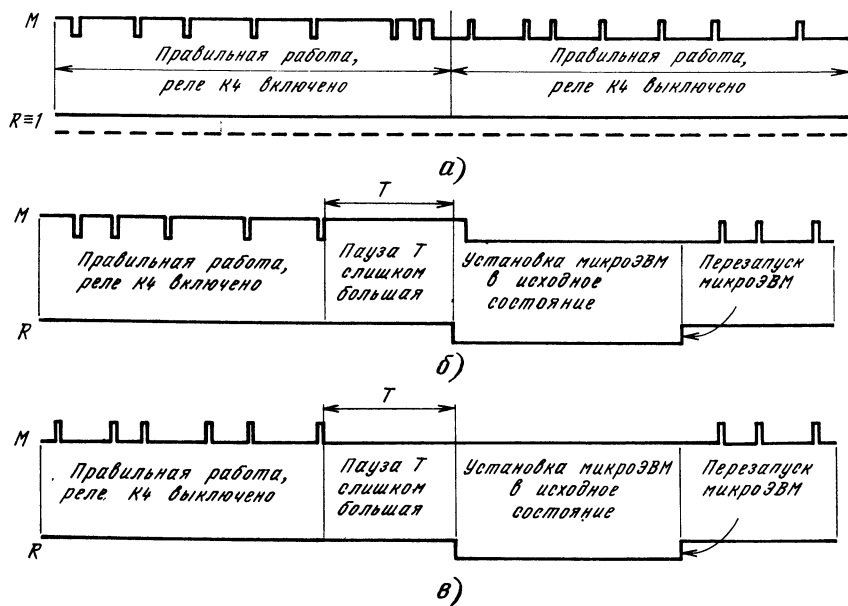


Рис. 2.35. Временные диаграммы работы схемы самоконтроля:  
 а—в отсутствие ошибок; б—при наличии ошибки на фоне сигнала  $M=1$ ; в—при наличии ошибки на фоне сигнала  $M=0$

прерывания микроЭВМ формируется сигнал  $INT=0$ , с появлением которого процессор временно прекращает выполнение текущего задания, запоминает параметры возврата к нему и переходит к выполнению прерывающей программы. При работе по этой программе процессор, как было описано ранее, путем опроса уточняет, какой ключ замкнут, затем выполняет соответствующие действия и возвращается к прерванному заданию.

Самоконтроль системы осуществляется следующим образом. Рабочая программа построена так, что при правильном ее выполнении (рис. 2.35, а) на фоне «статического» сигнала  $M$ , равного 1 или 0 в зависимости от необходимости поддержания реле  $K4$  во включенном или выключенном состоянии, формируются короткие, противоположные по логическому значению импульсы, которые фильтруются RC-цепью и поэтому не воздействуют на транзистор  $VT4$ . При правильной работе системы короткие импульсы следуют достаточно часто, поэтому формирователь импульса  $F$ , реагирующий на любой фронт сигнала, поддерживает на выходе сигнал  $R=1$  (отсутствие установки микроЭВМ в исходное состояние). Если пауза  $T$  между импульсами любой полярности превышает некоторую заданную величину (программа выполняется неверно), то формируется импульсный сигнал  $R=0$  (рис. 2.35, б, в), по которому микроЭВМ приводится в исходное состояние и перезапускается.

Передачик, формирующий сигнал  $M$ , в данном примере выполнен по схеме с открытым коллектором, поэтому его выход через нагрузочный резистор подключен к шине положительного напряжения источника питания. Интегрирующую RC-цепь можно исключить, а реле  $K4$  включить по той же схеме, что и реле  $K1—K3$ . Однако при этом инерционность реле  $K4$  должна быть достаточно большой или импульсные сигналы—достаточно короткими и следующими не слишком часто.

## 2.15. Контроль правильности формирования и обработки сигналов прерывания [33]

В обычной схеме связи микроЭВМ с контроллерами  $K_1 \dots K_N$  (рис. 2.36, а) сигналы прерываний  $RQ_1 \dots RQ_N$  генерируются при необходимости обслуживания внешних устройств со стороны ЦП. Эти сигналы поступают в контроллер прерываний, где из них выбирается наиболее приоритетный, который обрабатывается в первую очередь. Обработку прерываний можно запретить установкой в единицу разряда маски прерываний регистра состояния процессора ( $M=1$ ). Возможен также индивидуальный запрет формирования того или иного сигнала  $RQ_i$  установкой в единицу разряда маски прерывания регистра состояния соответствующего контроллера  $K_i$ .

Для самотестирования такой системы предусматривают возможность программного формирования сигналов  $RQ_i$ . Центральный процессор микроЭВМ последовательно устанавливает в единицу определенные разряды регистров состояния контроллеров и получает в ответ «искусственные» сигналы прерывания  $RQ_i$ , которые обрабатываются так же, как и обычные.

Приоритетную схему контроллера прерываний проверяют в несколько этапов. Сначала при  $M=1$  создают нужную комбинацию сигналов  $RQ_i$ , затем после снятия маски ( $M=0$ ) отслеживают правильность перехода процессора к наиболее приоритетной прерывающей программе (по результатам ее прогона). Такие проверки выполняются сравнительно медленно, так как контроллеры  $K_i$  нельзя адресовать одновременно. Кроме того, при обнаружении ошибки остается открытым вопрос «Кто виноват?» — контроллер  $K_i$ ,

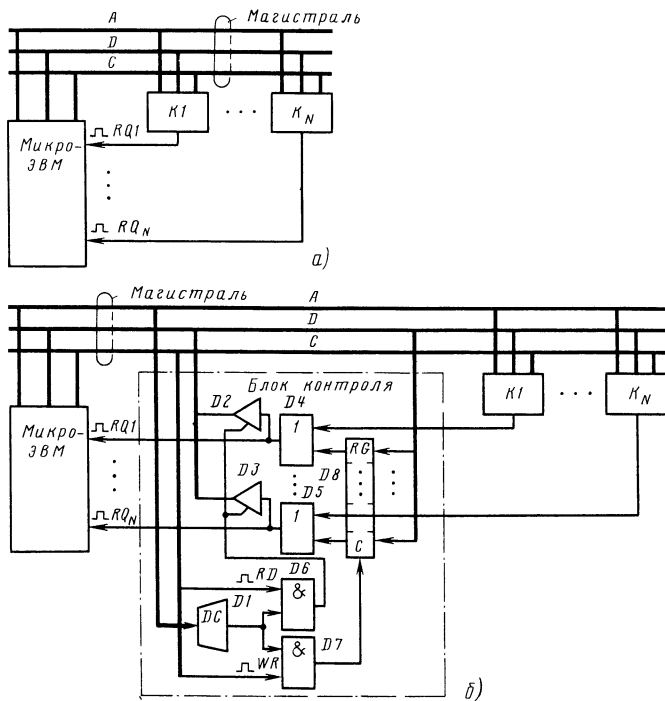


Рис. 2.36. Традиционная (а) и предлагаемая (б) схемы подключения контроллеров  $K_i$  к микроЭВМ

линия передачи сигнала  $RQ_i$  или оставшаяся часть аппаратуры. Предлагаемое решение (рис. 2.36, б) отчасти устраняет эти недостатки. В схему введен блок контроля, содержащий адресный дешифратор D1, усилители D2, D3 с «третьим состоянием», элементы ИЛИ D4, D5, И D6, D7 и регистр D8.

Элементы D2, D3 и D8 являются программно-доступными. Это означает, что к ним возможно обращение по некоторому адресу (как к обычной ячейке памяти), который распознаётся дешифратором D1. При чтении ( $RD=1$ ) срабатывает элемент И D6, усилители D2, D3 переходят в активное состояние и транслируют в процессор по шине данных D код с выходов элементов ИЛИ D4, D5. При записи ( $WR=1$ ) срабатывает элемент И D7, регистр D8 фиксирует данные, поступающие из процессора.

В нормальном режиме (в отсутствие тестирования) в регистре D8 хранится нулевой код, усилители D2, D3 выключены. Запросы прерываний от контроллеров  $K_i$  проходят через элементы ИЛИ D4, D5 и поступают в микроЭВМ, как в обычной схеме. В режиме тестирования отдельно проверяется правильность формирования запросных сигналов контроллерами и корректность работы остальной аппаратуры, размещенной в микроЭВМ и связанной с обработкой прерываний.

При проверке правильности формирования контроллерами запросных сигналов реакция на них запрещена, так как в регистре состояния процессора разряд маски (M) установлен в единицу. В регистре D8 хранится нулевой код. Процессор программно «заставляет» контроллеры формировать запросные сигналы в тех или иных сочетаниях и считывает их через усилители D2, D3, обращаясь к блоку контроля по некоторому адресу, как было показано. Считанные коды анализируются сопоставлением с эталонными.

При проверке остальной части схемы процессор запрещает формирование запросных сигналов от контроллеров  $K_i$ , записывая в регистр состояния каждого из них соответствующий код. Разряд M маски процессора устанавливается в 0, поэтому реакция на прерывания разрешена. Далее процессор записывает в регистр D8 код, имитирующий появление запросов в нужном сочетании, и отрабатывает процедуру прерывания с последующей программной оценкой правильности предпринятых действий. Подобные операции повторяются с разными кодами в регистре D8.

## **2.16. Использование диагностического процессора для оперативного отключения отказавших блоков от магистрали микроЭВМ [34]**

В микроЭВМ (рис. 2.37) блоками 1—N представлены процессоры, контроллеры устройств ввода—вывода, в том числе каналы прямого доступа в память, и иные функционально законченные узлы. Каждый блок конструктивно размещен на отдельной печатной плате. Особенность этой микроЭВМ состоит в том, что при отказе одного из блоков и его удалении из соответствующего разъема работоспособность микроЭВМ сохраняется, но ее производительность уменьшается.

В процессе работы микроЭВМ диагностический процессор периодически захватывает магистраль и проводит кратковременные сеансы связи с блоками 1—N, выявляя среди них неисправный. Если таковой найден, то он логически отключается от магистрали, чтобы не помешать остальным блокам обмениваться данными. Блок также отключается от магистрали при обнаружении неисправности внутриблочными схемами самоконтроля. Рассмотрим схемное решение, обеспечивающее логическое отключение неисправного блока (рис. 2.38).

Блок  $i$  соединен с шинами адреса A, данных D и управления C магистрали микроЭВМ через приемники D2, D4, D6 и передатчики D1, D3 и D5; последние выполнены по схеме с открытым коллектором, нагруженные

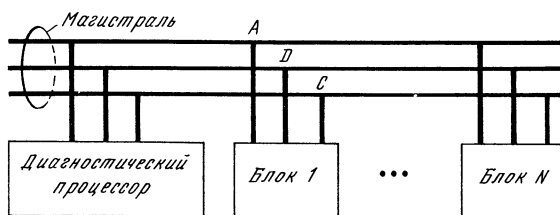


Рис. 2.37. МикроЭВМ с контролем работы блоков 1—N.  
Диагностический процессор выявляет отказавший блок и отключает его от магистрали

резисторы не показаны. Чтобы логически изолировать блок от магистрали и тем самым уменьшить вероятность отказа микроЭВМ в связи с отказом блока, передатчики D1, D3 и D5 в необходимых случаях переводятся в выключенное состояние с большим выходным сопротивлением сигналом  $EN=1$ . Разумеется, если отказ эквивалентен короткому замыканию одной или нескольких линий магистрали с шиной питания, шиной нулевого потенциала, электрическому объединению этих линий и т. п., то логическая изоляция блока не даст положительных результатов—придется физически удалить этот блок из микроЭВМ. Однако, будучи оптимистами, полагаем, что в большинстве случаев неисправности не столь «ужасны».

Итак, в отсутствие неисправностей блока  $i$  работа передатчиков D1, D3 и D5 разрешена ( $EN=0$ ) и они готовы транслировать сигналы в магистраль. Если блок находится в пассивном состоянии, то на всех входах передатчиков присутствуют сигналы лог. 1—напряжения высокого уровня. Выходные транзисторы этих передатчиков выключены и не препятствуют обмену данными по магистрали между «чужими» устройствами. Если блок активен, то он в соответствии с принятым интерфейсом использует линии магистрали для передачи и приема адресных, информационных и управляющих сигналов.

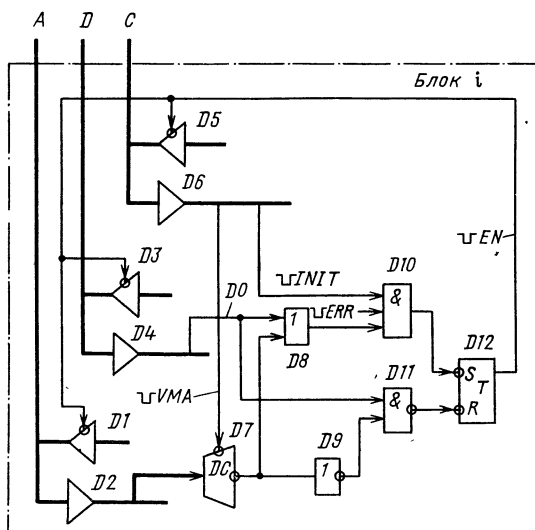


Рис. 2.38. Фрагмент схемы блока  $i$

Чтобы изолировать блок *i* от магистрали (в связи с неудовлетворительным результатом его тестирования), диагностический процессор обращается в режиме записи кода  $\times \times \times \dots \times 0_2$  ( $\times$  — произвольное значение разряда, 0 или 1) по некоторому адресу, который распознается дешифратором D7 при наличии сигнала VMA=0 подтверждения истинности адреса. Сигнал лог. 0 с выхода дешифратора поступает на нижний вход элемента ИЛИ D8, в то время как на его верхнем входе присутствует сигнал D0=0, полученный из диагностического процессора по линии младшего разряда шины данных. Поэтому элемент И D10 закрывается по нижнему входу (на остальных присутствуют сигналы лог. 1), триггер D12 переводится в состояние EN=1, блок *i* изолируется от магистрали.

Тот же результат получаем при воздействии на элемент И D10 сигнала ERR=0 от внутренней системы самоконтроля, когда ею констатируется отказ блока. И наконец, блок *i* изолируется от магистрали при поступлении сигнала INIT=0 начальной установки микроЭВМ. Этот сигнал может приходиться от кнопки на пульте управления, от блока питания при его включении или формироваться иным путем.

Для перевода триггера D12 в состояние EN=0 диагностический процессор записывает код  $\times \times \times \dots \times 1$  по адресу, на который реагирует дешифратор D7. В данном случае сигнал лог. 0 с выхода дешифратора проходит через инвертор D9, на нижнем входе элемента И—НЕ D11 формируется сигнал лог. 1. Так как D0=1, сигнал лог. 0 с выхода этого элемента воздействует на вход R установки нуля триггера D12.

После включения напряжения питания микроЭВМ все блоки 1—N логически отключены от магистрали. Диагностический процессор последовательно подключает к магистрали и проверяет каждый блок, так что далее в работу включаются только исправные блоки, которые, в свою очередь, периодически проверяются и при возникновении отказов изолируются.

Вместо логического отключения можно снимать напряжение питания с блока. Однако при этом необходимо выполнить два условия. Во-первых, отключение напряжения должно быть плавным, чтобы не было «ударов» по цепям питания, которые могли бы привести к ложным срабатываниям элементов исправных блоков. Во-вторых, элементы D1—D6 должны обладать высоким сопротивлением со стороны магистрали в обесточенном состоянии, чтобы не препятствовать обмену сигналами по магистрали между исправными блоками.

## 2.17. Быстрое выявление обращений процессора по «несуществующим» адресам [35, 36]

При работе микроЭВМ возможны ситуации, когда в результате сбоя, неисправности аппаратуры или вследствие ошибки программиста на адресной шине магистрали формируется «несуществующий» адрес, который не распознается ни одним из устройств, входящих в микроЭВМ. Так как ни одно из устройств не присылает ответный сигнал, инициатор обмена (процессор, канал прямого доступа) не может завершить начатую операцию и «зависает».

Чтобы состояние зависания не длилось неограниченно долго, обычно используют схему формирования искусственного ответного сигнала, которая срабатывает при отсутствии реакции абонента в течение определенного времени. Например, в микроЭВМ «Электроника МС1201.04» процессор ожидает ответ в течение примерно 10 мкс и в его отсутствие переходит к выполнению прерывающей программы реакции на зависание.

В предлагаемом решении для выявления обращения по «несуществующему» адресу не требуется цикл ожидания, т. е. зависания распознаются так же быстро, как выполняются обращения к реально существующим адресуемым

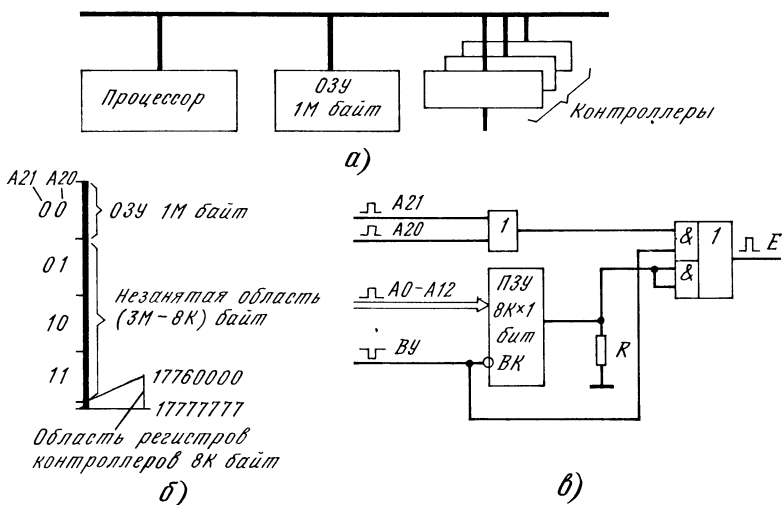


Рис. 2.39. МикроЭВМ (а), распределение ее адресного пространства (б) и схема выявления обращений по «несуществующим» адресам (в)

элементам. Пусть, например, магистраль микроЭВМ содержит 22-разрядную адресную шину; адресация байтовая, A21—старший, A0—младший разряды адреса (рис. 2.39). В адресном пространстве объемом 4М байт 0—1777777<sub>8</sub> ОЗУ размещено в области 0—3777777, регистры контроллеров внешних устройств занимают отдельные ячейки в пределах области объемом 8К байт: 17760000—17777777. Оставшаяся часть адресного пространства не занята. При обращении к регистрам внешних устройств процессор формирует в управляющей линии ВУ магистрالي напряжение низкого уровня, при обращении за пределы области 17760000—17777777 в этой линии присутствует напряжение высокого уровня.

Схема выявления обращений по «несуществующим» адресам (рис. 2.39, в) представляет собой адресный дешифратор, настроенный на опознание всех незанятых адресов. При их обнаружении на выходе дешифратора формируется сигнал E=1 (схема фиксации результата задержанным сигналом подтверждения истинности адреса не показана). В данном примере адреса представлены в прямом коде, сигналу лог. 1 соответствует напряжение высокого уровня. Схема работает следующим образом.

При обращении процессора к ОЗУ в двух старших разрядах A21 и A20 адреса установлен код 00, на выходе элемента ИЛИ сформирован сигнал лог. 0. Сигнал ВУ отсутствует, т. е. равен лог. 1, ПЗУ отключено, на его выходе за счет резистора R поддерживается сигнал лог. 0, E=0. При обращении процессора по адресам, содержащим в двух старших разрядах кодовые комбинации 01, 10 и 11 (исключая обращение к области 8К байт 17760000—17777777), сигнал ВУ также отсутствует (ВУ=1), на выходе элемента ИЛИ сформирован сигнал лог. 1, E=1.

При обращении к области 17760000—17777777 сигнал ВУ=0 закрывает верхний элемент ИИ схемы 2И—ИЛИ и активизирует вход выбора кристалла ВК ПЗУ. В зависимости от кода в 13 младших разрядах адреса A12—A0 из ПЗУ считывается закодированный в нем бит—0 или 1. Кодировка ПЗУ такова, что занятым в микроЭВМ адресам контроллеров соответствует 0, не занятым—1. Поэтому при ошибочных обращениях формируется сигнал E=1.

## 2.18. Устранение ошибок адресации, вызывающих беспорядочное «блуждание» процессора по памяти [37]

В типовой микроЭВМ (рис. 2.40) программы и данные, которыми оперирует ЦП, размещены в блоках памяти, в данном примере — в ПЗУ. Обычно в памяти остаются свободные (не занятые полезной информацией) области, которые не используются при правильной работе процессора.

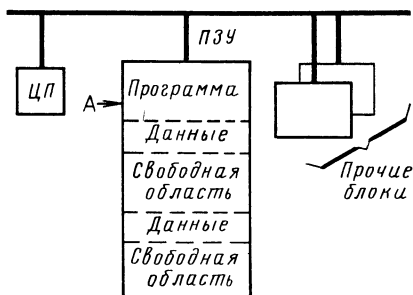


Рис. 2.40. Структура типовой микроЭВМ. Свободные области ПЗУ предлагается заполнить командами Сброс или Безусловный переход по адресу А

Однако при возникновении ошибок (например, в результате действия электрических помех по информационным, управляющим линиям или цепям источника питания) процессор может «сорваться» с предписанной ему траектории движения по программе и начать беспорядочное «блуждание» по памяти, рассматривая данные как команды, команды — как операнды или их адреса и т. п. При таком блуждании, вероятно, процессор попытается извлечь команду из свободной области. С увеличением числа свободных областей и их объема растет вероятность ошибочного попадания в них.

Предлагается заполнить свободные области блоков памяти кодами команд Сброс или Безусловный переход по адресу А. После такого заполнения «заблудившийся» процессор автоматически вернется к исходной или некоторой заданной промежуточной точке А программы, встретив одну из указанных команд.

Для повышения вероятности возврата процессора «на путь истинный» в микроЭВМ можно ввести дешифратор не задействованных адресов, который срабатывает при ошибочных обращениях к незанятым областям адресного пространства (не путать их со свободными областями адресов блоков памяти, см. рис. 2.40). При срабатывании этого дешифратора либо формируется сигнал Ошибка, Прерывание, Сброс и т. п., либо в шину данных магистрали выдается «распаянный» код одной из упомянутых команд для возврата процессора к начальной или заданной промежуточной точке программы.

## 2.19. Схема контроля длительности выполнения программного цикла, целостности линий магистрали и порядка следования выполняемых фрагментов программы [38, 39]

Если программа выполняется циклически, с заранее известной максимальной длительностью цикла, то, измеряя с помощью внешнего таймера фактическое время прохождения каждого цикла программы, можно судить о правильности работы микроЭВМ.

Обычно для контроля в соответствии с данным критерием используют аппаратный счетчик, на вход синхронизации которого подаются импульсы

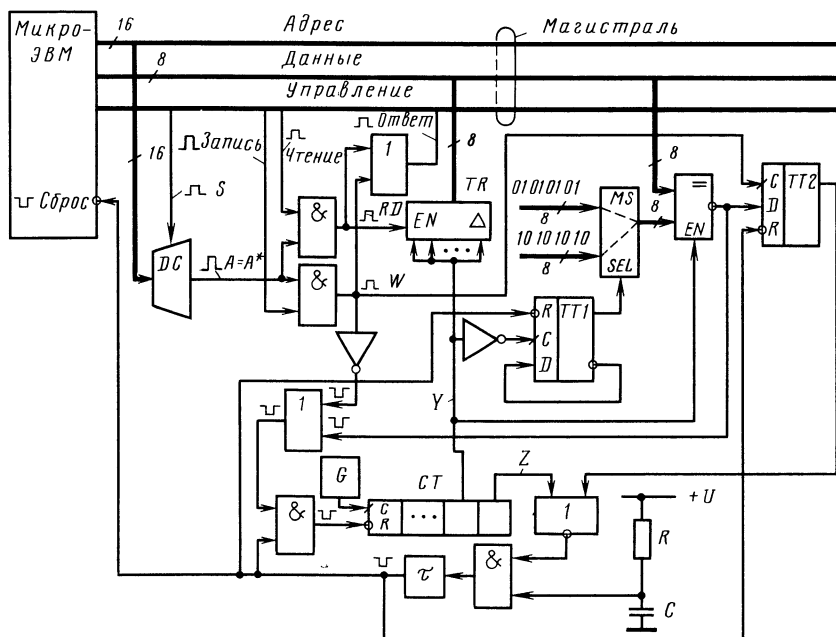


Рис. 2.41. МикроЭВМ со схемой контроля.

Сигнал сброса микроЭВМ формируется либо при длительном отсутствии сигнала установки 0 счетчика (таймера) СТ, либо при несовпадении кодов на входах компаратора в момент записи переменной М по адресу А\*, либо при включении напряжения питания

от генератора тактовой частоты, а на вход установки нуля—сигнал от микроЭВМ. При правильной работе микроЭВМ импульсы установки 0 счетчика поступают достаточно часто, в каждом цикле выполнения программы, поэтому переполнения счетчика не возникает. Если по каким-либо причинам импульс установки 0 счетчика не сформирован, то сигнал переполнения вызывает начальную установку микроЭВМ и ее перезапуск.

Такой метод контроля, однако, не обеспечивает обнаружения ошибок, при которых микроЭВМ работает неправильно, но счетчик тем не менее периодически устанавливается в 0 и его переполнения не возникает. Это может произойти, например, если некоторый фрагмент циклического участка программы не выполняется из-за искажения кодов в ОЗУ или по каким-либо иным причинам. Чтобы повысить достоверность контроля, предлагается устанавливать счетчик в 0 только при выполнении определенных условий. Прежде чем пояснить эту идею, рассмотрим работу составных частей схемы контроля, приведенной на рис. 2.41.

С точки зрения процессора микроЭВМ схема контроля представлена 8-разрядной ячейкой памяти с некоторым адресом А\*, принадлежащим незанятой области адресного пространства. Процессор может записать или считать из этой ячейки 8-разрядный код. Для этого в программе предусматривается команда пересылки содержимого внутреннего регистра процессора в ячейку памяти с адресом А\* (запись) или команда пересылки содержимого ячейки памяти с этим адресом во внутренний регистр процессора (чтение). Однако указанная ячейка, как станет ясно из дальнейшего описания, «не простая» и предназначена вовсе не для хранения в ней записываемого кодового слова (считываемый код никогда не совпадает с ранее записанным).

Это всего лишь «ячейка связи» между микроЭВМ и схемой контроля, через которую осуществляется периодическая установка 0 счетчика (внешнего таймера) СТ и опрос состояния сигнала Y в одном из его разрядов (а именно в разряде, соседнем со старшим).

При выполнении операции записи кода в ячейку памяти с адресом, равным A\*, этот адрес устанавливается процессором в шине адреса магистрали, а записываемый код передается в шину данных. В линию Запись шины управления магистрали процессор выдает сигнал лог. 1 (напряжение высокого уровня). Истинность информации, установленной в шинах адреса, данных и управления, с некоторой задержкой подтверждается сигналом S=1, который открывает дешифратор DC, на его выходе вырабатывается сигнал лог. 1 — признак опознания адреса схемой контроля. В результате формируется сигнал W=1. Положительный фронт этого сигнала вызывает прием информации в D-триггер TT2 с выхода компаратора 8-разрядных кодов.

При правильной работе микроЭВМ триггер TT2 находится в состоянии лог. 0 и всякий раз с поступлением положительного фронта сигнала W=1 подтверждает это состояние, так как в этот момент компаратор формирует на инверсном выходе нулевой сигнал — признак совпадения входных кодов. Иными словами, при правильной работе микроЭВМ и выполнении ею операции записи по адресу A\* код, установленный процессором в шине данных магистрали, должен совпадать с кодом, поступившим с выхода мультиплексора MS. Этот мультиплексор в зависимости от сигнала на его входе SEL выбора направления передачи информации транслирует на выход одну из «распаянных» на его входах констант: 01010101 или 10101010. Управление мультиплексором MS осуществляется от D-триггера TT1, включенного в режиме деления частоты на 2.

Продолжая рассмотрение процесса записи кода по адресу A\*, отметим, что сигнал W=1 проходит через элементы НЕ, ИЛИ, И и воздействует на вход R установки 0 счетчика СТ, вход С синхронизации которого подключен к выходу тактового генератора G, формирующего непрерывную последовательность синхроимпульсов относительно высокой частоты. Наконец, сигнал W=1 проходит через элемент ИЛИ и в качестве ответного сигнала поступает в процессор, который, получив ответ, снимает ранее установленные сигналы S=1, Запись=1, а затем освобождает шины адреса и данных. На этом процесс записи заканчивается. Как видим, смысл этой операции не в «записи», а в проверке состояния компаратора и при удачной проверке — в установке счетчика (таймера) в исходное (нулевое) состояние.

При выполнении процессором операции считывания кода из ячейки памяти с адресом A\* этот адрес устанавливается в адресной шине магистрали, в шину управления выдается сигнал Чтение=1, а затем после некоторой задержки — сигнал S=1. Сигнал RD=1 открывает магистральные передатчики TR с объединенными информационными входами, а также проходит через элемент ИЛИ и в виде сигнала Ответ поступает в процессор.

Процессор выжидает определенное время, достаточное для окончания переходных процессов в шине данных, считывает код с этой шины, затем освобождает магистраль. На этом операция чтения завершается. Код, считанный процессором с шины данных, представляет собой восьмикратно размноженный сигнал Y. Условием Y=0 и Y=1 соответствуют считанные коды 00000000 и 11111111. Если в момент считывания сигнал Y переходит из состояния лог. 0 в состояние лог. 1, то в результате неодновременности формирования сигналов в линиях шины данных процессор может принять любой из 254 промежуточных кодов. В этом случае процессор повторяет считывание и при правильной работе системы получает код 11111111, так как к моменту повторного опроса сигнал Y заведомо принял установившееся значение, равное 1. Отметим, что при правильной работе невозможна ситуация, при которой в момент опроса сигнала Y этот сигнал переходит из состояния лог. 1 в состояние лог. 0.

Импульсный сигнал Сброс формируется при включении напряжения питания (благодаря RC-цепи), а также при обнаружении неправильной работы

системы, когда в старшем разряде счетчика СТ формируется сигнал  $Z=1$  или в триггер ТТ2 записывается лог.1. Длительность сигнала Сброс определяется временем распространения сигнала через элемент задержки  $t$ . По окончании импульса Сброс триггеры ТТ1, ТТ2 и счетчик СТ установлены в нулевое состояние, счетчик начинает накопление единиц, а процессор приступает к выполнению рабочей программы, которая разбита на фрагменты, разделенные проверочными операциями (рис. 2.42).

Блок 1 соответствует программно-управляемой установке внутренних и внешних программно-доступных элементов микроЭВМ (портов ввода—вывода, регистров, триггеров, ячеек памяти и т. п.) в исходное состояние. В частности, некоторым переменным М, В и С (каждая из которых хранится в соответствующей ячейке ОЗУ) присваиваются начальные значения:  $M=01010101$ ,  $B=0$ ,  $C=0$ .

Далее микроЭВМ приступает к выполнению основной работы, которая состоит, например, в управлении некоторой технологической установкой. Программа имеет циклический характер и выполняется «бесконечно». Она в данном примере разбита на три фрагмента (блоки 2, 5 и 9), содержание которых не представляет для нас интереса.

После выполнения фрагмента 1 основной программы (блок 2) процессор опрашивает состояние сигнала  $Y$  считыванием и последующим анализом

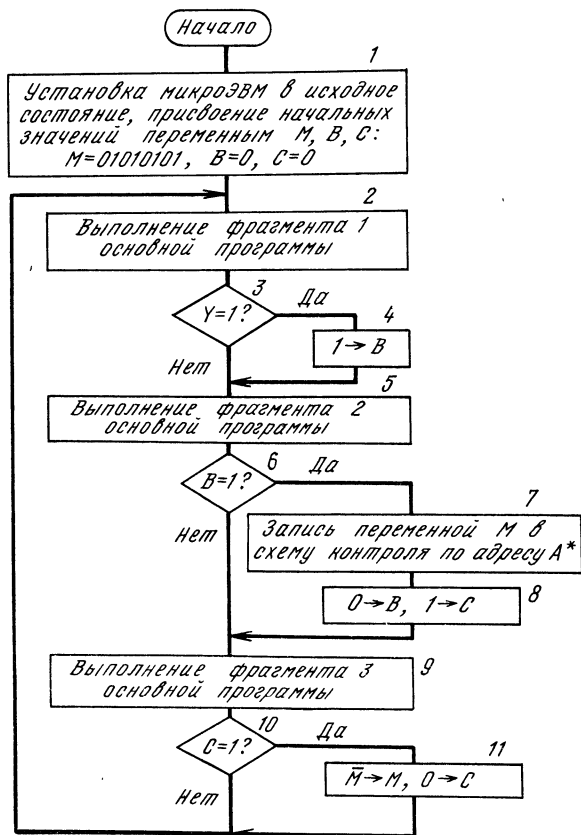


Рис. 2.42. Блок-схема алгоритма работы микроЭВМ (рис. 2.41)

содержимого ячейки памяти с адресом  $A^*$  (блок 3), как было показано ранее. При первом опросе (и, возможно, при ряде последующих)  $Y=0$ , так как счетчик СТ еще не успел накопить достаточного числа импульсов, поступающих с генератора Г, однако в дальнейшем наступает момент, когда при очередном опросе выясняется, что  $Y=1$ . Если это так, то переменной В присваивается единичное значение (блок 4) и далее управление передается фрагменту 2 основной программы (блок 5).

Как отмечалось, при считывании процессором ячейки памяти с адресом  $A^*$  могут быть получены промежуточные 8-разрядные коды, содержащие от одной до семи единиц. Если это произошло, то процессор сразу же повторяет считывание. Если, однако, и при повторном считывании не получен правильный результат (11111111), то это означает, что микроЭВМ работает неправильно, например, из-за неисправности линии шины адреса или данных. В этом случае процессор принимает к сведению факт появления ошибки и соответствующим образом реагирует на нее (если он еще способен реагировать). Описанные действия (повторное чтение, анализ и т. п.) не отражены на рис. 2.42 для его упрощения.

После выполнения фрагмента 2 основной программы (блок 5) проверяется значение переменной В (блок 6). Если  $B=1$ , то осуществляется запись переменной М по адресу  $A^*$  (блок 7). В результате (при правильной работе) подтверждается нулевое состояние триггера ТТ2, счетчик СТ устанавливается в 0, триггер ТТ1 переключается в противоположное состояние, мультиплексор МS меняет константу на нижнем входе компаратора для изменения условий проверки при последующем вхождении в блок 7.

Далее переменной В присваивается нулевое, а переменной С — единичное значение (блок 8). После выполнения фрагмента 3 основной программы (блок 9) проверяется значение С. Если  $C=1$ , то переменная М инвертируется, чтобы при последующем вхождении в блок 7 компаратор зарегистрировал совпадение нового кода данных с новым эталоном; переменной С присваивается нулевое значение (блок 11). Затем вновь выполняется фрагмент 1 основной программы (блок 2) и т. д.

Если при правильной работе сигнал  $Y=1$  чуть опоздал к моменту его проверки (блок 3), то он должен оставаться в состоянии лог. 1 до тех пор, пока не будет пройден путь 5—6—9—10—2—3—5—6—7 до момента установки счетчика в 0 блоком 7 (иначе в результате дальнейшего подсчета импульсов сформируется сигнал  $Z=1$  и сработает схема выдачи сигнала Сброс, хотя ошибки не было). Зная максимальное время прохождения этого пути, можно рассчитать требуемую частоту сигналов генератора Г или (и) необходимую разрядность счетчика СТ.

Подводя итог приведенному описанию, можно отметить, что рассредоточение моментов проверок и периодическая смена кодов (паролей), используемых для сравнения, уменьшают вероятность обнаружения ошибок. Иными словами, схема контроля в значительной мере «соединилась воедино» с программой, и подавляющая часть возможных отклонений от заданного алгоритма (рис. 2.42) влечет за собой либо несовпадение предьявленного кода с паролем из-за «срывов» программы, либо появление сигнала  $Z=1$  в старшем разряде счетчика СТ из-за длительного отсутствия сигнала установки 0 счетчика. В обоих случаях производится перезапуск системы с попыткой нового вхождения в рабочий цикл.

## **2.20. Межпроцессорный обмен данными с обнаружением ошибок формирования ответных сигналов [40]**

В многопроцессорной системе (рис. 2.43, а) платы расширения 1—N устанавливаются в разъемы, смонтированные на плате центрального (главного) процессора. Каждая плата расширения содержит периферийный процессор; число этих плат определяется потребностями пользователя системы. Нас

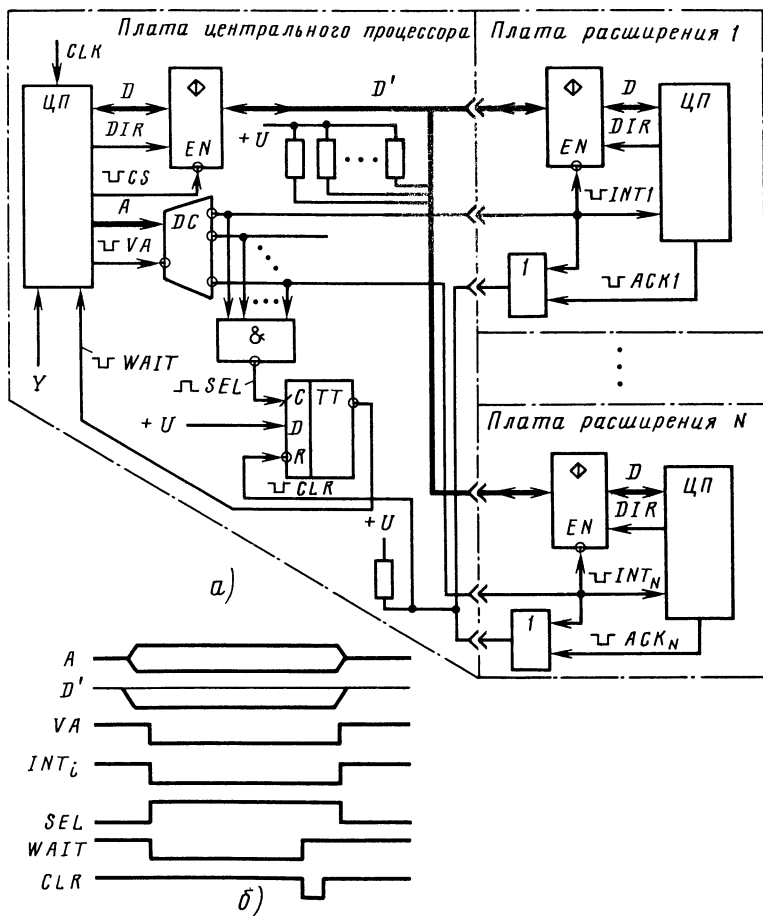


Рис. 2.43. Многопроцессорная система (а) и временные диаграммы (б) передачи слова из главного процессора в периферийный.

При отсутствии ответного сигнала от выбранной платы расширения триггер ТТ неограниченно долго остается в состоянии  $WAIT=0$ , главный процессор заторможен, система оказывается неработоспособной

будет интересоваться процесс взаимодействия главного процессора с периферийными, поэтому на рисунке показаны лишь те элементы и связи, которые имеют прямое отношение к этому процессу.

Обмен информацией между главным и выбранным с помощью дешифратора DC периферийным процессорами осуществляется по общей шине данных  $D'$ , которая соединена с индивидуальными шинами данных  $D$  процессоров через двунаправленные приемопередатчики. Приемопередатчики активизируются при подаче сигнала лог. 0 (напряжения низкого уровня) на вход  $EN$  разрешения работы. Направление передачи информации через приемопередатчик (слева направо или справа налево) задается сигналом  $DIR$ , поступающим из процессора. Шины  $D$  могут выполняться по схеме с открытым коллектором или с тремя состояниями (лог. 0, лог. 1, выключено). Сигнал

CLK поступает на синхронизирующий вход главного процессора с тактового генератора (на рисунке не показан).

Шина D' выполнена по схеме с открытым коллектором; каждая ее линия соединена через нагрузочный резистор с положительным полюсом +U источника питания. Напомним, что такая шина выполняет функцию Монтажное ИЛИ по отношению к сигналам, представленным напряжениями низкого уровня, т. е. допускает одновременную выдачу в нее информации из разных источников без их электрической перегрузки. Хотя суммирование сигналов в шине D' при нормальной работе системы не предусмотрено, оно может реализоваться при ошибочной работе процессоров. Если бы шина D' была выполнена по схеме с тремя состояниями, то одновременная (ошибочная) выдача в нее разной информации из двух или более приемопередатчиков могла привести к их отказам из-за электрической перегрузки выходных транзисторов.

В исходном состоянии все процессоры работают в автономном режиме, шина D' не используется, в ней поддерживаются сигналы лог. 1, сигналы DIR во всех платах соответствуют передаче информации из шины D' в шину D, все элементы ИЛИ с открытым коллектором получают на входы сигналы  $INT_i=1$  и  $ACK_i=1$ , в точке объединения выходов этих элементов присутствует пассивный сигнал лог. 1 ( $CLR=1$ ), триггер ТТ установлен в 0,  $WAIT=1$  (цепи начальной установки триггеров здесь и далее в этом параграфе не показаны на рисунках для их упрощения). Сигнал Y пока не принимаем во внимание, он используется при усовершенствовании системы.

Обмен информацией между главным и периферийным процессорами осуществляется в режиме записи или чтения по инициативе главного процессора. В режиме записи главный процессор сигналом  $CS=0$  активизирует приемопередатчики своей платы, сигналом DIR настраивает их на передачу информации из шины D в шину D' и выдает в шину D информационное слово—данные или команду подготовки периферийного процессора к режиму чтения в последующем сеансе связи. Одновременно с этим главный процессор выдает в шину A адрес периферийного процессора, в который нужно записать слово, и затем подтверждает истинность адреса сигналом  $VA=0$  (рис. 2.43, б). Этот сигнал открывает дешифратор DC, на одном из его выходов формируется сигнал  $INT_i=0$ , остальные выходы продолжают находиться в исходном состоянии (лог. 1).

Сигнал  $INT_i=0$  с выхода дешифратора DC поступает в плату расширения с номером i, активизирует приемопередатчики этой платы и воздействует на вход прерывания выбранного периферийного процессора. Одновременно с этим на выходе элемента И—НЕ формируется сигнал  $SEL=1$ , по фронту которого триггер ТТ устанавливается в 1, на его нулевом плече формируется сигнал ожидания  $WAIT=0$ . Главный процессор, получив этот сигнал, «замораживает» свое состояние и поддерживает его неизменным до момента установки триггера ТТ в 0.

Выбранный периферийный процессор временно откладывает выполнение текущего задания, считывает информацию с шины D, формирует импульсный сигнал ответа  $ACK_i=0$ , анализирует принятую информацию и возвращается к прерванному заданию. Сигнал  $ACK_i=0$  проходит через элемент ИЛИ своей платы и поступает на вход установки 0 триггера ТТ ( $CLR=0$ ), в результате снимается сигнал ожидания WAIT, главный процессор завершает выполнение команды записи, т. е. снимает все ранее выданные сигналы, система переходит в исходное состояние—процессоры работают по своим программам, шина D' свободна.

Режим чтения осуществляется после предупреждения о нем выбранного периферийного процессора передачей ему соответствующей команды при выполнении только что рассмотренного режима записи. Получив такую команду, периферийный процессор знает, что последующий сигнал прерывания будет означать, что необходимо переключить приемопередатчики в направлении  $D \rightarrow D'$  и выдать слово в шину D своей платы на некоторое время, сопроводив

это слово ответным сигналом  $ACK_i=0$ . Таким образом, режим чтения отличается от режима записи направлением передачи данных между главным и периферийным процессорами, в остальном схема работает аналогично. После выполнения режима чтения периферийный процессор расценивает очередной сигнал прерывания как соответствующий режиму записи (возможны, конечно, и иные соглашения).

Недостаток схемного решения, приведенного на рис. 2.43, а, состоит в том, что при отсутствии адресуемой платы расширения или при неисправности размещенной на ней аппаратуры, когда обобщенный ответный сигнал  $CLR=0$  не формируется, триггер ТТ не устанавливается в исходное состояние, главный процессор неограниченно долго остается в состоянии ожидания окончания обмена информацией ( $WAIT=0$ ), система зависает.

Если триггер ТТ исключить из системы, показанной на рис. 2.43, а, а вместо него ввести схему, приведенную на рис. 2.44, а, то получим систему, устойчивую к зависаниям. В усовершенствованной системе используется сигнал  $Y$ , который поступает на вход главного процессора и служит индикатором ошибки при межпроцессорном обмене данными.

При нормальной работе усовершенствованной системы одновременно с установкой в 1 триггера ТТ запускается одновибратор F и формирует сигнал  $H=1$  (см. рис. 2.44, б). Длительность  $\tau$  этого сигнала выбирается большей, чем максимальное время ожидания ответного сигнала CLR. Однако сигнал  $H=1$  заканчивается досрочно, в момент поступления ответного сигнала, триггер также устанавливается в исходное состояние, так что ранее сфор-

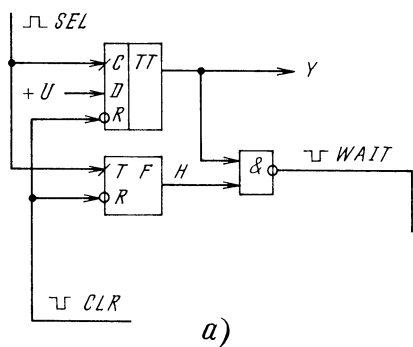
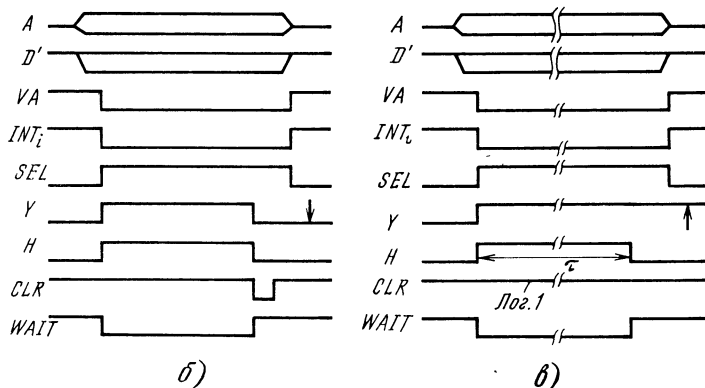


Рис. 2.44. Первый вариант усовершенствованной схемы формирования сигнала WAIT (а); временные диаграммы записи данных при получении (б) и неполучении (в) ответного сигнала от выбранного периферийного процессора.

Стрелки на временных диаграммах показывают моменты опроса сигнала  $Y$  главным процессором



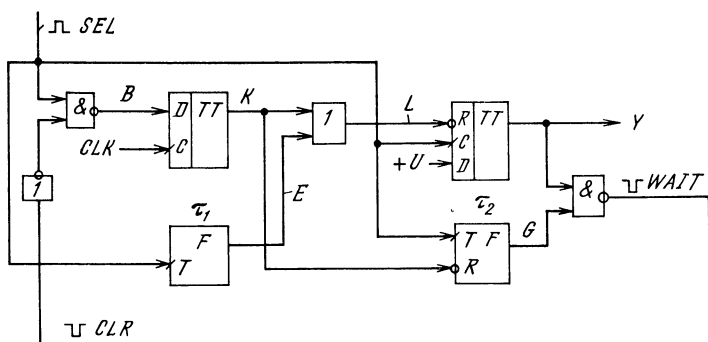


Рис. 2.45. Второй вариант усовершенствованной схемы формирования сигнала WAIT. При постоянных сигналах  $CLR \equiv 0$  или  $CLR \equiv 1$  сигнал WAIT остается импульсным, а сигнал Y переходит в состояние лог. 1 при обнаружении ошибки

мированный сигнал  $WAIT=0$  снимается. Процессор завершает обмен и проверяет состояние сигнала Y в момент, показанный стрелкой. Так как  $Y=0$ , процессор принимает к сведению, что обмен данными завершился успешно.

Если ответный сигнал  $CLR=0$  не поступил в течение времени  $\tau$  действия импульса  $H=1$  (рис. 2.44, в), то по окончании этого импульса сигнал  $WAIT=0$  снимается, процессор продолжает работу, однако в момент проверки сигнала Y, отмеченный на рисунке стрелкой, выясняется, что  $Y=1$  (абонент не выполнил предписанных ему действий). Таким образом, система не зависает, а главный процессор способен отличить состоявшийся обмен от не состоявшегося.

Рассмотренная схема (рис. 2.44, а), однако, неспособна зафиксировать ошибку, связанную с постоянным пребыванием сигнала CLR в состоянии лог. 0 из-за неисправности какого-либо компонента системы. Действительно, при  $CLR=0$  процессор не получает сигнал  $WAIT=0$  (это нормально с его точки зрения) и работает без циклов ожидания, при этом за счет постоянного действия сигнала лог. 0 на вход установки 0 триггера TT признак ошибки Y постоянно равен 0, процессор считает, что любой обмен завершается успешно и продолжает работу, фактически потерявшую смысл.

Чтобы избежать подобных ситуаций, можно применить более сложную схему формирования сигнала WAIT, показанную на рис. 2.45. Временные диаграммы работы усовершенствованной многопроцессорной системы приведены на рис. 2.46—2.48.

При отсутствии ошибок в системе процесс записи развивается так (рис. 2.46). В середине первого цикла  $T_1$  работы главного процессора данные выданы им в шину D', адрес — в шину A. В начале второго цикла  $T_2$  сигналом  $VA=0$  открывается дешифратор DC, формируются сигналы  $INT_1=0$ ,  $SEL=1$ , запускаются одновибраторы F. В середине цикла  $T_2$  процессор обнаруживает сигнал  $WAIT=0$ , и, пока действует этот сигнал, процессор выполняет «пустые» циклы ожидания  $T_w$ . Длительность  $\tau_1$  сигнала  $E=1$  на выходе первого одновибратора F выбирается меньшей минимального времени ожидания ответного сигнала  $CLR=0$ , поэтому к моменту поступления ответного сигнала первый одновибратор уже вернулся в исходное состояние  $E=0$ .

Второй одновибратор формирует импульс  $G=1$  длительностью  $\tau_2$ , превышающей максимальное время ожидания ответного сигнала  $CLR=0$ , однако этот импульс принудительно «обрезается» при поступлении сигнала  $K=0$ . Далее сигнал  $WAIT=0$  снимается, о чем процессор узнаёт в середине последнего цикла  $T_w$ . Процессор завершает этот цикл, освобождает шины A, D, снимает сигнал  $VA=0$  и в середине цикла  $T_3$  проверяет значение

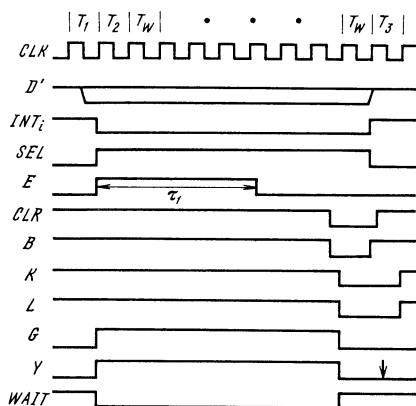


Рис. 2.46. Временные диаграммы работы системы (рис. 2.43, 2.45) в режиме записи при отсутствии ошибок

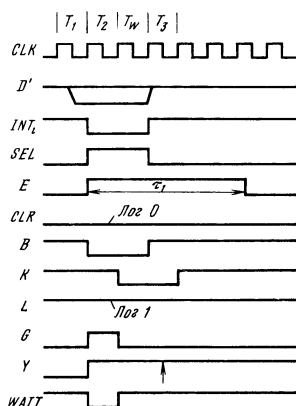


Рис. 2.47. Временные диаграммы работы системы (рис. 2.43, 2.45) в режиме записи при постоянном поддержании сигнала CLR в состоянии лог. 0

сигнала Y (момент проверки отмечен на рисунке стрелкой). В данном случае Y=0, что соответствует отсутствию ошибок.

При постоянном сигнале  $CLR \equiv 0$  (рис. 2.47) формируется сигнал WAIT=0 длительностью, равной одному периоду синхросигнала CLK. В середине цикла  $T_2$  процессор воспринимает сигнал WAIT=0 и выполняет цикл ожидания  $T_w$ , в середине которого процессор обнаруживает, что WAIT=1. Поэтому запись завершается и далее выполняется цикл  $T_3$ , в котором проверяется состояние сигнала Y. Так как в момент проверки, отмеченный на рисунке стрелкой, Y=1, главный процессор узнаёт, что обмен информацией с выбранным периферийным процессором не состоялся.

При постоянном сигнале  $CLR \equiv 1$  (рис. 2.48), т. е. при отсутствии ответа от абонента в течение времени  $\tau_2$  действия импульса G, формируемого одновибратором F, сигнал WAIT снимается, однако ранее установленный в 1 сигнал ошибки Y не меняет значения, что фиксируется процессором в середине цикла  $T_3$ .

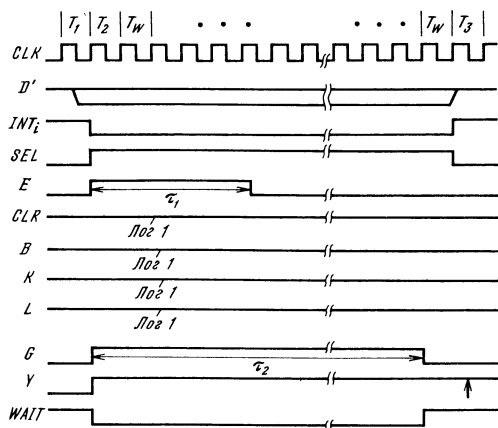


Рис. 2.48. Временные диаграммы работы системы (рис. 2.43, 2.45) в режиме записи при отсутствии ответного сигнала от выбранного периферийного процессора ( $CLR \equiv 1$ )

## 2.21. Точный или приближенный поиск эталонного кода в потоке последовательных входных данных [41]

Входные данные в виде последовательного потока бит поступают на информационный вход сдвигового регистра RG1 (рис. 2.49) и продвигаются в нем слева направо со скоростью, определяемой частотой сигнала синхронизации CLK. Задача состоит в том, чтобы распознать в проходящем потоке заранее заданную (эталонную) кодовую комбинацию или близкую к ней. Если такая комбинация найдена, формируется сигнал Поиск завершен.

В данном примере эталонный код содержит 12 бит: 00011001100 (показан в верхней части рисунка). Регистр RG1 содержит 13 разрядов. В общем случае при  $n$ -разрядном эталоне разрядность этого регистра равна  $n+1$  бит. Выходы разрядов A, D, G, I, K и M регистра RG1 соединены с входами первого сумматора. В общем случае к входам первого сумматора подключаются выходы первого (A) и последнего (M) разрядов сдвигового регистра RG1, а также ряд промежуточных выходов, соответствующих переходам  $0 \rightarrow 1$  (разряды D, I) и  $1 \rightarrow 0$  (разряды G, K) при просмотре эталонного кода (показанного над регистром RG1) слева направо.

Сумматор 1 обрабатывает входную информацию, рассматривая ее как совокупность чисел со знаком и руководствуясь следующими правилами. Если  $A=0$ , то операнд  $A'$ , участвующий в суммировании, принимается равным 0. Если  $A=1$ , то операнд  $A'$  принимается равным  $+1$ . (Если бы левый

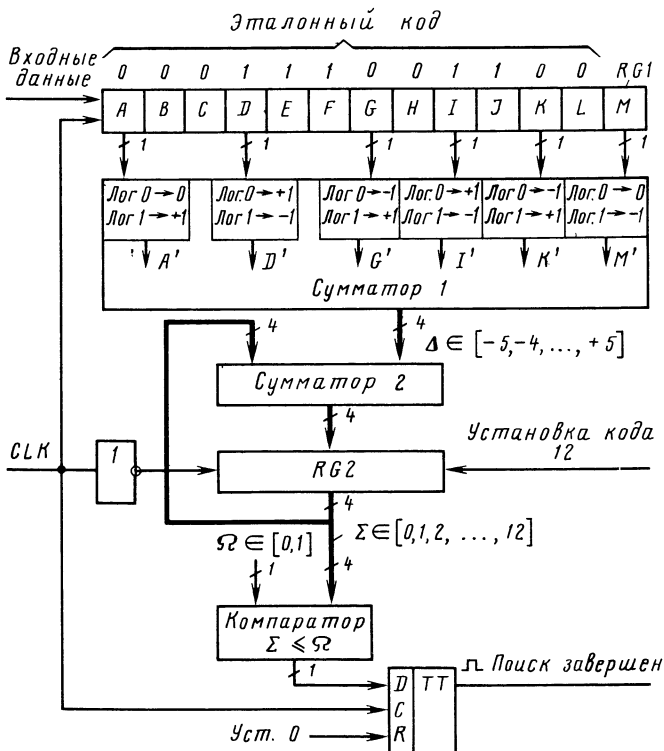


Рис. 2.49. Схема устройства для точного или приближенного поиска эталонного кода в потоке последовательных входных данных

разряд эталонного кода был равен 1, то при  $A=0$   $A'=+1$ , а при  $A=1$   $A'=0$ ). Логические значения сигналов с выходов разрядов D и I (соответствующих переходам  $0 \rightarrow 1$  в эталонном коде) преобразуются в арифметические значения операндов для суммирования следующим образом: при  $D=0$  ( $I=0$ )  $D'=+1$  ( $I'=+1$ ), при  $D=1$  ( $I=1$ )  $D'=-1$  ( $I'=-1$ ). Аналогично в разрядах G и K, соответствующих переходам  $1 \rightarrow 0$  в эталонном коде, условия преобразования противоположные: при  $G=0$  ( $K=0$ )  $G'=-1$  ( $K'=-1$ ), при  $G=1$  ( $K=1$ )  $G'=+1$  ( $K'=+1$ ).

Если  $M=0$ , то операнд  $M'$ , участвующий в суммировании, принимается равным 0. Если  $M=1$ , то  $M'=-1$ . (Если бы правый разряд эталонного кода был равен 1, то при  $M=0$   $M'=-1$ , а при  $M=1$   $M'=0$ .) В результате алгебраического суммирования полученных шести одноразрядных чисел со знаками образуется некоторая величина  $\Delta = A' + D' + G' + I' + K' + M'$  ( $-5 \leq \Delta \leq +5$ ), отражающая динамику сравнения проходящих через регистр RG1 кодов с заданным эталоном, как будет показано далее на примере.

Вычисленное сумматором 1 значение  $\Delta$  алгебраически складывается с числом  $\Sigma$ , поступающим из регистра RG2, результат сложения вновь записывается в этот регистр по отрицательному фронту сигнала CLK (сдвиг информации в регистре RG1 происходит в момент поступления положительного фронта сигнала CLK). Число  $\Sigma$  равно числу несовпадений между разрядами текущего и эталонного кодов к моменту формирования сигнала CLK=0. При точном совпадении текущего кода в разрядах A—L регистра RG с эталонным кодом выполняется условие  $\Sigma=0$ , при отличии кодов в одном (любом) разряде  $\Sigma=1$  и т. д., при отличии во всех 12 (n) разрядах  $\Sigma=12$  ( $\Sigma=n$ ). Исходно в разряды A—L регистра RG1 записан инвертированный эталонный код, в регистр RG2 записано число 12.

При работе устройства в каждом такте вычисляется очередное значение  $\Sigma$  и с помощью компаратора сравнивается с некоторым порогом. В данном примере порог  $\Omega$  устанавливается равным 0 или 1. Условие  $\Omega=0$  соответствует поиску точного совпадения текущего кода с эталонным. При  $\Omega=1$  поиск считается успешно завершенным, если обнаружено точное совпадение или приближенное, когда сравниваемые коды отличаются друг от друга только в одном разряде. Выход компаратора соединен с входом D-триггера, который принимает информацию по положительному фронту сигнала CLK. Исходно этот триггер установлен в 0, выходной сигнал завершения поиска отсутствует.

Рассмотрим пример, поясняющий работу устройства (рис. 2.50). В исходном состоянии в разрядах A—L регистра RG1 записан код 111000110011 (инверсия эталонного кода), в разряде M этого регистра записан 0 или 1 (см. символ  $\times$  в строке 1),  $\Sigma=12$  (отличия максимальны), CLK=0, входные данные, подготовленные для последующего поразрядного сдвига в регистр RG1, представляют собой эталонный код 000111001100. (Конечно, эталонный код может поступить на входы устройства не сразу после его пуска, а может и вообще не поступить, так что данный пример отражает наиболее благоприятную ситуацию.) Триггер ТТ установлен в 0.

В момент поступления первого положительного фронта сигнала CLK информация в регистре RG1 и входные данные сдвигаются на один разряд вправо, при этом освободившийся разряд входных данных заполняется единицей, как показано в строке 2. В регистре RG1 формируется код 0111000110011, который в разрядах A—L отличается от эталонного кода 000111001100 в семи битовых позициях:

$$\begin{array}{rcl} \oplus & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & & \text{— Код в разрядах A—L регистра RG1} \\ & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & & \text{— Эталонный код} \\ \hline & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & & \text{— Результат сравнения; число единиц} \\ & & & & & & & & & & & & & \text{(несовпадений) равно 7} \end{array}$$

Поэтому для правильного отражения текущей ситуации значение  $\Sigma$  должно уменьшиться от 12 до 7. Покажем, как этот результат формируется схемой.

Эталонный код → 0001111001100																												
	Входные данные																											
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
2	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
3	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
4	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
5	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
6	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
7	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
8	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
9	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
10	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
11	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
12	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
13	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
14	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
15	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
16	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
17	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
18	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
19	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
20	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
21	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
22	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
23	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
24	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
25	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
26	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
27	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	

A'	D'	G'	I'	K'	M'	Δ	Σ
0	-1	-1	-1	-1	-1	-5	12
0	-1	-1	-1	-1	-1	-5	7
+1	-1	-1	-1	-1	0	+1	2
+1	+1	+1	+1	0	0	+5	3
0	+1	+1	-1	-1	-1	+1	8
0	+1	+1	-1	-1	0	-3	6
+1	-1	-1	-1	-1	0	-3	4
+1	-1	-1	-1	+1	0	+1	4
+1	+1	+1	+1	0	0	+5	9
0	-1	+1	+1	-1	-1	-5	10
0	-1	-1	-1	-1	-1	-5	5
+1	+1	+1	+1	0	0	+5	5
+1	+1	+1	+1	0	0	+5	10
+1	+1	+1	-1	-1	-1	0	10
+1	-1	-1	-1	-1	0	-4	6
+1	-1	-1	+1	0	0	-1	5
+1	-1	+1	+1	0	0	+1	6
+1	+1	+1	+1	-1	-1	+2	8
+1	-1	+1	+1	-1	0	0	8
+1	+1	-1	-1	-1	0	-2	6
+1	-1	+1	-1	-1	0	-1	5
+1	-1	+1	-1	0	0	+1	6
+1	-1	+1	-1	0	0	+1	7
+1	-1	+1	-1	+1	-1	0	7
+1	-1	+1	-1	+1	-1	0	7

A'	D'	G'	I'	K'	M'	Δ	Σ
0	-1	-1	-1	-1	-1	-5	12
0	-1	-1	-1	-1	-1	-5	7
+1	+1	+1	+1	0	0	+5	2
0	+1	+1	+1	-1	-1	+5	3
0	+1	+1	+1	-1	-1	+5	8
0	-1	-1	-1	-1	-1	-3	9
+1	-1	-1	-1	-1	-1	-3	6
+1	-1	-1	-1	0	0	-3	3
+1	+1	+1	+1	0	0	+5	4
0	-1	-1	-1	-1	-1	-5	9
0	-1	-1	-1	-1	-1	-5	10
+1	+1	+1	+1	0	0	+5	5
+1	+1	+1	+1	-1	-1	+5	5
+1	+1	+1	+1	-1	-1	+5	10
+1	-1	-1	-1	-1	-1	-4	6
+1	-1	-1	-1	0	0	-1	5
+1	-1	-1	-1	0	0	-1	6
+1	+1	+1	+1	-1	-1	+2	8
+1	-1	-1	-1	-1	-1	-2	6
+1	+1	+1	+1	-1	-1	-1	5
+1	-1	-1	-1	0	0	+1	6
+1	-1	-1	-1	0	0	+1	7
+1	-1	-1	-1	-1	-1	0	7
+1	-1	-1	-1	-1	-1	0	7

Рис. 2.50. Процесс поиска эталонного кода при его прохождении через сдвиговый регистр RG1 (рис. 2.49).  
 × — произвольное состояние разряда

Так как  $A=0$ ,  $D=1$ ,  $G=0$ ,  $I=1$ ,  $K=0$ ,  $M=1$ , то операнды для сложения (с помощью первого сумматора) примут следующие значения:  $A'=0$ ,  $D'=-1$ ,  $G'=-1$ ,  $I'=-1$ ,  $K'=-1$ ,  $M'=-1$ ; их сумма  $\Delta=-5$ . Новое значение  $\Sigma$  образуется алгебраическим сложением числа 12 с числом  $-5$ , в результате в момент поступления отрицательного фронта сигнала CLK в регистре RG2 фиксируется число 7, что и требуется.

После второго сдвига информации в регистре RG1 (строка 3)  $\Delta=-5$ ,  $\Sigma=7-5=2$  (совпадение кодов лучшее, но еще не достаточное для завершения работы), триггер ТТ остается в состоянии лог. 0. Далее процесс продолжается, и когда исходный входной код полностью вдвигается в разряды  $A-L$  регистра RG1 (строка 13), то формируется нулевое значение  $\Sigma$ , срабатывает компаратор, триггер ТТ устанавливается в 1 и поиск считается завершенным. На рис. 2.50 для наглядности процесс сдвига продолжен до полного вытеснения эталонного кода из регистра RG1.

Вместо цепи из последовательно соединенных сумматоров можно применить ПЗУ или ПЛМ. Если число входов сумматора 1 велико, то можно заменить его пирамидальной структурой из более простых сумматоров или опрашивать эти входы последовательно либо группами, используя мультиплексоры. Чтобы работать с разными эталонными кодами, можно предусмотреть возможность программного установления связей между регистром RG1 и сумматором 1.

## 2.22. Двумерная решетчатая структура для точного или приближенного распознавания символического сообщения [42]

Чтобы понять идею предложенного решения, рассмотрим его упрощенную «одномерную» модель (рис. 2.51). В исходном состоянии в регистрах RG хранятся искомые символы, размещенные в нужном порядке: A, B, C, D. После снятия сигнала S/R начальной установки, т. е. после его перехода в нуль, крайний левый D-триггер установлен в единицу, остальные — в нуль, устройство готово к выявлению цепочки символов ABCD в проходящем потоке данных.

Предположим, что именно эта цепочка подается на вход устройства. В течение некоторого периода  $Y=A$ , на выходе левого компаратора сформирован сигнал совпадения (лог. 1),  $L1=1$ . С поступлением первого положительного фронта синхросигнала С сигнал  $L1=1$  переписывается во второй D-триггер, а первый устанавливается в нуль, так как его D-вход соединен с шиной лог. 0. Затем наступает период, когда  $Y=B$ ,  $L2=1$ . При поступлении второго положительного фронта синхросигнала С сигнал лог. 1 перемещается в третий триггер и т. д. В конечном счете формируется сигнал опознавания  $F=1$ .

Как видим, каждое совпадение способствует продвижению сигнала лог. 1 в сдвиговом регистре. Любое несовпадение поступившего символа с ожидаемым препятствует такому продвижению, в результате сдвиговый регистр устанавливается в нуль, сигнал опознавания не формируется. Таким образом, рассмотренное устройство может распознавать лишь точные совпадения поступающего кода с эталоном, что ограничивает область его применения.

На практике задача ставится шире — необходимо отыскать если не точное сочетание символов, то близкое к заданному. Это связано с возможностью появления опечаток в анализируемом тексте, с действием помех в канале связи и иными факторами. Приведем примеры искажений кодов.

1. Пропадание символа: вместо кода A B C D принят код A B D.
2. Замена правильного символа неправильным: вместо кода A B C D принят код A B C̄ D, где C̄ — символ, отличный от C.
3. Вставка чужеродного символа: вместо кода A B C D принят код A B X C D, где X — некоторый посторонний символ, «вклинившийся» в сообщение.

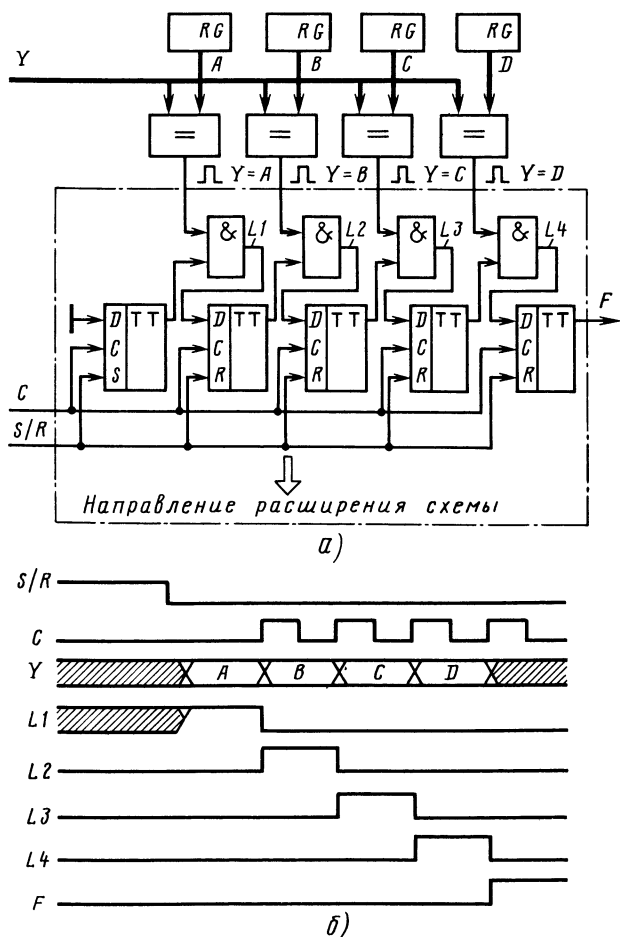


Рис. 2.51. Устройство для распознавания цепочки символов ABCD (а) и временные диаграммы (б) его работы

Чтобы более точно судить о степени несовпадения принятого и искомого сообщений, воспользуемся понятием дистанции между ними. В табл. 2.3 приведены примеры кодов, отстоящих от эталонного на разные дистанции.

Дистанция равна нулю при точном совпадении, единице — при пропадании, замене или вставке одного символа (см. приведенные ранее примеры 1—3), двум, трем и т. д. — при обнаружении более существенных отличий.

Предлагаемое решение позволяет отображать дистанцию между эталонным и принятым кодами. Для этого часть схемы, выделенная на рис. 2.51 штриховой линией, наращивается «во втором измерении» в направлении, показанном стрелкой, так что образуется матричная структура (рис. 2.52).

Входы  $C$  синхронизации всех D-триггеров объединены (не показаны для упрощения рисунка), поэтому вся структура представляет собой единый «двумерный» сдвиговый регистр, в котором изменения состояний триггеров

Таблица 2.3

Дистанция	Примеры принятых кодов
0	A B C D
1	A B D, A B $\bar{C}$ D, A B X C D
2	A D, A $\bar{B}$ C D, A X B X C D, A C X D, B $\bar{C}$ D, A $\bar{B}$ X C D
3	A, $\bar{A}$ B C D, A X B X C X D, A D X, A $\bar{D}$ , A X $\bar{B}$ X C D

возможны только в моменты поступления положительных фронтов сигнала синхронизации или при формировании сигнала начальной установки.

Схема приводится в исходное состояние сигналом  $Q=1$ , который поступает на вход установки единицы левого верхнего триггера, а также на входы установки нуля (не показаны) остальных триггеров, за исключением помеченных знаками \* (лежащих на главной диагонали матричной структуры). Сигнал лог. 1 с выхода левого верхнего триггера асинхронно распространяется вдоль главной диагонали и переводит триггеры, ранее установленные в нуль, в состояние лог. 1 воздействием по входу S. Это воздействие прекращается сразу после перехода триггера в единицу, так как нулевой сигнал с его инверсного плеча закрывает двухвходовый элемент И по верхнему входу.

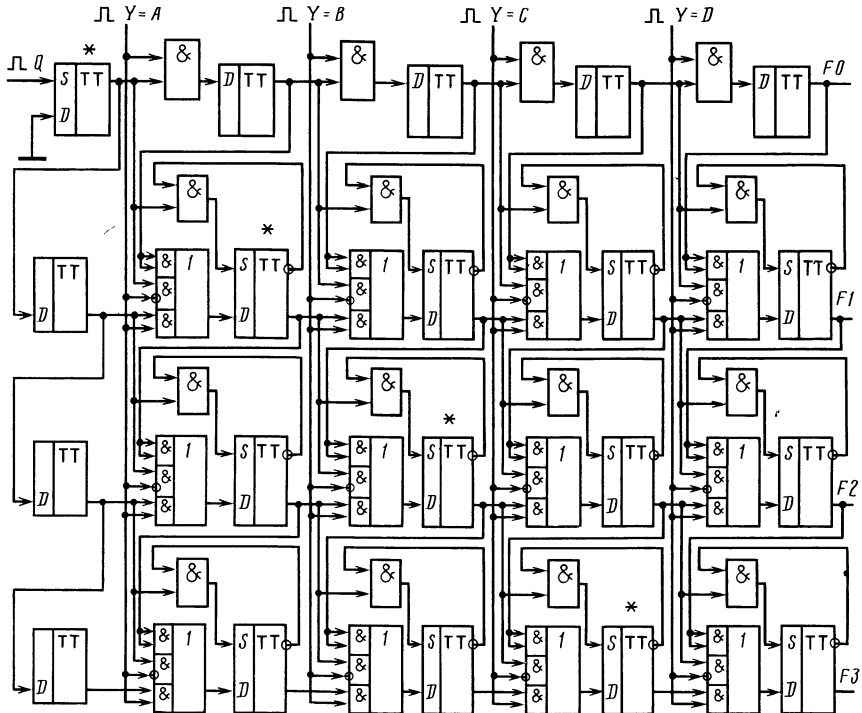


Рис. 2.52. Матричная структура для распознавания символического сообщения

Забегая вперед, отметим, что аналогичные процессы заполнения других диагоналей сигналами лог. 1 наблюдаются при дальнейшей работе устройства—стоит установить какой-либо триггер в единицу, как все последующие триггеры соответствующей диагонали устанавливаются в это же состояние бегущим по входам  $S$  кратковременным положительным импульсом. Если двухвходовый элемент И, на нижний вход которого поступает сигнал лог. 1 с выхода предыдущего триггера данной диагонали, закрыт сигналом лог. 0 с инверсного плеча триггера, то это означает, что соответствующий триггер (и все последующие в диагонали) уже установлен в состояние лог. 1.

Возвращаясь к моменту начальной установки структуры, приходим к заключению, что триггеры, помеченные знаками \*, находятся в состоянии лог. 1, остальные—в состоянии лог. 0. Для наглядности будем использовать матричное отображение сигналов в узлах структуры. В данном случае получим следующее отображение:

$$\text{Исходное состояние: } \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.1)$$

Работа схемы (рис. 2.52) сопровождается «эволюцией» единичных сигналов главной диагонали матрицы (2.1)—эти сигналы причудливым образом перемещаются по строкам, диагоналям и столбцам матрицы с поступлением положительных фронтов синхроимпульсов. При этом верхняя единица из числа присутствующих на выходах  $F_0$ — $F_3$  (если они вообще присутствуют) отражает текущую дистанцию между принятым и эталонным кодами. В частности, при  $F_0=1$  дистанция равна нулю, а при  $F_0=F_1=F_2=F_3=0$ —четырем или более.

Чтобы в какой-то мере облегчить понимание работы структуры, воспользуемся более абстрактным ее представлением (рис. 2.53). В каждый узел (за исключением узла 1, 1) направлены одна или четыре стрелки, что соответствует одному или четырем альтернативным путям приема единичного сигнала.

Символы  $A$ ,  $B$ ,  $C$  и  $D$  отображают выполнение условий  $Y=A$ ,  $Y=B$ ,  $Y=C$  и  $Y=D$ ; соответствующие горизонтальные стрелки показывают направление передачи единичных сигналов между соседними столбцами матрицы. Например, при выполнении условия  $Y=C$  и поступлении положительного фронта синхросигнала единичные сигналы из узлов 1,3; 2,3; 3,3; 4,3 переписываются в соответствующие узлы 1,4; 2,4; 3,4 и 4,4 (далее срабатывают рассмотренные ранее схемы распространения единичных сигналов по диагоналям, см. диагональные стрелки с символами 1).

Символы  $\bar{A}$ ,  $\bar{B}$ ,  $\bar{C}$  и  $\bar{D}$  отображают условия несравнения символа  $Y$  с соответствующими символами  $A$ ,  $B$ ,  $C$  и  $D$ . Если, например,  $Y=E$ , то одновременно открываются все пути переписи сигналов по диагоналям

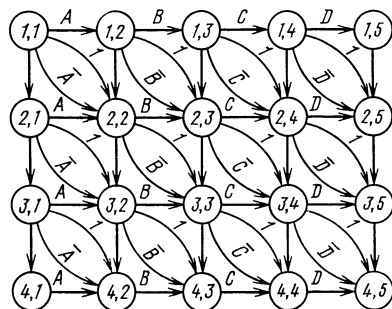


Рис. 2.53. Представление структуры на рис. 2.52 в виде графа

при поступлении положительного фронта синхроимпульса. Если  $Y=B$ , то из этих путей исключаются обозначенные символами В и т. д. Вертикальные стрелки, не помеченные какими-либо символами, соответствуют безусловной передаче сигналов лог. 1 в соответствующих направлениях при поступлении положительного фронта синхроимпульса.

Предположим, что на входы устройства последовательно поступают символы А, В, С и D. В этом случае исходная матрица данных (2.1) последовательно преобразуется в матрицы (2.2)—(2.5):

$$Y=A: \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}; \quad (2.2)$$

$$Y=B: \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}; \quad (2.3)$$

$$Y=C: \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}; \quad (2.4)$$

$$Y=D: \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (2.5)$$

На первом шаге (2.1), (2.2) при поступлении первого положительного фронта сигнала синхронизации структуры диагональ из логических единиц (1,1; 2,2; 3,3; 4,4) «раздвигается» в обе стороны и частично «стирается». Точнее, благодаря выполнению условия  $Y=A$  сигнал лог. 1 передается из узла 1,1 в узел 1,2, а узел 1,1 переводится в состояние лог. 0, так как D-вход верхнего левого триггера подключен к шине нулевого потенциала. В то же время сигнал лог. 1 из узла 1,1 безусловно переписывается в узел 2,1. Сигналы лог. 1 из узлов 1,2 и 2,1 асинхронно распространяются по соответствующим диагоналям матрицы, так что узлы 3,2; 4,3; 2,3; 3,4 и 4,5 переходят в единичные состояния.

Узел 2,2 переходит в состояние 0, так как узлы 1,2 и 2,1 исходно находились в состоянии 0, а узел 1,1 не способен передавать в узел 2,2 «старую» единицу, поскольку выполнено условие  $Y=A$ , препятствующее такой передаче. Узел 3,3 принимает «старую» единицу из узла 2,2, так как выполнено условие  $Y=B$ . Узел 4,4 остается в состоянии 1, так как узел 3,3 передает в него «старую» единицу благодаря выполнению условия  $Y=C$ .

Как отмечалось, дистанция между принятым и эталонным кодами соответствует уровню подъема верхней единицы по выходам  $F3-F0$ . После приема символа А дистанция между ним и эталоном (А В С D) равна трем, что соответствует сигналу  $F3=1$ .

После приема второго синхроимпульса, положительный фронт которого подтверждает выполнение условия  $Y=B$ , матрица (2.2) преобразуется в матрицу (2.3), сигнал лог. 1 перемещается на более высокий уровень ( $F2$ ), что соответствует уменьшению дистанции между кодами до двух (сравните коды А В и А В С D). Последующее преобразование (матрица (2.3)—матрица (2.4)) уменьшает дистанцию до единицы (коды А В С—А В С D), и, наконец, при переходе к матрице (2.5) регистрируется точное совпадение кода с эталоном А В С D,  $F0=1$ , дистанция равна нулю.

Противоположная ситуация, при которой не зарегистрировано ни одного совпадения, отражена матрицами (2.6)—(2.10):

$$\text{Исходное состояние: } \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}; \quad (2.6) \quad Y=X: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}; \quad (2.7)$$

$$Y=X: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}; \quad (2.8) \quad Y=X: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}; \quad (2.9)$$

$$Y=X: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (2.10)$$

Рассмотрим, наконец, ситуацию, при которой вместо кода А В С D принимается код А С Х D (матрицы (2.11) — (2.15)):

$$\text{Исходное состояние: } \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}; \quad (2.11) \quad Y=A: \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}; \quad (2.12)$$

$$Y=C: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}; \quad (2.13) \quad Y=X: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}; \quad (2.14)$$

$$Y=D: \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (2.15)$$

В окончательном состоянии структуры верхняя единица присутствует на выходе F2, что соответствует дистанции между кодами А В С D и А С Х D, равной двум. Для выделения верхней единицы к выходам структуры можно подключить комбинационный приоритетный шифратор.

## 2.23. Схема контроля дешифратора [95]

Для контроля дешифратора или иного устройства, на выходах которого должен присутствовать один, и только один сигнал лог. 1, можно применить схему, показанную на рис. 2.54. Входной блок (рис. 2.55, а) анализирует пару сигналов с выходов проверяемого устройства и при обнаружении двух единиц формирует признак ошибки E=1, который распространяется через промежуточные и выходной блоки. В результате вырабатывается суммарный сигнал ошибки ERR=1.

Если единичные сигналы поступают на разные входные блоки, то соответствующие сигналы S, продвигаясь по пирамидальной структуре, встречаются в промежуточных или (и) выходном блоках (см. рис. 2.55, а) анализирует пару сигналов с выходов проверяемого устройства и при обнаружении двух единиц формирует признак ошибки E=1, который распространяется через промежуточные и выходной блоки. В результате вырабатывается суммарный сигнал ошибки ERR=1. При отсутствии сигналов лог. 1 на выходах проверяемого устройства входные и промежуточные блоки формируют сигналы E=0, S=0, поэтому ERR=1. Таким образом, условие ERR=0 выполняется при наличии только одного сигнала лог. 1 на выходах проверяемого устройства.

Схема, приведенная на рис. 2.56, инвертирует сигналы в каждом каскаде; ее составные части показаны на рис. 2.57. Такое решение может быть полезным, когда инвертирующие логические элементы имеют большее быстродействие, чем не инвертирующие. Если пирамидальная структура формирует на входах выходного блока сигналы E и S в прямой фазе, то этот блок может быть аналогичен рассмотренному ранее (см. рис. 2.54); если сигналы представлены в обратной фазе, то выходной блок соответственно дорабатывается (например, на его входах устанавливаются четыре элемента НЕ). Возможны иные решения, учитывающие особенности применяемой элементной базы.

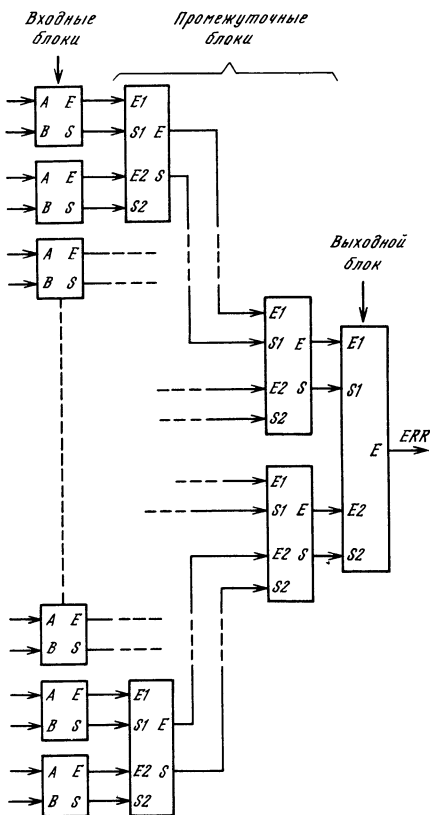


Рис. 2.54. Схема контроля дешифратора (первый вариант)

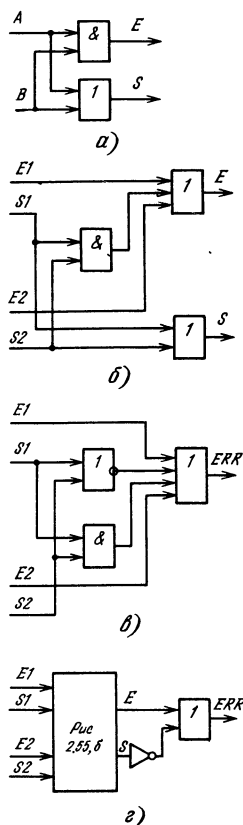


Рис. 2.55. Варианты построения входного (а), промежуточного (б) и выходного (в, г) блоков (рис. 2.54)

В схеме, приведенной на рис. 2.58, каждый блок принимает сигналы с трех направлений. Входной блок формирует сигналы E и S в соответствии со следующими условиями:  $E = AB + BC + AC$ ;  $S = A + B + C$ . Промежуточный блок:  $E = E1 + E2 + E3 + S1S2 + S2S3 + S1S3$ ;  $S = S1 + S2 + S3$ . Выходной блок:  $ERR = E1 + E2 + E3 + S1S2 + S2S3 + S1S3 + S1S2S3$ .

## 2.24. Самоконтролируемая схема проверки нечетности числа единиц в передаваемых кодах [96]

Для контроля правильности обмена данными между различными устройствами микроЭВМ (такими как процессор, память, контроллер) в нее обычно вводят дополнительные информационные и управляющие линии и сигналы. Например, добавочный информационный сигнал может

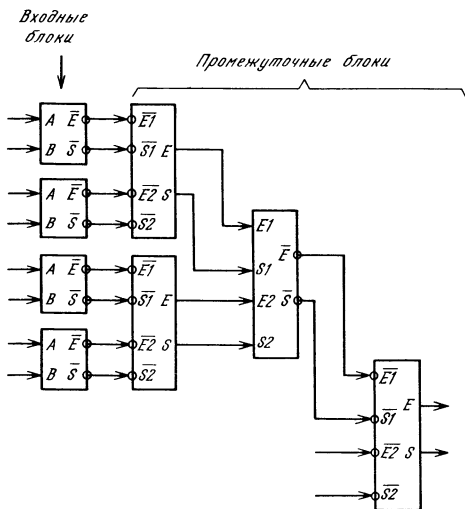


Рис. 2.56. Фрагмент схемы контроля дешифратора (второй вариант). Выходной блок не показан

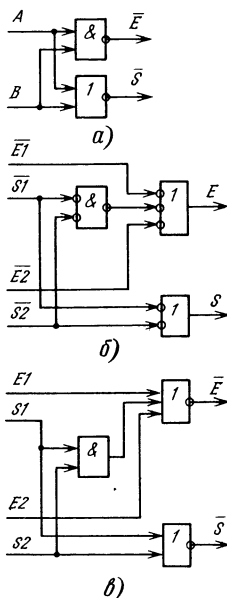


Рис. 2.57. Варианты построения входного (а) и промежуточных (б, в) блоков (рис. 2.56)

служить контрольным разрядом нечетности передаваемого кода, а управляющий—признаком ошибки, обнаруженной при проверке нечетности принятого кода.

Проблема состоит в следующем. В результате отказа схемы контроля принятого кода, когда по тем или иным причинам сигнал ошибки не может формироваться, неправильный обмен данными не будет зарегистрирован, что не всегда допустимо. Поэтому в процессе выполнения полезной работы по пересылке данных желательно периодически (как можно чаще) проверять исправность самой схемы контроля.

Схема, приведенная на рис. 2.59, позволяет контролировать пересылку данных D и обнаружить отказы в цепи формирования сигнала ошибки ERR. Рассмотрим процесс взаимодействия передатчика и приемника (например, процессора и памяти).

1. Передатчик устанавливает в информационной (или адресной) шине код D. Сумматор по модулю два D1 с инверсным выходом формирует контрольный разряд PARD, дополняющий число единиц в передаваемом 9-разрядном коде (D, PARD) до нечетного.

2. Приемник проверяет общую нечетность числа единиц в полученном 9-разрядном коде. Для этого используются сумматоры по модулю два D2 и D3. При правильной передаче на верхнем входе элемента И D4 сформирован сигнал лог. 1, при неправильной (когда имеется одиночная ошибка в коде)—сигнал лог. 0.

3. После окончания переходных процессов на выходе элемента D3 передатчик формирует положительный импульс (строб) STB, подтверждающий истинность данных D. По фронту этого импульса передатчик запоминает старое значение сигнала ERR в некотором триггере (не показан), а приемник опрашивает состояние выхода элемента D3. Если ошибки нет, то на выходе

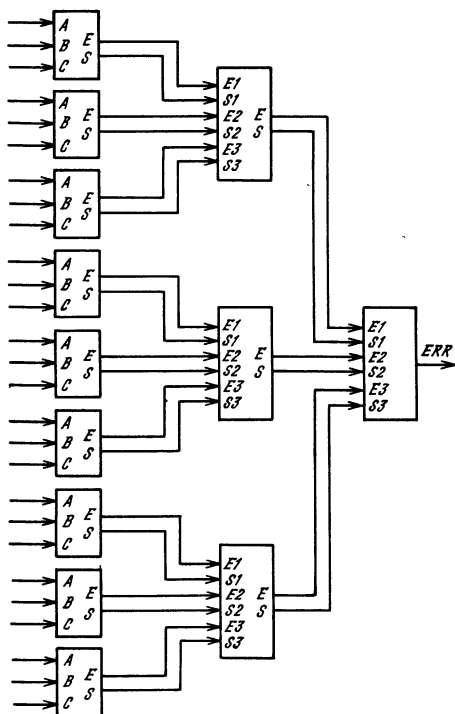


Рис. 2.58. Схема контроля дешифратора (третий вариант)

элемента  $D4$  формируется положительный фронт сигнала, триггер  $D5$ , включенный в режиме одноразрядного счетчика, меняет состояние. При наличии ошибки элемент И  $D4$  закрыт по верхнему входу, поэтому сигнал  $ERR$  не изменяется.

4. Передатчик сравнивает старое значение сигнала  $ERR$  с новым и в случае их совпадения фиксирует ошибку.

Если в результате неправильной работы схемы контроля сигнал  $ERR$  останется неизменным хотя бы в одном сеансе связи, то этот факт будет оперативно зарегистрирован передатчиком так же, как и в рассмотренном примере.

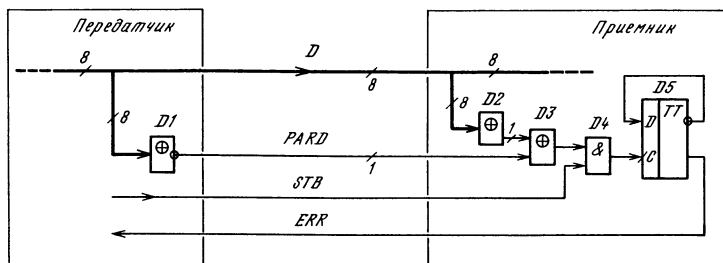


Рис. 2.59. Схема передачи данных с контролем по нечетности.

В отличие от традиционной схемы, сигнал ошибки  $ERR$  меняет значение в каждом сеансе связи передатчика с приемником

## 2.25. Защита от неправильного подключения плат контроллеров к системной плате микроЭВМ [97]

Системная (основная) плата микроЭВМ обычно содержит процессор, ОЗУ, ПЗУ и контроллеры устройств ввода-вывода (минимальная конфигурация). Для расширения функциональных возможностей микроЭВМ в системной плате предусматривают ряд соединителей, в которые можно устанавливать дополнительные платы памяти, контроллеров и иных устройств. Обычно одноименные контакты всех соединителей включены параллельно, поэтому все посадочные места для установки дополнительных плат равноправны — пользователь может устанавливать любую плату на любое место.

Чтобы уменьшить аппаратные затраты при расширении функциональных возможностей микроЭВМ, посадочные места или по крайней мере часть из них специализируют для установки вполне определенных устройств. Например, один из соединителей может использоваться только для расширения накопителя динамического ОЗУ; тогда через этот соединитель передаются соответствующие сигналы, «понятные» только дополнительному накопителю: мультиплексированный адрес, данные, признак записи WR, сигналы выбора строки RAS и столбца CAS накопителя. Другой соединитель системной платы может использоваться только для установки кэш-памяти и т. д.

Если такие соединители конструктивно одинаковы, то неправильная установка в них плат расширения (на чужие посадочные места) может привести к электрическим перегрузкам цепей, ошибочно соединенных между собой, и, как следствие этого, — к выходу из строя элементов и узлов

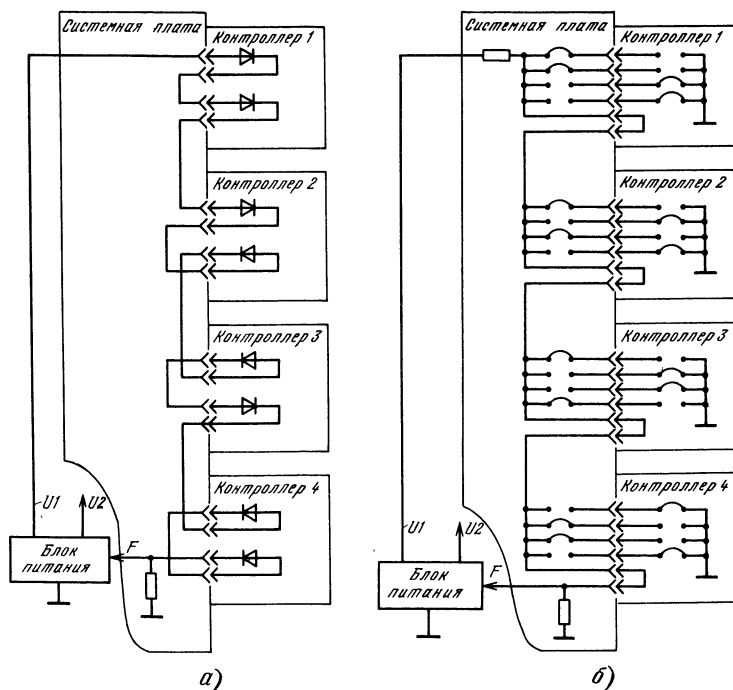


Рис. 2.60. Варианты построения схемы запрета включения напряжения питания U2 микроЭВМ при неправильном порядке установки контроллеров в соединители

микроЭВМ. В предлагаемом решении напряжение питания подается на микроЭВМ только при правильном размещении плат расширения (каждая — на своем посадочном месте).

В примере, приведенном на рис. 2.60, *а*, блок питания формирует «дежурное» положительное напряжение  $U_1$ , которое вызывает протекание тока через диоды, размещенные в платах контроллеров 1—4. При правильном подключении плат (как показано на рисунке) на входе управления блока питания сформировано положительное напряжение высокого уровня ( $F=1$ ), поэтому блок выдает напряжение  $U_2$  питания системной платы и плат контроллеров.

Если хотя бы две платы поменять местами, т. е. установить их в «чужие» соединители, то согласное включение диодов на том или ином участке заменяется встречным, ток через цепь из диодов практически отсутствует,  $F=0$ , блок питания не формирует основное напряжение питания  $U_2$  (напряжение  $U_1$  может выдаваться). Если число диодов в каждой плате равно  $n$ , то в системе могут присутствовать до  $2^n$  плат (по числу вариантов распайки диодов).

В схеме (рис. 2.60, *б*) при правильной установке плат перемычки со стороны контроллеров не препятствуют поддержанию напряжения высокого уровня на управляющем входе блока питания ( $F=1$ ), поэтому микроЭВМ находится в работоспособном состоянии. Если поменять местами хотя бы две платы, то управляющий вход блока питания подключится к шине нулевого потенциала ( $F=0$ ), этот блок не формирует напряжение  $U_2$ .

В данном примере максимальное число плат равно числу сочетаний из четырех по два:  $C_4^2=6$ . Перемычки на системной плате показаны лишь для пояснения идеи; их следует заменить обычными печатными проводниками.

## Глава 3

### Адресация, обмен данными

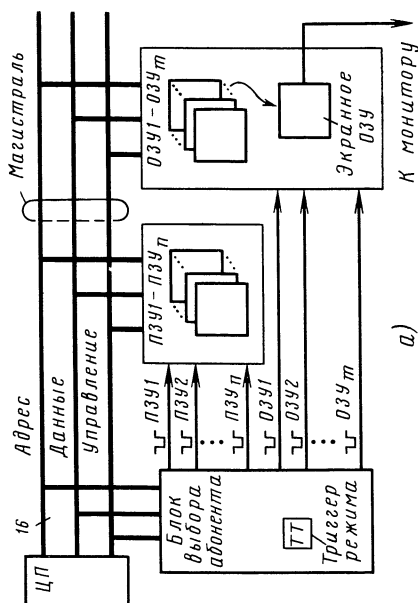
#### 3.1. Способ расширения памяти микроЭВМ [43]

В микроЭВМ обычно имеется ряд блоков ПЗУ и ОЗУ, причем один или несколько блоков ОЗУ выделены для накопления информации, отображаемой на экране монитора, как показано на рис. 3.1, *а*. Блок выбора абонента представляет собой дешифратор адреса (триггер режима пока не принимается во внимание), который выделяет блокам ПЗУ и ОЗУ необходимые области адресного пространства в диапазоне  $0000\text{—}FFFF_{16}$  (разрядность адресной шины магистрали в данном примере равна 16 бит).

При обращении ЦП по адресам, лежащим в диапазоне  $0000\text{—}3FFF$ , блок выбора абонента формирует сигнал разрешения работы ПЗУ1, при попадании адреса в диапазон  $4000\text{—}7FFF$  вступает в действие ПЗУ2 и т. д. Экранное ОЗУ (ОЗУ3) с точки зрения процессора отличается от ОЗУ1 и ОЗУ2 только задействованными адресами и доступно со стороны магистрали как по чтению, так и по записи. Это ОЗУ имеет дополнительный канал считывания, по которому информация циклически передается в монитор для отображения на экране электронно-лучевой трубки.

Процессор обычно использует экранное ОЗУ только для записи в него отображаемой информации. Однако в некоторых применениях микроЭВМ желательнее также по мере необходимости считывать содержимое этого ОЗУ, что эквивалентно «просмотру экрана изнутри». Если такая возможность имеется, то процессор может, например, отыскать на экране нужный фрагмент изображения, проанализировать и изменить его цвет и т. п.

В примере, приведенном на рис. 3.1, *б*, все адресное пространство заполнено, поэтому для подключения к микроЭВМ дополнительных блоков памяти вытесняются некоторые из имеющихся. Далее рассмотрены пять вариантов размещения блоков памяти в адресном пространстве (рис. 3.2—3.6), причем,



0000	ПЗУ1
3FFF	
4000	ПЗУ2
7FFF	
8000	ОЗУ1
BFFF	
C000	ОЗУ2
DFFF	ОЗУ3
E000	ОЗУ3 (экранное)
FFFF	

Рис. 3.1. Фрагмент структуры типовой микроЭВМ (а) и пример распределения ее адресного пространства (б).  
Дополнительно введен триггер ТТ и усовершенствован дешифратор в блоке выбора абонента (см. варианты схем на рис. 3.2—3.6)

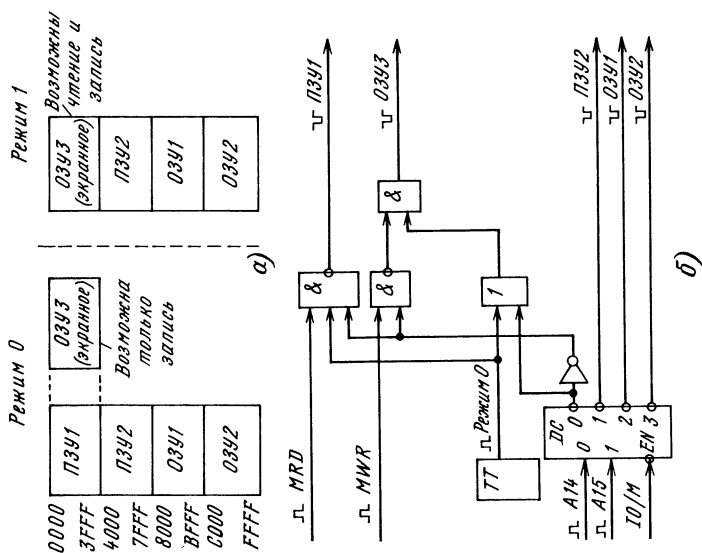


Рис. 3.2. Первый вариант размещения блоков памяти в адресном пространстве (а) и соответствующая ему схема блока выбора абонента (б) (см. рис. 3.1)

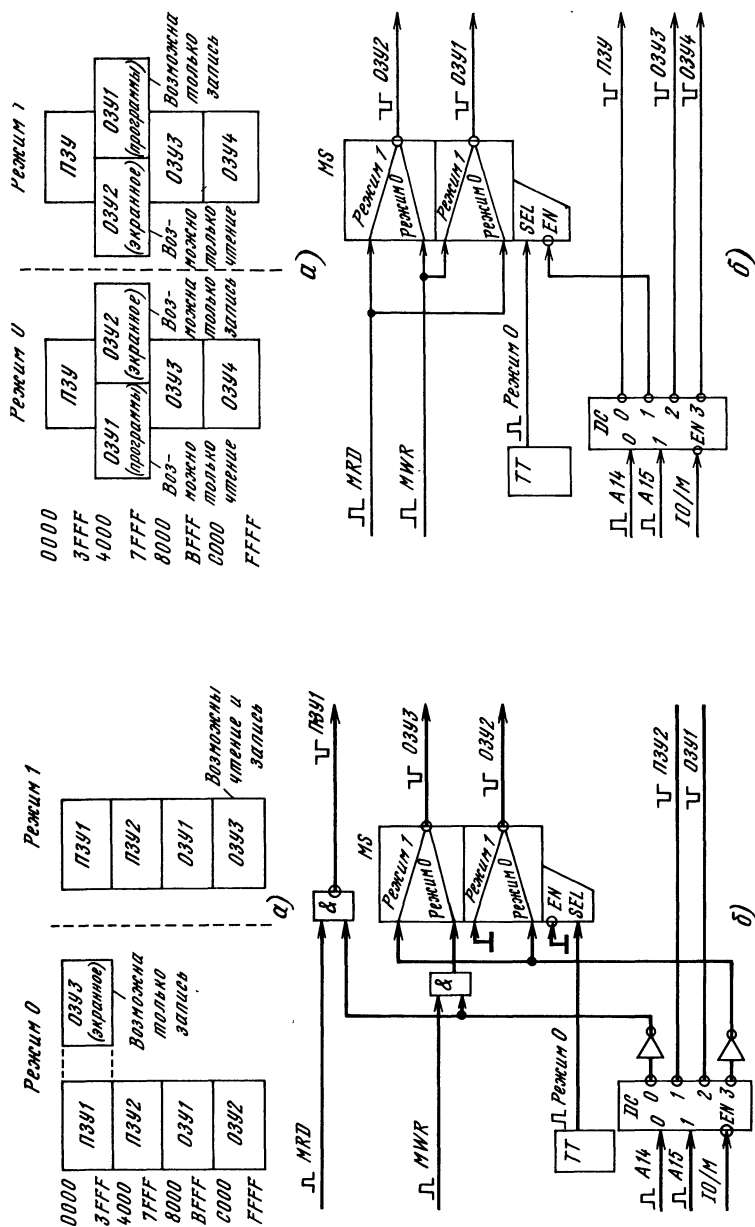


Рис. 3.3. Второй вариант размещения блоков памяти в адресном пространстве (а) и соответствующая ему схема блока выбора абонента (б) (см. рис. 3.1)

Рис. 3.4. Третий вариант размещения блоков памяти в адресном пространстве (а) и соответствующая ему схема блока выбора абонента (б) (см. рис. 3.1)

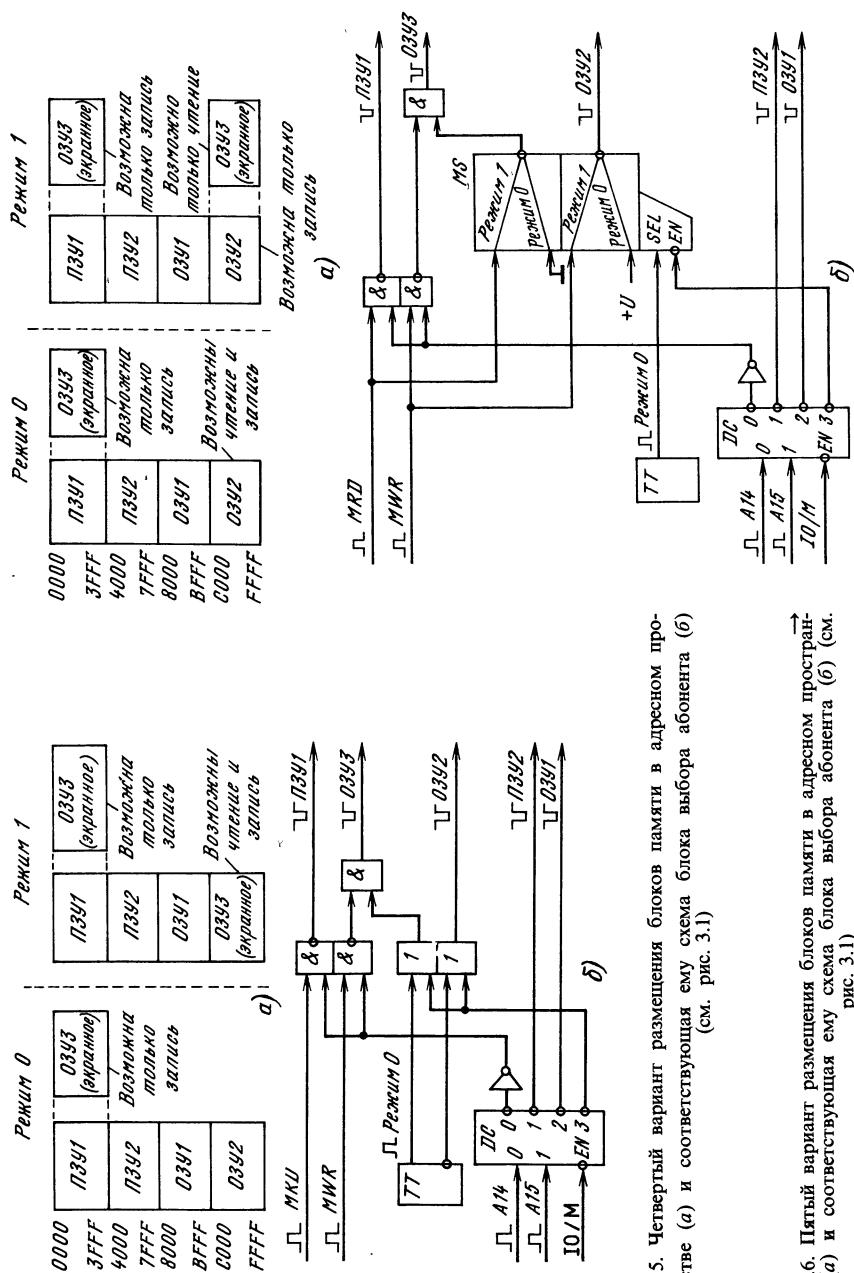


Рис. 3.5. Четвертый вариант размещения блоков памяти в адресном пространстве (а) и соответствующая ему схема блока выбора абонента (б) (см. рис. 3.1)

Рис. 3.6. Пятый вариант размещения блоков памяти в адресном пространстве (а) и соответствующая ему схема блока выбора абонента (б) (см. рис. 3.1)

каждый вариант предусматривает работу микроЭВМ в двух режимах: 0 и 1. Эти режимы задаются состоянием программно-доступного триггера ТТ, введенного в блок выбора абонента. Для адресации этого триггера используется адресное пространство устройств ввода—вывода, изолированное от показанного на рисунках адресного пространства памяти, в котором развиваются описанные далее события.

Первый вариант (режим 0, рис. 3.2, а) соответствует размещению экранного ОЗУ (ОЗУ3) в той же области 0000—3FFF адресного пространства, что и ПЗУ1. Если процессор выполняет операцию чтения из ячейки, принадлежащей данной области, то активизируется ПЗУ1, а ОЗУ3 не выбирается (выдача информации из ОЗУ3 на экран монитора производится постоянно, непрерывно). Если процессор выполняет операцию записи по адресу, принадлежащему области 0000—3FFF, то активизируется соответствующая ячейка ОЗУ3 (она принимает слово данных из процессора), а ПЗУ1 и другие блоки памяти остаются в пассивном состоянии и отключены от магистрали. В данном режиме, таким образом, экранная память доступна процессору только по записи, а считывать ее ячейки он не может.

Если процессор желает проанализировать содержимое экранной памяти, то он, выполняя команду вывода информации во внешнее устройство с заданным адресом, переводит триггер ТТ в состояние, соответствующее режиму 1. После смены режима ОЗУ3 занимает тот же диапазон адресов, что и прежде, однако теперь возможна не только запись, но и считывание кодов из ячеек этого ОЗУ. С другой стороны, ПЗУ1 исчезло из адресного пространства, т. е. доступ к нему невозможен. Поэтому программы работы процессора в режиме 1 не должны храниться в этом ПЗУ.

Схема, приведенная на рис. 3.2, б, работает следующим образом. В исходном состоянии на инверсном входе EN разрешения работы дешифратора DC присутствует сигнал запрета (лог. 1), дешифратор закрыт, на его выходах сформированы сигналы лог. 1, элементы И—НЕ закрыты сигналами лог. 0, поступающими на их верхние и нижние входы, на входы элемента И поданы сигналы лог. 1, сигналы выбора абонентов (ОЗУ1—ОЗУ3, ПЗУ1, ПЗУ2) отсутствуют, т. е. равны лог. 1. При обращении процессора к одной из ячеек адресного пространства памяти он выдает адрес, данные (при записи), признак IO/M=0 и в зависимости от типа обращения—сигнал MRD (чтение памяти) либо MWR (запись в память).

Дешифратор DC открывается по управляющему входу EN и в зависимости от комбинации сигналов в двух старших разрядах A15 и A14 адреса формирует сигнал лог. 0 на одном из выходов 0—3. Сигналы лог. 0 с выходов 1—3 активизируют ПЗУ2, ОЗУ1 и ОЗУ2 независимо от состояния триггера ТТ. Сигнал лог. 0 на выходе 0 дешифратора DC формируется при обращении процессора в область 0000—3FFF. В режиме 0 триггер ТТ установлен в 1, в результате сигналы MRD и MWR проходят на выходы выбора ПЗУ1 и ОЗУ3, что и требуется. В режиме 1 триггер ТТ установлен в 0, верхний элемент И—НЕ постоянно закрыт сигналом с выхода триггера, сигнал выбора ПЗУ1 не формируется. Сигнал лог. 0 с выхода 0 дешифратора DC проходит через элементы ИЛИ, И и служит для выбора ОЗУ3 как при чтении, так и при записи.

Как отмечалось, возможность чтения и записи информации в экранную память обеспечивается за счет вытеснения ПЗУ1 из адресного пространства (см. рис. 3.2). Если это недопустимо из-за потери актуальных для работы в режиме 1 программ, размещенных в этом ПЗУ, то можно вытеснить любой другой блок. В примере, приведенном на рис. 3.3, экранное ОЗУ (ОЗУ3) в режиме 1 занимает место ОЗУ2, а не ПЗУ1, как в только что описанном варианте размещения блоков памяти.

В схеме блока выбора абонента (см. рис. 3.3, б) использован инвертирующий мультиплексор MS, у которого на инверсный вход EN разрешения работы подан сигнал лог. 0 с шины нулевого потенциала, а на вход SEL выбора направления передачи информации—сигнал с выхода триггера ТТ.

В режимах 0 и 1 мультиплексор MS передает инвертированную информацию на выходы с нижних и верхних разрядных входов, как показано на рисунке.

В примере, приведенном на рис. 3.4, в ОЗУ1 размещены программы, а ОЗУ2 представляет собой экранную память. В режиме 0 (это основной режим) ОЗУ1 доступно только по чтению, а ОЗУ2, размещенное в той же области адресного пространства, доступно только по записи. При необходимости анализа отображаемой информации или (и) при загрузке ОЗУ1 процессор временно использует режим 1. В этом режиме ОЗУ1 и ОЗУ2 меняются ролями—экранное ОЗУ (ОЗУ2) оказывается доступным только по чтению, а ОЗУ программ (ОЗУ1)—только по записи. После выполнения вспомогательных операций процессор вновь возвращается в режим 0 и продолжает нормальную работу.

В примере, приведенном на рис. 3.5, режим 1 обеспечивает получение двух неравноценных копий ОЗУ3 (экранной памяти). Первая, как и в режиме 0, размещена в области 0000—3FFF и доступна только по записи, вторая, доступная как по чтению, так и по записи, образована за счет вытеснения ОЗУ2 из области C000—FFFF.

В примере, приведенном на рис. 3.6 (режим 1), также имеются две копии экранного ОЗУ (ОЗУ3), однако ОЗУ2 вытеснено лишь «наполовину»—к нему возможен доступ только по записи. Одна из копий ОЗУ3 (размещенная в области 0000—3FFF) доступна только по записи, а другая (из области C000—FFFF)—только по чтению.

### **3.2. Расширение ПЗУ без увеличения выделенного ему адресного пространства [44]**

На основной плате микроЭВМ имеются две розетки (рис. 3.7), рассчитанные на установку двух микросхем ПЗУ со структурой 16K×8 бит—по одной микросхеме на каждый байт 16-разрядного слова. Адресные ( $AS_i$ ) и управляющие (CE, OE) сигналы подаются на обе розетки параллельно, данные выдаются в разряды D0—D7 и D8—D15 шины данных. При обращении к ПЗУ управляющие сигналы CE и OE равны 0.

Задача состоит в расширении ПЗУ без изменения схемных решений, использованных в микроЭВМ. Иными словами, пользователю доступны только контакты двух розеток, из которых извлечены «штатные» микросхемы ПЗУ (со структурой 16K×8 бит каждая). В данном примере пользователь желает подключить к микроЭВМ две микросхемы ПЗУ (ПЗУ1 и ПЗУ2) со структурой 64K×8 бит каждая, т. е. увеличить первоначальный объем ПЗУ в 4 раза. Эти микросхемы имеют по 16 адресных входов, соединенных параллельно (на рисунке показаны только индивидуальные связи микросхемы ПЗУ2 с микроЭВМ, параллельные соединения между адресными и управляющими входами обеих микросхем подразумеваются). Поэтому необходимо расширить первоначальный 14-разрядный адрес до 16-разрядного. Для этого используется блок выбора страницы (рис. 3.8).

Микросхемы ПЗУ1 и ПЗУ2 можно представить содержащими 8 страниц по 8K байт каждая. В зависимости от комбинации сигналов в разрядах A13—A15 выбирается одна из 8 страниц; адрес ячейки на странице задается разрядами A0—A12. Из схемы рис. 3.8 следует, что при  $AS_{14}=0$  двухвходовые элементы И закрыты по нижним входам,  $A_{13}=A_{14}=A_{15}=0$ , выбирается нулевая страница. При  $AS_{14}=1$  элементы И транслируют на выходы содержимое регистра RG, в который предварительно загружен код номера страницы (от 000 до 111<sub>2</sub>).

Код номера страницы загружается в регистр RG при выполнении операции считывания некоторой ячейки. В разрядах  $AS_1$ — $AS_3$  адреса этой ячейки указывается номер страницы, с которой предстоит работать, в разрядах  $AS_4$ — $AS_{13}$  устанавливаются сигналы лог. 1, в разряде  $AS_{14}$ —лог. 0, при этом  $A_{13}=A_{14}=A_{15}=0$ . После окончания переходных процессов в шине адреса, с поступлением сигнала  $SEL=0$  на выходе 12-входового элемента И фор-

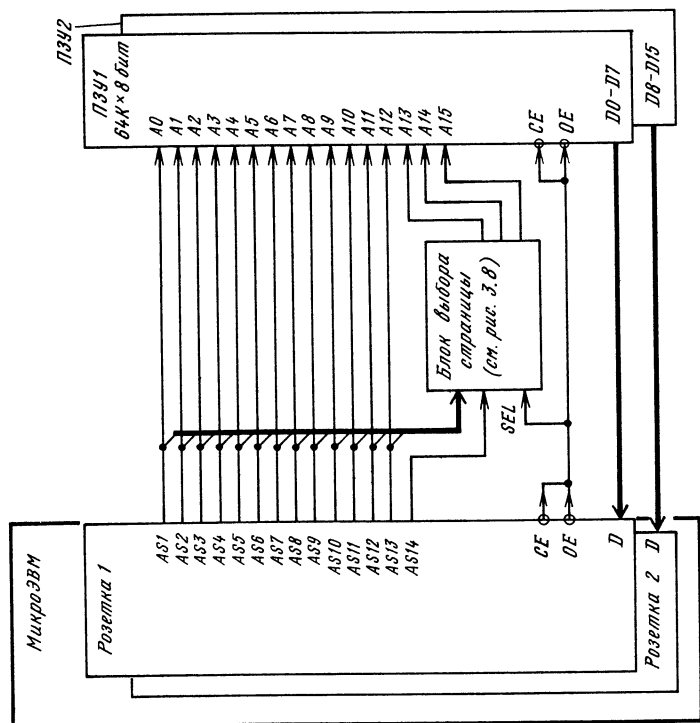


Рис. 3.7. Подключение двух ПЗУ со структурой 64K × 8 бит к контактам розеток, установленных на плате микроЭВМ и рассчитанных на применение ПЗУ со структурой 16K × 8 бит

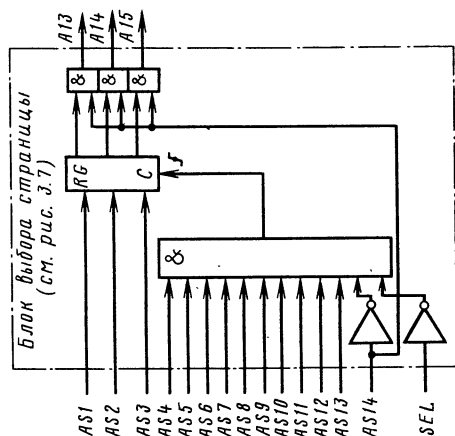


Рис. 3.8. Структура блока выбора страницы  
(см. рис. 3.7)

мируется сигнал лог. 1, по положительному фронту которого в регистре RG фиксируется код  $AS1—AS3$ . В качестве «побочного эффекта» (которому вряд ли можно найти достойное применение) при загрузке регистра RG из нулевых страниц ПЗУ1 и ПЗУ2 считывается содержимое соответствующих ячеек.

Таким образом, используя только режим считывания, можно настраивать регистр RG на выбор нужной страницы. После включения напряжения питания этот регистр может находиться в произвольном состоянии. Однако при обращении по адресу, принадлежащему первой половине адресного пространства, выделенного для штатного ПЗУ, когда  $AS14=0$ , однозначно выбирается некоторая подразумеваемая (в данном примере — нулевая) страница ПЗУ1 и ПЗУ2; в дальнейшем переход к иным страницам осуществляется программно, перезагрузкой содержимого регистра RG.

### 3.3. Адресация блоков памяти, имеющих разное быстродействие [45]

В микроЭВМ, показанной на рис. 3.9, блоки памяти 1—3 выполнены на основе микросхем D1—D3, имеющих разное быстродействие. Например, микросхемы D2 и D3 могут представлять собой статическое ОЗУ и ПЗУ с временем считывания информации, равным соответственно 40 и 400 нс.

При обращении к блоку памяти с номером  $i$  ( $i=1, 2, 3$ ) процессор формирует соответствующий сигнал выбора блока  $SEL_i=0$ . Этот сигнал поступает на вход выбора кристалла CS микросхемы  $D_i$  и подключает ее к магистрали. В то же время сигнал  $SEL_i=0$  преобразуется в 4-разрядный код  $B4\ B3\ B2\ B1$ , который считывается процессором. В зависимости от числа диодов в блоке и варианта их подключения возможно формирование 16 ответных кодовых комбинаций.

Каждая такая комбинация соответствует одной из 16 градаций быстродействия блоков памяти. Процессор, получив код  $B4\ B3\ B2\ B1$ , формирует временную диаграмму магистральных сигналов с учетом быстродействия

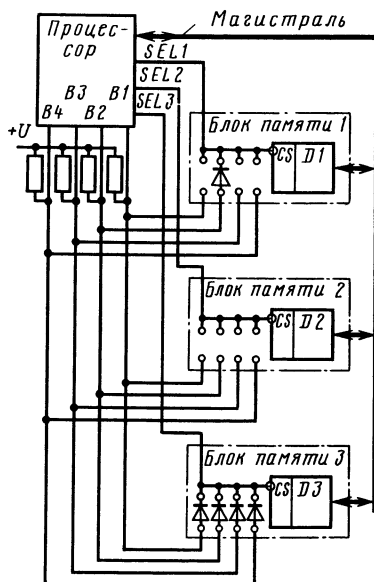


Рис. 3.9. МикроЭВМ с блоками памяти, имеющими разное быстродействие.

Сигнал выбора блока  $SEL_i=0$  возвращается в процессор в виде кода  $B4...B1$ , характеризующего быстродействие блока  $i$

адресуемого абонента, т. е. без лишних временных запасов, как это делалось бы при расчете на худший случай. Обмен информацией между процессором и абонентом осуществляется по магистрали в соответствии с некоторым интерфейсом, который не представляет для нас интереса.

Подобный способ опроса абонентов применим не только для выявления их быстройдействия. С помощью диодов можно, например, кодировать информацию о типе устройства (блок памяти, контроллер, адаптер магистралей и т. п.) и о его характеристиках. Например, для контроллера последовательного канала связи код В4 В3 В2 В1 может отображать скорость передачи информации по каналу и т. п.

### 3.4. Адресное взаимодействие устройств и передача запросов прерываний по параллельной информационной шине и двум управляющим линиям [46]

В схеме, показанной на рис. 3.10, активное устройство F способно инициировать адресный обмен информацией (в режиме чтения или записи) с пассивным устройством А, В или С и реагировать на сигналы запросов прерываний со стороны пассивных устройств. Взаимодействие между устройствами осуществляется по магистрали М, содержащей две управляющие линии C1, C2 и шину D данных (строго говоря, ее следовало бы назвать шиной адресов — команд — данных). Число пассивных устройств не ограничивается тремя; разрядность шины D может быть произвольной.

Все линии магистрали М двунаправленные. Линии C1 и C2 выполнены по схеме с открытым коллектором, нагрузочные резисторы (или резисторные делители напряжения) на рисунке не показаны. В пассивном состоянии за счет нагрузочных резисторов, соединенных с шиной питающего напряжения, в этих линиях поддерживаются напряжения высокого уровня (Н). При срабатывании хотя бы одного элемента — передатчика, подключенного к линии, в ней формируется напряжение низкого уровня (L). Таким образом, линии C1 и C2 реализуют функцию Монтажное ИЛИ по отношению к сигналам, представленным напряжениями низкого уровня. Линии шины D могут быть выполнены по схеме с открытым коллектором или с тремя состояниями: активный 0, активная 1, выключено. Рассмотрим сначала операцию чтения (рис. 3.11).

1. В момент  $t_0$  активное устройство F, предварительно убедившись в том, что в линии C1 присутствует Н-уровень, устанавливает в линиях шины

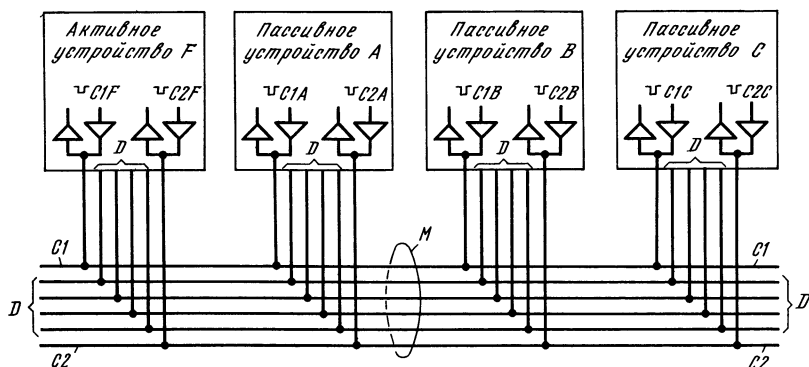


Рис. 3.10. Схема связи между активным и пассивными устройствами. Магистраль М позволяет активному устройству передавать и принимать данные от выбранного пассивного устройства, а также реагировать на запросы прерываний

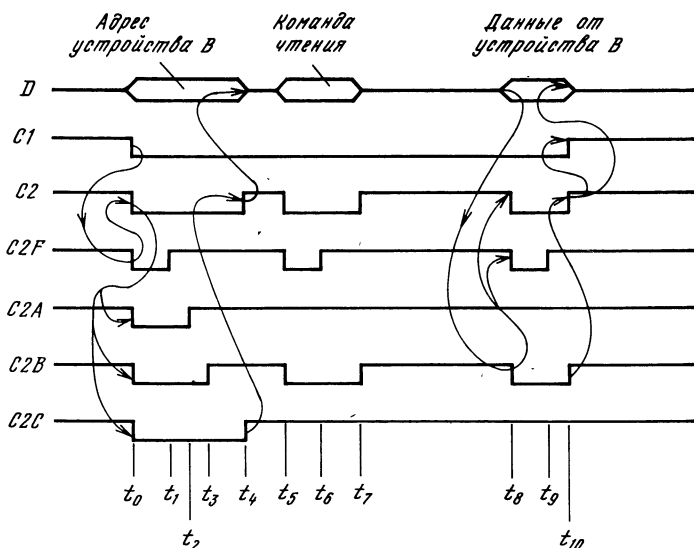


Рис. 3.11. Временные диаграммы обращения активного устройства F к пассивному устройству В в режиме чтения

D адрес (номер) пассивного устройства (в данном примере — устройства В), с которым предстоит провести сеанс связи. Если к моменту  $t_0$  в линии C1 присутствует L-уровень, то это означает, что одно или несколько пассивных устройств установили сигналы запросов прерывания программы, выполняемой активным устройством, или, иными словами, одно или несколько пассивных устройств требуют обслуживания. Эта ситуация будет рассмотрена позже.

Одновременно с выдачей адреса устройства В в шину D активное устройство F формирует сигнал C1F=L, который передается в общую управляющую линию C1 и оповещает всех абонентов о том, что магистраль M занята. Затем после незначительной задержки, достаточной для окончания переходных процессов в линиях шины D (эта задержка на рисунке не отражена), активное устройство F формирует сигнал C2F=L, который передается в общую управляющую линию C2.

Начиная с этого момента, пассивные устройства А, В и С одновременно вступают в сеанс связи с активным устройством F. В подтверждение этого они формируют сигналы C2A=C2B=C2C=L. Таким образом, в момент, близкий  $t_0$ , в шине D установлен адрес пассивного устройства В, в линии C1 — сигнал L-уровня, исходящий от активного устройства F (и, возможно, от пассивных устройств, которые успели установить в этой линии сигналы запросов прерываний, пока развивались описанные процессы, но об этом — позже), в линии C2 — сигнал L-уровня, порожденный одновременно всеми четырьмя устройствами. Пассивные устройства А, В и С одновременно приступают к дешифрации адреса, установленного в шине D.

2. В момент  $t_1$  активное устройство F снимает сигнал C2F, т. е. переводит его в состояние H. В данном примере пассивные устройства А, В и С к моменту  $t_1$  еще не закончили дешифрацию адреса и коллективно поддерживают на управляющей линии C2 магистраль сигнал L-уровня. Поэтому активное устройство, проверив состояние линии C2 (свой сигнал C2F уже снят), продолжает выдачу адреса пассивного устройства В в шину D и ожидает момента перехода этой линии в состояние H.

3. В момент  $t_2$  пассивное устройство А заканчивает дешифрацию адреса, установленного в шине D, и снимает сигнал C2A, т. е. переводит его в состояние Н. Это, однако, не приводит к изменению состояния линии C2, так как оставшиеся (более медленные) пассивные устройства В и С продолжают дешифрацию и поддерживают эту линию в состоянии L сигналами C2B=C2C=L. Пассивное устройство А в дальнейшем не участвует в данном сеансе связи с активным устройством F, так как его собственный адрес не совпал с адресом, установленным в шине D.

4. В момент  $t_3$  пассивное устройство В заканчивает дешифрацию адреса и обнаруживает, что ему предстоит участвовать в обмене информацией с активным устройством F в режиме чтения или записи. Снятие сигнала C2B не влияет на состояние линии C2, так как самое медленное устройство (С) еще не закончило дешифрацию адреса.

5. В момент  $t_4$  устройство С заканчивает дешифрацию адреса, снимает сигнал C2C и в дальнейшем не участвует в текущем сеансе связи (пока линия C1 находится в состоянии L). Сигнал в линии C2 переходит в состояние Н, что побуждает активное устройство F закончить цикл ожидания, освободить шину D и продолжить сеанс связи с выбранным абонентом — устройством В.

Адресный код абонента может передаваться по шине D в несколько этапов, если его разрядность превышает число линий в этой шине. При этом по мере передачи фрагментов адресного кода конкретизируется выбираемый абонент, а остальные выбывают из рассмотрения и ожидают начала следующего сеанса связи.

6. В момент  $t_5$  активное устройство F выдает в шину данных D команду — в данном примере — код операции чтения, а затем после окончания переходных процессов в этой шине формирует сигнал C2F=L, который проходит в линию C2. Пассивное устройство В, выбранное в качестве абонента, получив сигнал L-уровня из линии C2, формирует сигнал C2B=L, который также поступает в эту линию. Устройства А и С не участвуют в обмене информацией и находятся в пассивном состоянии.

7. В момент  $t_6$  активное устройство F снимает сигнал C2F и проверяет состояние линии C2. Так как устройство В еще не закончило прием команды, состояние линии C2 не меняется, активное устройство вынуждено продолжать выдачу команды.

8. В момент  $t_7$  пассивное устройство В снимает сигнал C2B, в линии C2 формируется Н-уровень. Команда воспринята, поэтому активное устройство F освобождает шину D и ожидает поступления данных от устройства В. Команда может передаваться в несколько этапов, если она «длинная».

9. В момент  $t_8$  данные, считанные из пассивного устройства В, выдаются в шину D. Их истинность с некоторой задержкой подтверждается сигналом C2B=L, который проходит в линию C2. Активное устройство F регистрирует переход линии C2 в состояние L, приступает к приему считанных данных и формирует сигнал C2F=L, дополнительно фиксирующий линию C2 в состоянии L.

10. В момент  $t_9$  активное устройство F заканчивает прием данных и снимает сигнал C2F.

11. В момент  $t_{10}$  пассивное устройство В по собственной инициативе снимает сигнал C2B и затем проверяет состояние линии C2. Если эта линия все еще находится в состоянии L, то это означает, что прием данных активным устройством еще не закончен, поэтому пассивное устройство В вынуждено ждать перехода линии C2 в состояние Н. В нашем примере, однако, данные уже приняты активным устройством F, поэтому снятие сигнала C2B пассивным устройством В влечет за собой формирование Н-уровня в линии C2.

Это событие позволяет пассивному устройству В завершить выдачу данных в шину D и перейти в исходное состояние, а активному устройству F — завершить выполнение всей операции (моменты  $t_0$ — $t_{10}$ ) и освободить магистраль М снятием сигнала C1F, что приводит к переходу линии C1

в состояние Н. Отметим, что сигналы запросов на прерывание, которые могли бы воспрепятствовать формированию Н-уровня в линии С1, к моменту  $t_{10}$  заведомо отсутствуют—они были погашены первым же отрицательным импульсом в линии С2.

Данные могут передаваться в несколько этапов, если разрядность шины D недостаточна для отображения полного слова. Пассивное устройство может содержать несколько элементов памяти, отличающихся адресами, например регистр данных и регистр состояния.

При выполнении операции записи события развиваются аналогично, но на конечном этапе обмена (моменты  $t_8$ — $t_{10}$ ) активное и пассивное устройства меняются ролями, так как данные передаются в противоположном направлении—от активного устройства к пассивному. Процедура записи завершается так же, как и процедура чтения,—активное устройство F переводит линию С1 в состояние Н. (Временные диаграммы записи здесь не рассматриваются.)

Из приведенного описания следует, что скорость обмена информацией между устройствами зависит от их быстродействия. На начальном этапе обмена, при передаче адреса, приходится ориентироваться на худший случай, чтобы самое медленное устройство (С) успело завершить дешифрацию адреса (период  $t_0$ — $t_4$ ). Однако в дальнейшем самое медленное устройство (С) выбывает, и скорость обмена информацией возрастает в той мере, насколько быстрым является адресуемое устройство (В).

В системе могут использоваться «широковещательные» команды, адресованные всем устройствам или группам устройств (например, Общий сброс, Запрет прерываний, Чтение флагов прерываний (через разные линии шины D) и т. п.). При выполнении таких команд длительность отрицательных импульсов в линии С2 определяется временными параметрами самого медленного устройства в группе—эта линия, как отмечалось, выполняет функцию Монтажное ИЛИ по отношению к сигналам L-уровней. Поэтому гонок не возникает. Рассмотрим теперь работу системы в режиме прерывания (рис. 3.12).

В момент  $t_0$  пассивное устройство А (источник прерывания), предварительно убедившись, что в линии С1 присутствует Н-уровень, формирует сигнал  $C1A=L$ , который передается в линию С1 (о конфликтах—позже). Активное устройство F по отрицательному фронту сигнала в линии С1 (если не оно само этот сигнал сформировало) временно прекращает выполнение текущей (фоновой) программы, запоминает параметры возврата к ней и переходит к прерывающей программе. Вместе с тем активное устройство F формирует сигнал  $C1F$ , который передается в линию С1. Начиная с этого момента, активное устройство перехватило инициативу, отобрав магистраль у пассивного устройства, и взяло на себя все тяготы управления магистралью.

Прерывающая программа предусматривает поиск источника прерывания путем последовательного опроса регистров состояния (или флаговых триггеров) пассивных устройств А, В, С. В данном примере предполагается, что последовательность опроса следующая—В, А, С (устройству В присвоен максимальный приоритет), поэтому первым опрашивается устройство В, точнее, его регистр состояния, по содержанию которого можно судить о том, имеется ли запрос на прерывание. Временные диаграммы, сопровождающие считывание информации из устройства В, уже рассматривались, поэтому они «скрыты» на рисунке.

Важно отметить, что по первому отрицательному фронту сигнала в линии С2 гасятся все имеющиеся запросы на прерывание, в данном примере снимается запросный сигнал  $C1A$ . Точнее говоря, флаговые разряды регистров состояний пассивных устройств не гасятся, а всего лишь запрещается выдача запросных сигналов в линию С1 на протяжении времени проведения текущего сеанса связи, т. е. до тех пор, пока в этой линии присутствует сигнал L-уровня. Установка в 0 флагового разряда регистра состояния пассивного устройства, требующего обслуживания, осуществляется программно, в процессе взаимодействия с активным устройством.

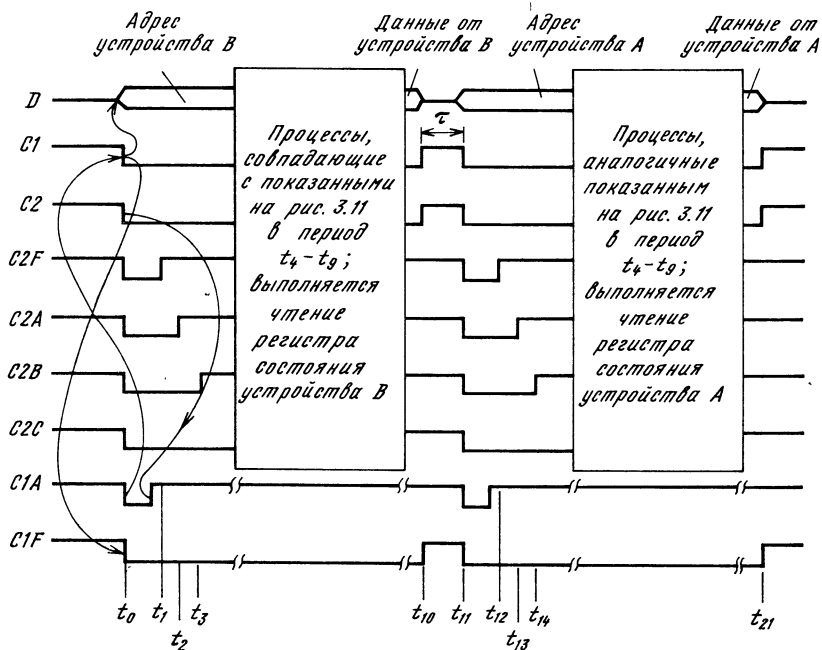


Рис. 3.12. Временные диаграммы работы системы (рис. 3.10) в режиме прерывания. В каждом сеансе связи после формирования отрицательного фронта сигнала C2 и до освобождения магистрали пассивным устройствам запрещено пользоваться линией C1 для выдачи в нее запросных сигналов

В момент  $t_{10}$  активное устройство F завершает чтение регистра состояния устройства B, затем программно анализирует полученный код и принимает к сведению, что устройство B не требовало обслуживания.

Через некоторый достаточно малый фиксированный интервал времени  $\tau$  после освобождения магистрали устройство A вновь формирует сигнал  $C1A=L$ —запрос на прерывание. (Это устройство еще не знает, что предыдущий его запрос уже воспринят и начал обрабатываться.) Процессы повторяются—инициатива вновь переходит к активному устройству F. Оно продолжает опрос (игнорируя запросы прерываний, чтобы не «зациклиться») и на этот раз считывает содержимое регистра состояния устройства A, в котором флаговый разряд установлен в 1. Источник прерывания найден, поэтому активное устройство F обслуживает устройство A, в частности устанавливает в 0 флаговый разряд его регистра состояния, чтобы в дальнейшем повторно не отреагировать на уже удовлетворенный запрос.

По окончании обслуживания одного или всех обнаруженных источников прерываний активное устройство F возвращается к выполнению прерванной (фоновой) программы.

Как упоминалось, опрос флагов можно вести групповым способом в один или несколько этапов, в зависимости от числа устройств и разрядности шины данных. При таком опросе флаговые триггеры, размещенные в разных устройствах, имеют одинаковые адреса; эти триггеры опрашиваются одновременно и выдают информацию в разные разряды шины данных, так что одно считывание позволяет узнать о состоянии сразу нескольких пассивных устройств.

В системе возможны ситуации, связанные с одновременным (или почти одновременным) формированием L-уровня в линии С1 по инициативе как активного, так и пассивных устройств. При этом, однако, неразрешимых конфликтов не возникает. Действительно, возможен один из двух исходов.

1. Активное устройство «не заметило», что отрицательный фронт сигнала С1 сформирован по инициативе устройства—источника прерывания, так как одновременно или почти одновременно выдан свой сигнал  $C1F=L$ . Иными словами, информация о запросе прерывания вроде бы потеряна (по сигналу  $C2F=L$  запрос снимается, как было показано ранее). Выручает то, что запрос вновь формируется после окончания текущего сеанса связи (см. рис. 3.12, момент  $t_{11}$ ) и воспринимается в ближайшем или последующих циклах работы магистрали (бесконечные гонки возможны только теоретически, да и то при неправильно выбранном значении  $\tau$ ).

2. Активное устройство все же успело заметить, что отрицательный фронт сигнала в линии С1 сформирован кем-то из соседей, прежде чем им самим. В этом случае выполняется описанная ранее процедура прерывания, поиска и обслуживания источника с последующим возвратом к фоновой программе.

### **3.5. МикроЭВМ с расширенными возможностями адресации [47, 48]**

Центральный 8-разрядный процессор ЦП микроЭВМ [47] (рис. 3.13) имеет 16-разрядную шину адреса и, следовательно, способен адресовать  $2^{16}=64K$  байт. Однако, как видно из схемы, процессор сопряжен с ПЗУ ( $64K \times 8$  бит), ОЗУ ( $8K \times 8$  бит) и двумя контроллерами, так что общее число адресуемых элементов превышает 64K. Как же они «умещаются» в адресном пространстве микроЭВМ?

Рассмотрим сначала содержимое ПЗУ (рис. 3.14). Массив ячеек ПЗУ условно разделен на три области. В первой и второй размещена основная (фоновая) программа, в третьей—прерывающая. Последняя периодически вызывается процессором при поступлении на его вход сигналов прерывания INTR от одного из контроллеров. Точнее, при получении сигнала INTR процессор временно откладывает выполнение основной программы, запоминает в ОЗУ параметры возврата к ней, переходит по адресу 0038, исполняет несколько команд из близлежащих ячеек и затем приступает к работе по программе из третьей области. По окончании работы по этой программе управление вновь передается в первую область ПЗУ, выполняются несколько команд из этой области и, наконец, процессор возвращается к прерванному (фоновому) заданию.

Таким образом, начало и конец прерывающей программы размещены в первой области ПЗУ, а собственно прерывающая программа—в третьей области. Если процессор работает в третьей области, то он не пользуется информацией из второй—так построена прерывающая программа. Поэтому «в зоне постоянного внимания» процессора одновременно могут находиться либо первая и вторая области (когда нет сигналов прерываний и выполняется основная программа), либо первая и третья (когда обрабатывается запрос прерывания от контроллера).

Иными словами, не обязательно помещать в адресное пространство микроЭВМ одновременно всю информацию, содержащуюся в ПЗУ,—тогда все это пространство оказалось бы занятым и нигде было бы разместить другие элементы. Предлагается вставлять в адресное пространство микроЭВМ либо первую и вторую (рис. 3.15, а), либо первую и третью (рис. 3.15, б) области ПЗУ, а оставшуюся часть (8K байт) распределить между ОЗУ (ему выделяется только 6K байт) и регистрами контроллеров (2K байт).

То или иное распределение адресного пространства выбирается процессором в зависимости от текущей ситуации установкой программно-доступного

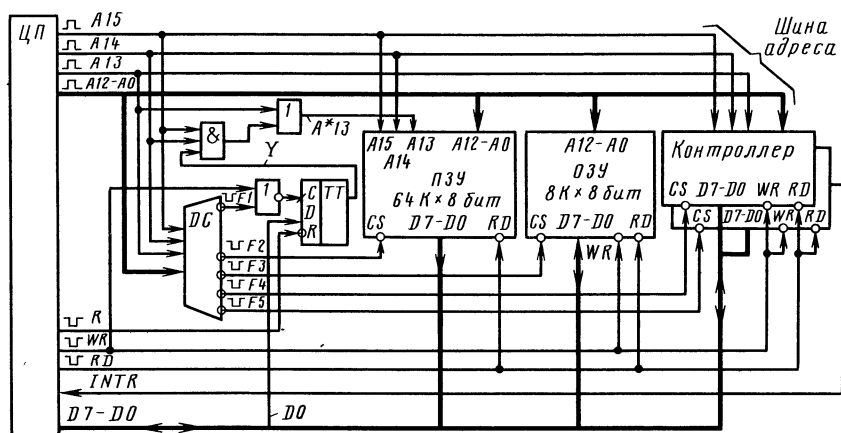


Рис. 3.13. МикроЭВМ с расширенными возможностями адресации

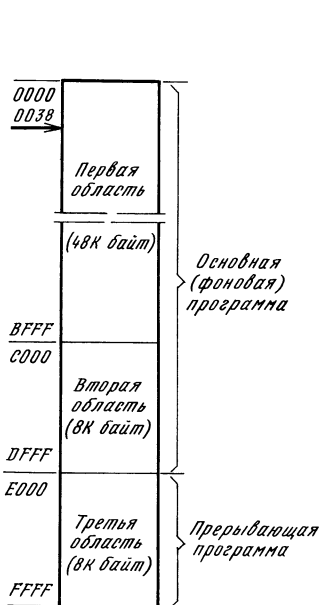


Рис. 3.14. Содержимое ПЗУ

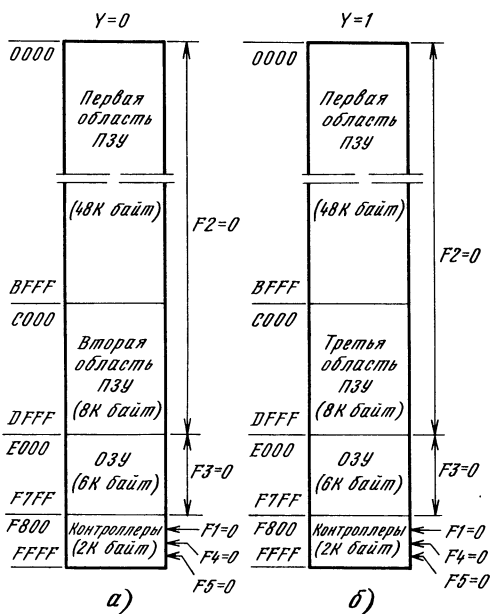


Рис. 3.15. Распределение адресного пространства микроЭВМ:

а — при выполнении основной (фоновой) программы; б — при обработке запроса прерывания

триггера ТТ (см. рис. 3.13) в нужное состояние. При установке этого триггера в 0 ( $Y=0$ ) в области C000—DFFF адресного пространства микроЭВМ присутствует вторая область ПЗУ с одноименными адресами; при  $Y=1$ —третья область с адресами ПЗУ E000—FFFF. В отсутствие прерываний используется распределение адресного пространства, соответствующее условию  $Y=0$ .

После получения сигнала прерывания, на начальном этапе выполнения прерывающей программы, когда процессор все еще работает в первой области ПЗУ, триггер ТТ программно устанавливается в 1, что приводит к перераспределению адресного пространства, т. е. включению в него третьей области ПЗУ. По завершении работы в третьей области процессор возвращается в первую и программно устанавливает триггер ТТ в исходное состояние ( $Y=0$ ), что приводит к восстановлению первоначального распределения адресного пространства микроЭВМ.

Адрес программно-доступного триггера ТТ размещен в области F800—FFFF адресного пространства микроЭВМ; в этой же области выделены адреса для обращения к контроллерам. Чтобы установить триггер ТТ в нужное состояние, процессор выполняет команду пересылки содержимого одного из своих внутренних регистров общего назначения в ячейку памяти с адресом, в ответ на который дешифратор DC формирует сигнал  $F1=0$ .

Байт, выданный процессором в шину данных, содержит в разряде D0 нужный бит (0 или 1), который поступает на информационный вход триггера ТТ (сигнал начальной установки на входе R этого триггера отсутствует). Выполнение команды записи сопровождается сигналом  $WR=0$ , поэтому при обращении к триггеру ТТ срабатывает элемент ИЛИ—НЕ, на входе С синхронизации формируется положительный фронт сигнала, по которому бит данных с D-входа запоминается в триггере.

Сигнал  $F2=0$  формируется при попадании адреса, сформированного процессором, в диапазон 0000—DFFF и поступает на вход CS выбора кристалла ПЗУ, разрешая его работу. Сигнал  $F3=0$  соответствует попаданию адреса в диапазон E000—F7FF. Сигналы  $F4=0$  и  $F5=0$  формируются при обращении процессора к контроллерам.

Перераспределение адресного пространства в зависимости от состояния триггера ТТ происходит так. При  $Y=0$  элемент И закрыт по нижнему входу, на его выходе постоянно присутствует сигнал лог. 0. Поэтому элемент ИЛИ транслирует сигнал A13 с адресной шины процессора на соответствующий вход ПЗУ,  $A*13=A13$ . Остальные сигналы ( $A0—A12$ ,  $A14$ ,  $A15$ ) поступают на входы ПЗУ непосредственно с адресной шины процессора, поэтому адресное пространство микроЭВМ совпадает с адресным пространством ПЗУ во всем выделенном ему диапазоне 0000—DFFF (рис. 3.15, а).

При  $Y=1$  и попадании адреса в диапазон 0000—BFFF по крайней мере один из сигналов  $A15$  или  $A14$  равен 0, поэтому элемент И закрыт, сигнал A13 передается через элемент ИЛИ на соответствующий вход ПЗУ ( $A*13=A13$ ), следовательно, каких-либо отклонений от «естественного» размещения ПЗУ в адресном пространстве микроЭВМ не наблюдается. Однако при  $Y=1$  и попадании адреса в диапазон C000—FFFF, когда  $A15=A14=1$ , на выходе элемента И формируется сигнал лог. 1, который безусловно передается через элемент ИЛИ на вход A13 ПЗУ. Это приводит к тому, что обращения процессора по адресам C000—DFFF (в которых  $A13=0$ ) с «точки зрения» ПЗУ подменяются обращениями по адресам E000—FFFF (тем же, но с установленным в единицу разрядом A13). Иными словами, вместо второй области ПЗУ активизируется третья, что и требовалось.

Так как разряд  $A*13$  поступает только в ПЗУ, другие устройства (ОЗУ, контроллеры) не реагируют на сигнал  $Y$  и не меняют положения в адресном пространстве при его преобразовании.

После включения напряжения питания процессор формирует сигнал  $R=0$ , по которому триггер ТТ устанавливается в 0, и приступает к работе по



фоновой программе. Распределение адресного пространства соответствует приведенному на рис. 3.15, а. Выполнение фоновой программы приостанавливается с получением сигнала прерывания. Структура прерывающей программы приведена на рис. 3.16. Команды установки  $Y=1$  и  $Y=0$  размещены в первой области ПЗУ.

Рассмотрим иное решение [48] (рис. 3.17). МикроЭВМ содержит ОЗУ объемом 16К байт и три ПЗУ, каждое объемом 64К байт, поэтому всего имеется 208К адресуемых ячеек. Адресное пространство, как и в предыдущем примере, составляет 64К ячеек, поэтому предлагаются шесть программно выбираемых способов его заполнения в зависимости от кода  $P2\ P1\ P0$  (рис. 3.18).

Блок А объемом 16К байт первого ПЗУ содержит часто используемые подпрограммы, к которым процессор обращается в любых ситуациях независимо от выбранного варианта распределения адресного пространства. Поэтому система адресации построена так, что этот блок выбирается во всех вариантах при условии равенства нулю двух старших разрядов адреса:  $A15=A14=0$ . При  $A15=A14=1$  во всех ситуациях выбирается ОЗУ, так как в нем хранится актуальная для всех вариантов информация, к которой необходим быстрый доступ.

В оставшуюся область адресного пространства объемом 32К байт (4000—BFFF) в зависимости от «намерений» процессора (программиста) встраивается одна из страниц ПЗУ: BC, DE, FG, HI, JK, L (последняя страница содержит только 16К байт, так что в адресном пространстве появляется незанятая область 8000—BFFF).

Перераспределение адресного пространства осуществляется коррекцией старшего разряда адреса, поступающего на ПЗУ (табл. 3.1), и формированием сигналов выбора ПЗУ CS1, CS2 и CS3 с помощью дешифратора DC, функции которого описаны в табл. 3.2. Сигнал  $VMA=1$  подтверждает истинность адреса, установленного процессором, сигнал  $WR/RD$  задает режим работы ОЗУ—запись или чтение, а сигнал  $CS4=1$  сопровождает выбор ОЗУ.

Таблица 3.2

P2	P1	P0	A15	A14	CS1	CS2	CS3
×	×	×	0	0	1	0	0
×	×	×	1	1	0	0	0
0	0	0	0	1	1	0	0
0	0	0	1	0	1	0	0
0	0	1	0	1	1	0	0
0	0	1	1	0	0	1	0
0	1	0	0	1	0	1	0
0	1	0	1	0	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	1
1	0	1	0	1	0	0	1
1	0	1	1	0	0	0	0

Примечание. ×—произвольное значение разряда (0 или 1).

После включения напряжения питания процессор стартует в одну из ячеек блока А—его положение в адресном пространстве не зависит от кода  $P2\ P1\ P0$ , который первоначально не определен. Затем, следуя программе начального пуска, хранимой в блоке А, процессор загружает в выходной порт нужный код и переходит к выполнению рабочего задания.

### 3.6. Подключение к микроЭВМ плат памяти одинакового или разного объема с автоматическим перераспределением адресного пространства [49]

Платы 1—4 (рис. 3.19) памяти микроЭВМ устанавливаются в соединители, смонтированные на основной плате. В минимальной конфигурации микроЭВМ должна присутствовать по крайней мере одна плата памяти. Порядок размещения плат памяти в соединителях произволен. Некоторые соединители могут оставаться незанятыми, их положение также произвольно.

Объем памяти на плате 1 (2—4) в данном примере может быть равным 128К, 256К или 512К слов. В зависимости от объема памяти с помощью перемычек S1 и S2 задается соответствующий код W1W2 (W3W4, W5W6, W7W8), который считывается центральным процессором микроЭВМ на этапе ее инициализации при включении питающего напряжения. Например, наличие только одной перемычки S1 в плате 1 соответствует объему памяти в этой плате, равному 128К слов (W1=0, W2=1); наличие единственной перемычки S2 отображает объем памяти, равный 256К слов (W1=1, W2=0); если установлены обе перемычки, то объем памяти в плате 1 составляет 512К слов (W1=W2=0). Такая (или подобная) кодировка выбрана в связи с тем, что в каждой плате памяти должна быть установлена хотя бы одна перемычка (S1, S2), чтобы ЦП микроЭВМ при опросе сигналов W1—W8 мог отличить занятые разъемы от незанятых, с которых считываются пары единичных сигналов.

После опроса сигналов W1—W8 ЦП микроЭВМ на основании полученных сведений о составе плат памяти формирует четыре 4-разрядные двоичные константы G1—G4 и пересылает их в программно-доступный регистр RG по шине данных D. Запись группы констант осуществляется по сигналу LD опознания обращения к регистру RG со стороны ЦП микроЭВМ.

После загрузки регистра RG с помощью компараторов D1—D4 и дешифратора DC платам памяти выделяются необходимые области адресного

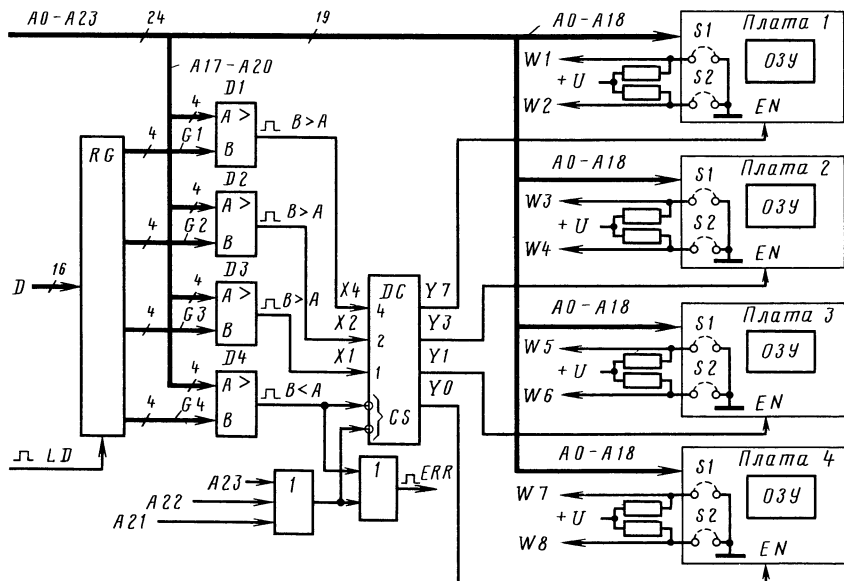


Рис. 3.19. Фрагмент схемы микроЭВМ.

Платы памяти 1—4 устанавливаются в разъемы, размещенные на основной плате

пространства, которые образуют единый непрерывный массив адресов, несмотря на то, что между занятыми соединителями могут оставаться незанятые. Это достигается благодаря описанному далее выбору констант  $G1—G4$ .

Если плата 1 не установлена в соединитель ( $W1=W2=1$ ), то  $G1=0000$ . Если плата 1 установлена в соединитель и объем памяти, размещенной на этой плате, составляет 128К, 256К или 512К слов, то константа  $G1$  выбирается соответственно равной 0001, 0010 или 0100.

Если плата 2 не установлена в соединитель, то константа  $G2$  принимается равной  $G1$ . Если плата 2 установлена в соединитель и объем памяти, размещенной на этой плате, составляет 128К, 256К или 512К слов, то константа  $G2$  выбирается соответственно равной  $G1+0001$ ,  $G1+0010$  или  $G1+0100$  (сложение арифметическое). Таким образом, значение  $G2$  может меняться от 0000 (когда платы 1 и 2 не установлены) до 1000 (когда платы 1 и 2 установлены и каждая из них имеет объем памяти, равный 512К слов).

Константа  $G3$  вычисляется аналогично. Так, в зависимости от объема памяти на плате 3  $G3=G2$  (плата 3 не установлена в разъем),  $G3=G2+0001$  (на плате 3 размещены 128К слов),  $G3=G2+0010$  (на плате 3 размещены 256К слов) либо  $G3=G2+0100$  (на плате 3 размещены 512К слов). В результате значение  $G3$  может меняться от 0000 (платы 1—3 не установлены в соединители) до 1100 (платы 1—3 установлены в соединители и каждая из них имеет объем памяти, равный 512К слов).

Константа  $G4$  вычисляется несколько по-другому. Если плата 4 не установлена в соединитель, то  $G4=G3-1$ . В этой ситуации обязательно установлена одна из плат 1—3 (как отмечалось, в микроЭВМ должна присутствовать хотя бы одна плата памяти), так что  $G4 \geq 0$ . Если плата 4 установлена в соединитель и объем памяти, размещенной на этой плате, составляет 128К, 256К или 512К слов, то константа  $G4$  выбирается соответственно равной  $G3$ ,  $G3+0001$  или  $G3+0011$  (в отличие от формул для получения констант  $G2$  и  $G3$ , здесь из результата сложения дополнительно вычитается единица). Таким образом, константа  $G4$  может меняться от 0000 (когда установлена только одна из плат 1—4 и объем имеющейся памяти составляет только 128К слов) до 1111 (когда присутствуют все платы и объем памяти в каждой из них максимален, т. е. составляет 512К слов).

Так как адресная шина микроЭВМ содержит 24 линии, объем адресного пространства составляет  $2^{24}=16\text{М}$  ячеек. В этом пространстве, начиная с нулевого адреса, размещена область памяти микроЭВМ. Минимальный ее объем составляет 128К ячеек, максимальный—2М ячеек. Поскольку максимальный объем памяти, содержащейся на плате, равен  $2^{19}=512\text{К}$  ячеек, к каждому разьему для установки платы подключены 19 младших разрядных линий ( $A0—A18$ ) адресной шины.

При обращении к платам 1—4 памяти в трех старших разрядах  $A23$ ,  $A22$  и  $A21$  адреса должны присутствовать лог. 0. Это условие соответствует попаданию текущего адреса в диапазон 0— $(2\text{М}-1)$  и формированию на выходе трехходового элемента ИЛИ сигнала лог. 0, который поступает на первый инверсный вход CS выбора кристалла дешифратора DC. Второй инверсный вход выбора кристалла дешифратора соединен с выходом компаратора D4.

Благодаря этому дешифратор DC открывается одновременно по обоим входам выбора кристалла только при попадании текущего адреса в диапазон адресов, занимаемых платами 1—4. При выходе адреса за этот диапазон на одном или двух инверсных входах CS выбора кристалла дешифратора формируются сигналы лог. 1, в результате дешифратор закрывается, на его выходах  $Y7$ ,  $Y3$ ,  $Y1$  и  $Y0$  поддерживаются сигналы лог. 0. Одновременно с этим на выходе двухходового элемента ИЛИ формируется единичный сигнал ERR (Ошибка), который свидетельствует о том, что адресный код, установленный процессором или иным устройством в адресной шине, превышает границу области адресов, занятых платами памяти.

При правильном обращении, когда текущий адрес попадает в диапазон адресов, занимаемых платами 1—4, сигнал ошибки не вырабатывается ( $ERR=0$ ), дешифратор DC анализирует трехразрядный код  $X_4 X_2 X_1$  ( $X_4$ —старший разряд) и формирует на одном из выходов  $Y_7, Y_3, Y_1$  или  $Y_0$  сигнал лог. 1, который поступает на вход EN разрешения работы соответствующей платы памяти. В соответствии с кодом в младших разрядах адреса, установленного в адресной шине, в выбранной дешифратором плате активизируется некоторая ячейка памяти, из которой считывается или в которую записывается слово данных (управляющие линии Чтение памяти, Запись памяти и шина данных ОЗУ на рисунке не показаны для его упрощения).

Плата 1 выбирается сигналом  $Y_7=1$ , который формируется при  $X_4=X_2=X_1=1$  (здесь и далее предполагается, что обращение правильное,  $ERR=0$ ). Условие  $X_4=1$  означает, что  $G_1>A$ , где  $A$ —4-разрядный код  $A_{20} A_{19} A_{18} A_{17}$ , поступивший из адресной шины. Аналогично условия  $X_2=1$  и  $X_1=1$  означают, что  $G_2>A$  и  $G_3>A$ . Действительно, в данной ситуации адрес обращения к ячейке памяти принадлежит области адресов, занимаемых платой 1, и, подавно, областям адресов, занимаемых платами 1—2, 1—2—3 и 1—2—3—4 (если платы 2, 3 и 4 установлены в соединители).

Плата 2 выбирается сигналом  $Y_3=1$ , когда  $X_4=0, X_2=X_1=1$ . Это, в свою очередь, означает, что  $G_1\leq A$  (плата 1 отсутствует или адрес вышел за границу области адресов, покрываемых этой платой),  $G_2>A$  и  $G_3>A$ . В данном случае адрес попадает в область адресов, отведенных плате 2, и тем более—в области адресов, отведенных платам 2—3 и 2—3—4, если платы 3 и 4 установлены в соединители.

Плата 3 выбирается сигналом  $Y_1=1$ , когда  $X_4=X_2=0, X_1=1$ . Это соответствует либо отсутствию обеих плат 1 и 2, либо (при наличии одной или двух плат 1 и 2) превышению адресом  $A$  границы области адресов, покрываемых этими платами, а точнее, попаданию адреса в область, покрываемую платой 3.

Плата 4 выбирается сигналом  $Y_0=1$ , когда  $X_4=X_2=X_1=0$ , т.е. при выдаче правильного адреса, составляющая  $A$  которого больше или равна  $G_1, G_2$  и  $G_3$ . Здесь также возможны разные ситуации: когда платы 1, 2 и 3 отсутствуют, либо какие-то из них установлены, либо установлены все платы, но адрес обращения достаточно большой. Отметим, что если соединитель не занят, то соответствующий сигнал выбора этой платы не формируется ни при каких обстоятельствах.

Предлагаемое решение позволяет оператору «не задумываться» при удалении платы памяти в связи с ее ремонтом, при изменении объема памяти микроЭВМ в сторону увеличения или уменьшения (когда «лишняя» память просто не нужна). В момент включения напряжения питания микроЭВМ новая конфигурация плат памяти будет принята к сведению операционной системы.

Контакты соединителей для передачи сигналов  $W_i$  можно освободить и передавать эти сигналы в некоторый регистр на начальном этапе работы микроЭВМ через те же контакты, по которым в нормальном режиме передаются адресные или управляющие сигналы. Однако это уже иное, «самостоятельное» решение (см. § 3.9).

### **3.7. Адресный дешифратор, обслуживающий блоки памяти разного объема [50]**

При использовании микросхем ОЗУ, совместимых по контактам, таких как K565PV5 и K565PV7 (со структурой  $64K \times 1$  бит и  $256K \times 1$  бит), можно, устанавливая на печатной плате те или (и) другие, получать память нужного объема. В схеме, приведенной на рис. 3.20, каждый из трех блоков памяти содержит  $n$  микросхем K565PV5 или K565PV7 ( $n$ —разрядность шины данных).

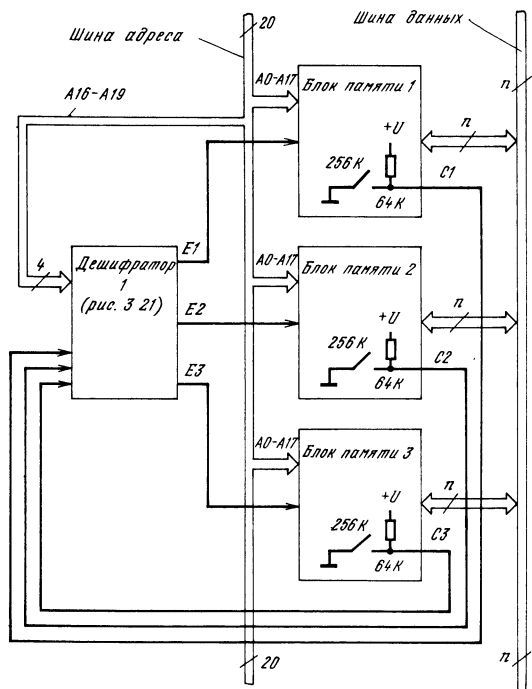


Рис. 3.20. Схема запоминающего устройства с тремя блоками памяти трех типов (в каждом блоке 256К или 64К n-разрядных ячеек). Дешифратор 1 выделяет блокам памяти области адресного пространства в соответствии с информацией об их типе. Шина управления магистрали не показана

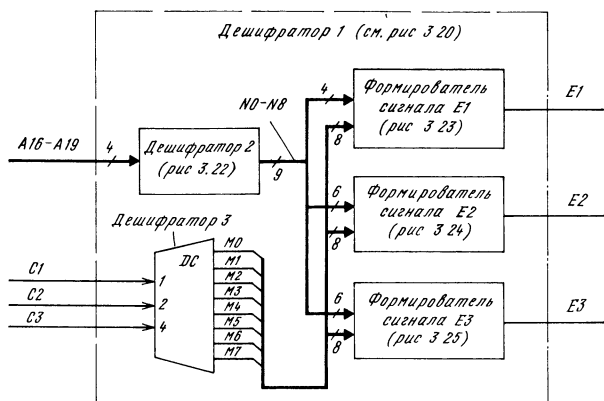


Рис. 3.21. Структурная схема дешифратора 1 (см. рис. 3.20)

Рис. 3.22. Функциональная схема дешифратора 2 (см. рис. 3.21)

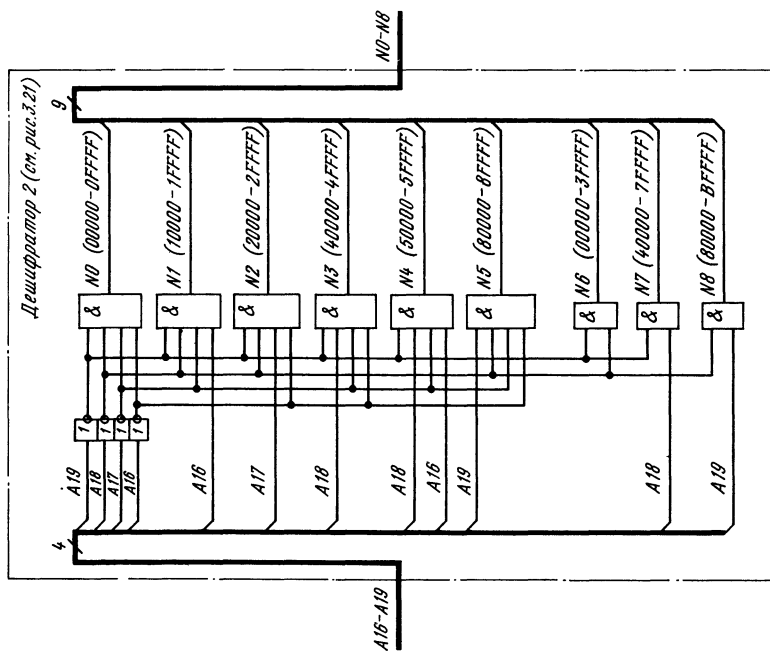
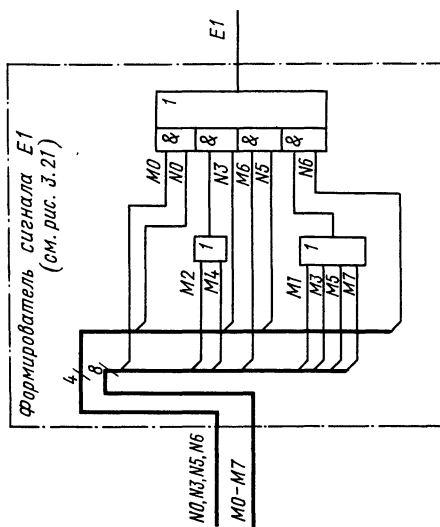


Рис. 3.23. Схема формирователя сигнала E1 (см. рис. 3.21)



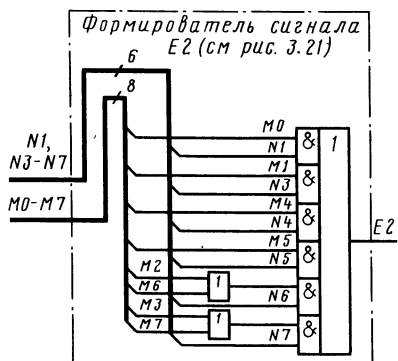


Рис. 3.24. Схема формирователя сигнала E2 (см. рис. 3.21)

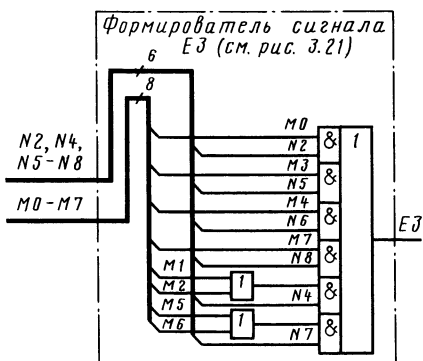


Рис. 3.25. Схема формирователя сигнала E3 (см. рис. 3.21)

В зависимости от типа примененных в блоке микросхем соответствующий ключ устанавливается в замкнутое или разомкнутое положение, как показано на рисунке. Далее предполагается, что задействованы все три блока памяти. Тогда возможны восемь ситуаций, отраженных в табл. 3.3. Дешифратор 1 выделяет блокам памяти области адресного пространства в соответствии с их объемом, что отражено в правой части таблицы. Структурная схема дешифратора 1 (рис. 3.20) приведена на рис. 3.21, ее составные части представлены на рис. 3.22—3.25.

Совместимость указанных микросхем означает, что на одно посадочное место на печатной плате может устанавливаться или та, или другая микросхема (рис. 3.26). Контакт 1 в микросхеме K565PY5 не задействован, так как она имеет 8 (а не 9, как в микросхеме K565PY7) адресных входов. Полный

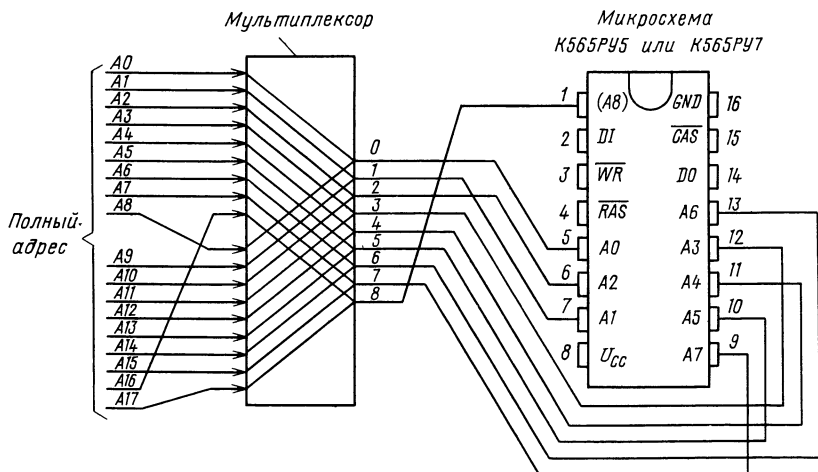


Рис. 3.26. Схема мультиплексирования адреса, обеспечивающая установку на печатной плате микросхем K565PY5 (со структурой 64K×1 бит) или K565PY7 (со структурой 256K×1 бит) без изменения топологии связей на печатной плате.

В микросхеме K565PY5 контакт 1 не задействован

адрес передается на адресные входы микросхемы памяти через мультиплексор в два этапа по отрицательным фронтам управляющих сигналов RAS и CAS.

Таблица 3.3

Число n-разрядных ячеек в блоках памяти			Области адресного пространства, выделяемые дешифратором 1 блокам памяти		
Блок 1	Блок 2	Блок 3	Блок 1	Блок 2	Блок 3
64К	64К	64К	00000—0FFFF	10000—1FFFF	20000—2FFFF
256К	64К	64К	00000—3FFFF	40000—4FFFF	50000—5FFFF
64К	256К	64К	40000—4FFFF	00000—3FFFF	50000—5FFFF
256К	256К	64К	00000—3FFFF	40000—7FFFF	80000—8FFFF
64К	64К	256К	40000—4FFFF	50000—5FFFF	00000—3FFFF
256К	64К	256К	00000—3FFFF	80000—8FFFF	40000—7FFFF
64К	256К	256К	80000—8FFFF	00000—3FFFF	40000—7FFFF
256К	256К	256К	00000—3FFFF	40000—7FFFF	80000—BFFFF

В зависимости от типа установленной на плату микросхемы разрядность полного адреса, используемого для выбора ячеек, равна 16 или 18. После двукратного мультиплексирования полного адреса образуются два 9-разрядных кода, при этом старшие разряды этих кодов, как следует из рис. 3.2б, отображают сигналы в линиях A16 и A17 полного адреса. Таким образом, информация на контакте 1 имеет смысл только при использовании микросхемы K565PY7, а при применении микросхемы K565PY5 эта информация просто теряется.

Регенерация хранимой информации осуществляется периодическим перебором 9-разрядных адресных кодов в режиме счета с периодом, примерно равным 15 мкс (0—младший, 8—старший разряды). При этом полный цикл перебора кодов в разрядах 0—6 не превышает 2 мс, в разрядах 0—8 не превышает 8 мс, что соответствует требованиям к циклу регенерации для микросхем K565PY5 и K565PY7. На время регенерации выходы счетчика адресных кодов подключаются к разрядам 0—8 (соответствующая аппаратура на рисунке не показана), регенерация сопровождается сигналом RAS, сигнал CAS не формируется.

### 3.8. Повышение быстродействия динамического ОЗУ путем «предсказания» номера блока памяти, к которому ожидается обращение [51]

В схеме (рис. 3.27) ЦП взаимодействует с динамическим ОЗУ (выделено штриховыми линиями) с использованием магистрали М и блока F преобразования адресов.

Шина данных магистрали содержит 32 линии, по которым из ОЗУ считывается двойное слово (четыре байта). В 32-разрядную ячейку ОЗУ могут записываться от одного до четырех байтов в зависимости от комбинации сигналов BE0—BE3 в линиях адресной шины. Из 16 возможных комбинаций этих сигналов используются только 10. В частности, при BE0=0, BE1=BE2=BE3=1 в 32-разрядную ячейку ОЗУ записывается только младший байт, остальные не изменяются; при BE0=BE1=BE2=BE3=0 в ячейку записываются четыре байта. Схема выбора байтов сигналами BE0—BE3 при записи данных в ячейку ОЗУ на рисунке не показана.

Логический адрес LA2—LA31, формируемый процессором, преобразуется блоком F в физический адрес A2—A25 одной из  $2^{24}=16\text{M}$  32-разрядных



После формирования отрицательного фронта сигнала CAS управляющие сигналы принимают следующие значения:  $C1=C2=1$ ,  $C3=0$ , что соответствует рассмотренному ранее исходному состоянию регистров RG1 и RG2.

Старшие разряды A22—A25 физического адреса поступают на информационные входы регистра RG3 и на входы компаратора, который сравнивает их со «старшими» разрядами, хранимыми в этом регистре. При совпадении входных 4-разрядных кодов компаратор формирует сигнал лог. 0, который разрешает прохождение сигнала записи  $WR=0$  из магистрали M через элемент ИЛИ на входы записи БИС ОЗУ.

Если на выходе компаратора присутствует сигнал лог. 1 (входные коды не совпали), то  $WR'=1$ . При этом микросхемы памяти, выбранные дешифратором DC (на их входы выбора кристалла подан сигнал разрешения работы с соответствующего выхода дешифратора), переводятся в режим чтения независимо от состояния сигнала WR в магистрали M. Регистр RG3 принимает входной код A22—A25 по положительному фронту сигнала C4, который формируется в конце каждого цикла обращения к памяти, а также, в некоторых случаях, в его начале.

Рассмотрим сначала работу ОЗУ в благоприятной ситуации, когда процессор выполняет ряд обращений к ячейкам памяти, размещенным в одном и том же блоке  $B_i$ . В регистре RG3 хранится и подтверждается в конце каждого обращения номер этого блока. Для доступа к ячейке памяти блока  $B_i$  процессор устанавливает в магистрали M логический адрес LA2—LA31, данные и сигналы BE0—BE3 (при записи), а также соответствующие сигналы управления. Блок F преобразования адресов вычисляет физический адрес ячейки памяти, разряды A2—A11 которого проходят сквозь регистр-защелку RG1 и поступают на входы всех БИС ОЗУ (усилители адресных и иных сигналов на рисунке не показаны).

В данном примере на выходе компаратора сформирован сигнал лог. 0, так как процессор обращается к тому же блоку  $B_i$ , что и в предыдущем цикле. Дешифратор DC заранее выбрал нужный блок памяти, так что операция чтения или записи начинается незамедлительно. Иными словами, из цикла обращения к ОЗУ исключаются задержки, обусловленные временем срабатывания дешифратора DC и соответствующих усилителей (не показаны).

В отсутствие регистра RG3, компаратора и элемента ИЛИ для ускорения работы ОЗУ можно было бы на начальном этапе обращения сигналами CS и RAS активизировать все его блоки, а сигнал CAS подавать только на выбранный блок. Однако такое решение привело бы к большим энергозатратам, так как ток, потребляемый выбранной (сигналами CS и RAS) микросхемой ОЗУ, примерно в 10 раз превышает ток в режиме пассивного хранения данных. В предлагаемом решении активизируется только один блок, а именно тот, на который одновременно поступают сигналы CS и RAS.

При «неудачном» обращении на выходе компаратора формируется сигнал лог. 1, который препятствует выполнению операции записи (сигнал  $WR'=1$  соответствует чтению) и воздействует на блок управления. В результате начатый цикл принудительно (досрочно) завершается, в регистр RG3 по сигналу C4 принимается правильный номер блока и далее начинается новое обращение по откорректированному адресу. Потери времени, связанные с неудачной адресацией, можно уменьшить при рациональном распределении физического адресного пространства, когда процессор преимущественно работает «внутри» блоков  $B_i$  (а не «перескакивает» из блока в блок).

Однако можно обойтись и без такого распределения, допуская возможность попеременных обращений процессора к разным блокам, если применить усовершенствования, показанные на рис. 3.28.

Для хранения номера блока вместо одного регистра RG3 использованы четыре «альтернативных»: RG3.1—RG3.4. В этих регистрах без какой-либо программной «поддержки» фиксируются номера блоков, к которым в некоторый период времени процессор проявляет «повышенный интерес». Точнее, регистр RG3.1 хранит номер блока памяти, в котором находится программа

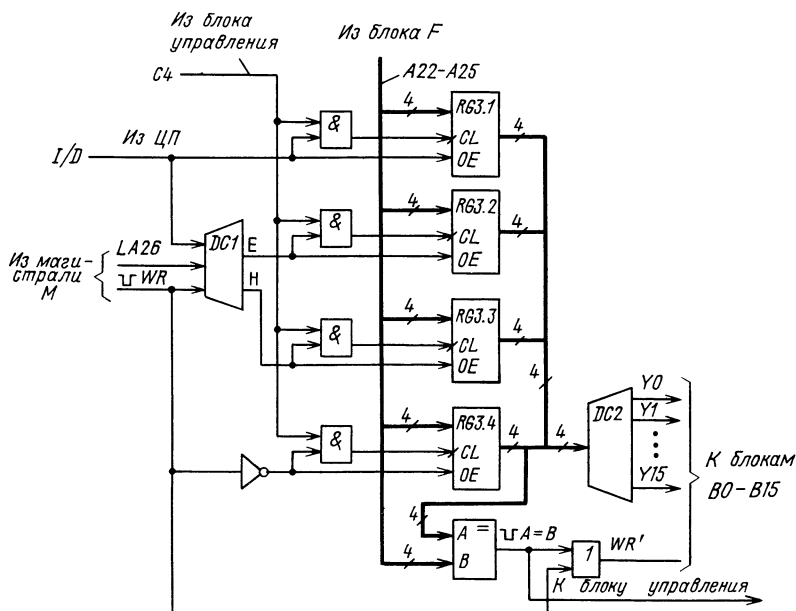


Рис. 3.28. Усовершенствование схемы (рис. 3.27)

(или ее текущий фрагмент); регистр RG3.2 содержит номер блока, в котором размещены глобальные переменные; регистр RG3.3 отображает местонахождение локальных переменных или стека; регистр RG3.4 указывает номер блока, в который в ближайшей предыстории записывались данные.

Как и в предыдущем примере (см. рис. 3.27), запись кода в один из регистров RG3.1—RG3.4 осуществляется сигналом C4 либо в конце цикла обращения к ОЗУ, либо в его начале (при неудачном сравнении кодов компаратором). Регистры выполнены по схеме с «третьим состоянием»; для выдачи содержимого регистра в общую выходную шину на его вход OE подается сигнал лог. 1. Коды в регистрах RG3.1—RG3.4 могут быть разными или совпадающими в любых сочетаниях. Рассмотрим условия выдачи кодов из регистров.

1. Регистр RG3.1 открывается по выходам при поступлении из процессора сигнала  $I/D=1$ , означающего, что из ОЗУ извлекается команда.

2. Выдача кода из регистра RG3.2 осуществляется по сигналу  $E=1$ , который формируется дешифратором DC1 при считывании процессором глобальной переменной. Эта ситуация, в свою очередь, характеризуется следующими условиями:  $I/D=0$  (нет считывания команды),  $WR=1$  (режим чтения),  $LA26=0$ . Последнее условие приведено в качестве примера — возможны иные критерии попадания адреса в область глобальных переменных.

3. Выдача кода из регистра RG3.3 сопровождается операцией считывания процессором локальной переменной или стека. Сигнал  $H=1$  формируется при  $I/D=0$ ,  $WR=1$ ,  $LA26=1$ .

4. Код из регистра RG3.4 поступает в общую выходную шину при записи в ОЗУ любой информации — командной, числовой, адресной и т. п. В этом режиме  $WR=0$ ,  $I/D=0$ , значение сигнала LA26 может быть любым (0 или 1).

Усовершенствованное ОЗУ (рис. 3.28) работает примерно так же, как и основное. Пока логический адрес преобразуется блоком F в физический

(на это требуется время порядка 50 нс), срабатывает цепь «прогнозирования»: дешифратор DC1—регистр RG3<sub>i</sub>—дешифратор DC2 с соответствующими усилителями (не показаны). В результате к моменту формирования физического адреса один из блоков B0—B15 уже подготовлен к работе.

Если прогноз оказался верным, то на выходе компаратора формируется сигнал лог. 0, начатый обмен продолжается. При ошибочном прогнозе обмен отвергается, по сигналу C4 в соответствующий регистр RG3<sub>i</sub> принимается правильный код, после чего начинается «правильный» обмен.

После включения напряжения питания в регистрах RG3.1—RG3.4 устанавливаются произвольные коды, совпадающие или не совпадающие с нужными. В дальнейшем ОЗУ адаптируется к «поведению» процессора, предопределяя его «намерения». Схему можно усовершенствовать и далее, например «размножением» регистра RG3.4 (как это было сделано с регистром RG3, см. рис. 3.27) для более точной квалификации операции записи: запись в стек, запись локальной переменной и т.п.

### 3.9. Определение типа платы, установленной в разъем, без использования дополнительных контактов этого разъема [52]

Вычислительное устройство, фрагмент которого показан на рис. 3.29, содержит основную плату 1 и дополнительную плату 2. На плате 1 размещены элементы, составляющие ядро микроЭВМ,—процессор, ОЗУ, ПЗУ, ряд контроллеров, адаптеров и т.п. (эти элементы на рисунке не показаны). Плата 2 предназначена для расширения функциональных возможностей микроЭВМ и может содержать дополнительную память, дополнительные контроллеры или иные устройства.

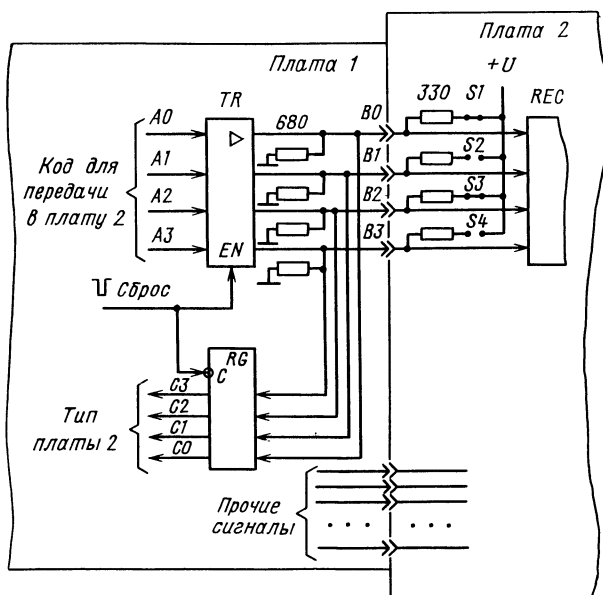


Рис. 3.29. Фрагмент схемы вычислительного устройства, содержащего основную плату 1 и дополнительную плату 2

В зависимости от типа платы 2, установленной в разъем, в микроЭВМ вводятся те или иные дополнительные ресурсы. В данном примере тип платы отображается 4-разрядным двоичным кодом  $C3\ C2\ C1\ C0$ . Если, например,  $C3=1$  (напряжение высокого уровня), то плата 2 содержит дополнительный блок памяти, если  $C3=0$ , то на плате 2 размещен один или несколько контроллеров. Более точные сведения об объеме дополнительной памяти или составе контроллеров содержатся в разрядах  $C2$ ,  $C1$  и  $C0$ .

Для задания кода, определяющего тип платы, обычно используют специально выделенные контакты разъема, на которых со стороны этой платы с помощью микропереключателей или паяных перемычек создают нужную комбинацию сигналов лог. 0 и лог. 1. В предлагаемом решении информация о типе платы передается в основную плату микроЭВМ с использованием тех же контактов разъема, через которые проходит «обычная» информация, например адресная или управляющая.

Схема приводится в исходное состояние при поступлении сигнала лог. 0 на вход Сброс (этот сигнал, в частности, сопровождает включение напряжения питания микроЭВМ). Магистральные усилители TR выключаются, их выходы переходят в состояния с высоким сопротивлением. В данном примере  $B0=1$ ,  $B1=0$ ,  $B2=1$ ,  $B3=0$ , так как в плате 2 перемычки  $S1$  и  $S3$  установлены, а перемычки  $S2$  и  $S4$  отсутствуют. С помощью перемычек задается тип платы 2. Чтобы можно было выделить ситуацию, когда плата 2 вообще не вставлена в разъем, при задании кодов должна быть установлена хотя бы одна перемычка. Регистр-защелка RG при  $C=0$  транслирует входную информацию на выходы, а при переходе сигнала C из 0 в 1 запоминает ее.

По окончании сигнала лог. 0 на входе Сброс, т. е. при переходе этого сигнала из 0 в 1, информация о типе платы запоминается в регистре RG и в дальнейшем используется операционной системой для учета и распределения имеющихся ресурсов. Магистральные усилители TR переходят в рабочее состояние (активный 0 или 1 на выходе) и транслируют входные сигналы  $A0—A3$  в приемник REC, размещенный в плате 2, через те же контакты разъема, которые ранее (на подготовительном этапе) использовались для передачи информации о типе платы.

Сигналы  $A0—A3$  могут представлять собой разряды адреса, приемник REC является, например, дешифратором, регистром или иным устройством. Резисторы (или резисторные делители напряжения, когда установлены перемычки) не препятствуют передаче информации из платы 1 в плату 2, так как усилители TR обладают достаточной нагрузочной способностью.

### **3.10. Изменение функций, выполняемых системой из нескольких микроЭВМ, заменой ППЗУ в одной из этих микроЭВМ [53]**

Многомашинная система управления автомобилем (рис. 3.30) содержит три микроЭВМ, объединенных последовательными каналами связи. Каждая микроЭВМ ориентирована на выполнение некоторого круга задач. В зависимости от типа автомобиля, в котором использована данная система, каждая микроЭВМ должна руководствоваться соответствующей версией программ и данных; эти версии обычно содержатся в кодировках микросхем ПЗУ или ППЗУ. Поэтому на платах микроЭВМ устанавливаются розетки, к которым подключают ПЗУ или ППЗУ с нужной кодировкой. Предлагается исключить такие розетки из всех микроЭВМ, кроме одной, устанавливая ПЗУ на плату без перспектив его замены, сохраняя, однако, возможность адаптации системы к конкретному типу автомобиля.

В данном примере в микроЭВМ2 и 3 установлены ПЗУ с не изменяемой кодировкой (микросхемы ПЗУ впаяны в платы и поэтому не могут быть заменены); в них содержатся все версии программ  $P1—P_N$  и данных  $D1—$

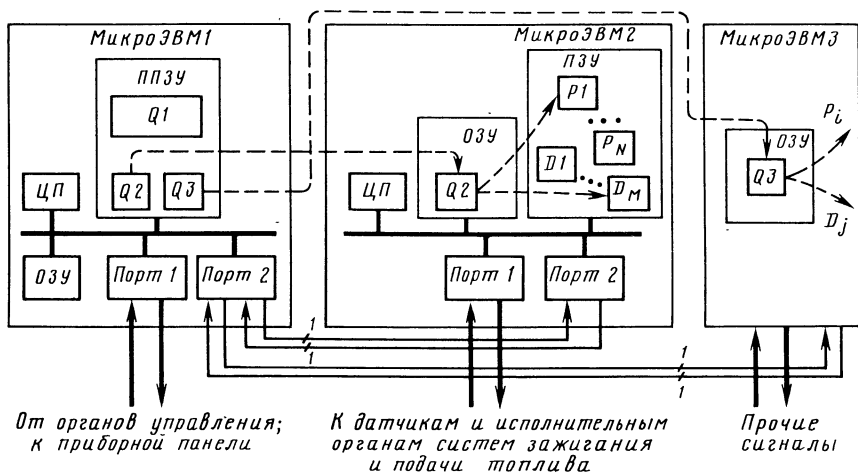


Рис. 3.30. Многомашинная система управления автомобилем. ППЗУ с кодировкой, зависящей от типа автомобиля, находится только в микроЭВМ1, в остальных микроЭВМ использованы масочные ПЗУ с неизменной кодировкой

$D_m$ , так что в зависимости от типа автомобиля достаточно лишь задать некоторую группу программных указателей, определяющих выбор нужной версии. Эти группы представлены в микроЭВМ2 и 3 блоками данных Q2 и Q3, размещенными в ОЗУ. Блок Q2 предписывает процессору микроЭВМ2 пользоваться программой P1 и данными  $D_m$ . Аналогично блок Q3 служит указателем для выбора программы P<sub>i</sub> и данных D<sub>j</sub>.

В микроЭВМ установлено сменное ППЗУ с кодировкой, зависящей от типа автомобиля. В этом ППЗУ имеется блок Q1 программ и данных, которыми пользуется только процессор микроЭВМ1, а также блоки Q2 и Q3, которые после включения напряжения питания системы и ее установки в исходное состояние передаются в ОЗУ микроЭВМ2 и 3. Точнее, события развиваются так.

После включения напряжения питания процессоры микроЭВМ1—3 приступают к выполнению программ начального пуска, хранимых в «своих» микросхемах постоянной памяти (ППЗУ, ПЗУ). Эти программы, в частности, предусматривают начальную установку портов 1 и 2, быстрое тестирование основных функциональных блоков и другие операции, после выполнения которых микроЭВМ 2 и 3 переходят к ожиданию сеансов связи с микроЭВМ1. В ходе последующих сеансов, проводимых по инициативе микроЭВМ1, она передает блоки Q2 и Q3 из своего ППЗУ через порт 2 по последовательным каналам связи в ОЗУ микроЭВМ 2 и 3 (см. штриховые стрелки на рис. 3.30). В соответствии с содержимым блоков Q2 и Q3 процессоры микроЭВМ2 и 3 выбирают нужные программы, данные и приступают к работе.

### 3.11. МикроЭВМ с многоплатной «этажерочной» конструкцией и автоматическим назначением собственных адресов плат [54]

МикроЭВМ (рис. 3.31) содержит основную печатную плату 1 и платы расширения 2—4 (плата 5 показана в процессе ее стыковки с платой 4). На основной печатной плате размещены ЦП, ПЗУ, ОЗУ, минимальный набор контроллеров периферийных устройств, видеоконтроллер и другие устройства.

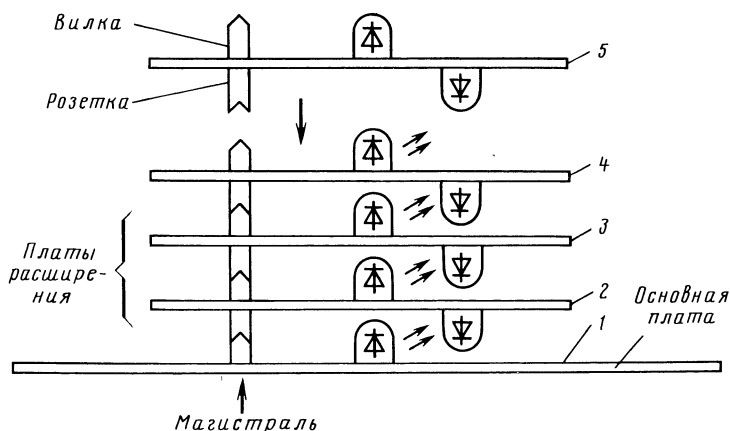


Рис. 3.31. Эскиз «этажерочной» конструкции микроЭВМ с оптической межъярусной дейзи-цепью.

Магистраль наращивается по мере увеличения этажности

На платах расширения могут размещаться дополнительные контроллеры, блоки памяти или иные устройства. Число сочлененных плат расширения может меняться в зависимости от потребностей пользователя.

Для опознавания адресного обращения к плате расширения в ней предусмотрен компаратор К (рис. 3.32). Он сравнивает старшие разряды текущего адреса, передаваемого из процессора (или другого активного устройства), с собственным адресом, хранимым в регистре RG. Загрузка регистров RG плат расширения производится процессором в соответствии с хранимой в ПЗУ программой начального пуска. Эта программа вызывается автоматически после включения напряжения питания микроЭВМ или при воздействии на процессор сигнала системного сброса.

Автоматическое назначение собственных адресов плат расширения избавляет оператора от излишних забот, связанных с ручным их заданием — обычно для этого используются микропереключатели или паяные перемычки на плате. В общих чертах процесс назначения собственных адресов плат расширения сводится к последовательной загрузке регистров RG, причем пары светодиод — фотоприемник соседних плат обеспечивают последовательную активизацию плат снизу вверх.

Сначала включается светодиод платы 1 (см. рис. 3.31) и излучает инфракрасный свет в сторону приемного фотодиода платы 2. Плата 2 таким образом подготавливается к приему собственного адреса в регистр RG. После приема собственного адреса из магистрали выключается светодиод платы 1, процессор выполняет проверочные операции, включается светодиод платы 2, в результате плата 3 подготовлена к загрузке регистра RG, далее этот регистр загружается с магистрали и т. д. Таким образом, используется дейзи-цепочка для последовательного обращения к платам расширения; оптическая связь между платами позволяет (всего лишь!) сэкономить по два контакта на каждом разъеме — по одному входному и одному выходному. Рассмотрим процесс назначения собственных адресов плат расширения более подробно.

При включении напряжения питания или при нажатии соответствующей клавиши формируется сигнал системного сброса. Все регистры RG устанавливаются в 0, платы расширения переходят в пассивное состояние, при котором они неспособны реагировать на адресные обращения со стороны магистрали даже при срабатывании компараторов К; все светодиоды выключены.

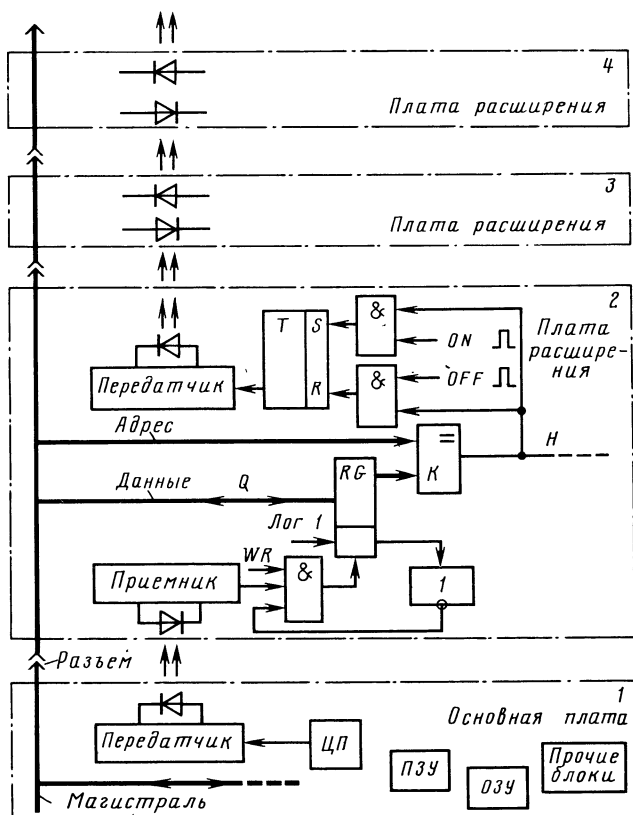


Рис. 3.32. Схема сопряжения основной платы 1 с платами расширения 2—4 (см. рис. 3.31).  
Центральный процессор на этапе начального пуска последовательно записывает в регистры RG плат 2—4 их собственные адреса

После окончания сигнала системного сброса процессор приступает к выполнению программы начального пуска, в которой, в частности, предусмотрена процедура установки собственных адресов плат расширения. Работа по программе сопровождается следующими событиями.

1. Процессор формирует статический сигнал включения передатчика, размещенного на основной плате. Светодиод платы 1 включается, на выходе приемника платы 2 формируется сигнал лог. 1, который поступает на средний вход трехходового элемента И. Плата 2 подготовлена к приему кода в регистр RG.

2. Процессор выполняет команду записи кода (собственного адреса платы 2) по некоторому произвольному адресу. Истинность данных в магистрале подтверждается сигналом записи  $WR=1$ . В плате 2 (только в ней) срабатывает трехходовый элемент И, по положительному фронту сигнала с его выхода в регистре RG фиксируется нужный код. Нижний (добавочный) разряд регистра RG устанавливается в 1, трехходовый элемент И «навсегда» закрывается по нижнему входу. Платы 3 и 4 по-прежнему находятся в пассивном состоянии и не реагируют на магистральные сигналы.

3. Процессор выключает передатчик, размещенный на основной плате, приемник платы 2 выключается. Регистр RG платы 2 хранит записанный в него код, плата 2, в отличие от плат 3 и 4, теперь способна реагировать на адресные обращения по магистрали, если старшие разряды текущего адреса совпадают с записанными в регистр RG (схемы, отражающие переход платы 2 в «работоспособное» состояние, на рисунке не показаны для его упрощения).

4. Процессор считывает (точнее—пытается считать, так как плат расширения может и не быть, но процессор пока об этом не знает) код из регистра RG платы 2, рассматривая этот регистр как обычный адресуемый элемент, занимающий в адресном пространстве известное (программе) положение. При считывании срабатывает компаратор К и вступают в действие элементы, не показанные на рисунке,—дешифратор младших разрядов адреса, буферные магистральные передатчики, схема формирования ответного сигнала и т.п. В результате содержимое регистра RG платы 2 по связям Q передается в магистраль и далее в процессор.

5. Если при считывании регистра RG по п. 4 ответный сигнал от платы 2 не поступил в течение некоторого времени, например равного 10 мкс, то процессор «считает», что ни одна из плат расширения не установлена в микроЭВМ. В этом случае процедура назначения собственных адресов завершается, как не имеющая смысла. Если считывание состоялось и произошло совпадение полученного кода с ожидаемым, то процесс продолжается.

6. Процессор приступает к более детальному исследованию платы расширения 2. Путем опроса ее программно-доступных элементов он определяет тип устройства, размещенного на плате 2 (сверяя полученную при опросе информацию со списком, заложенным в программу), объем его памяти и иные параметры. Процессор может протестировать обнаруженное устройство и убедиться в его работоспособности. При необходимости процессор может изменить собственный адрес платы 2, назначенный ранее, записав в программно-доступный регистр RG платы 2 новый код (запись осуществляется без использования элементов оптической связи—они однократно выполнили свои функции и «навсегда» выключились).

7. Процессор выполняет команду чтения (сигнал  $WR=0$ ) по некоторому адресу, соответствующему входу S триггера Т платы 2. При этом в результате дешифрации младших разрядов адреса формируется сигнал  $ON=1$ , и так как  $N=1$  (есть совпадение старших разрядов), то триггер Т устанавливается в 1, светодиод платы 2 включается. Плата 3 подготовлена к рассмотренному ранее «неадресному» занесению информации в свой регистр RG. (Так как  $WR=0$ , то попутная загрузка регистра RG платы 3 не производится, хотя можно было бы совместить действия по пп. 7 и 8, используя команду записи нужного кода по адресу, соответствующему входу S триггера Т платы 2.)

8. Процессор, выполняя команду записи ( $WR=1$ ) кода по произвольному адресу, загружает регистр RG платы 3. При этом в плате 3 протекают процессы, аналогичные описанным в п. 2.

9. Процессор переводит триггер Т платы 2 в исходное (нулевое) состояние, выполняя операцию чтения или записи по адресу, выделенному в адресном пространстве для входа R этого триггера. В результате дешифрации младших разрядов адреса формируется сигнал  $OFF=1$ , который совместно с сигналом  $N=1$  вызывает воздействие на вход R сброса триггера.

После этого процессор убеждается в том, что плата 3 подключена к микроЭВМ, исследует эту плату и «ставит ее на учет». Затем производится подготовка платы 4 к приему собственного адреса, и описанные процессы повторяются. В данном примере (рис. 3.32) попытка обнаружения платы 5 окажется безуспешной, поэтому процесс назначения собственных адресов на этом завершится.

### 3.12. Повышение эффективности схем опроса постоянной информации, заданной перемычками на плате [55]

Чтобы сообщить процессору микроЭВМ об особенностях системы, где эта микроЭВМ используется, обычно применяют набор микропереключателей или проводных перемычек. В зависимости от типа системы, состава периферийных устройств, скорости передачи информации по каналам связи и других параметров, микропереключатели устанавливаются в определенные положения. Если смена параметров производится редко или эти параметры вообще не меняются на протяжении времени жизни системы, то удобнее установить паяные перемычки между соответствующими парами контактов на печатной плате. При включении напряжения питания или в иных ситуациях процессор тем или иным способом считывает информацию, заданную с помощью перемычек, и принимает ее к сведению при дальнейшем выполнении программ.

Чтобы уменьшить число разрядов входного порта, выделенных для ввода постоянной информации, предлагается использовать разряды выходного порта, как показано на рис. 3.33. Предположим для определенности, что «обрыв» провода на входе A1 микроЭВМ воспринимается ею как сигнал лог. 1 (напряжение высокого уровня, подаваемое от источника питания через внутренний резистор). Тогда схема, приведенная на рис. 3.33, позволит распознать 5 ситуаций, соответствующих отсутствию перемычек S1—S4 или установке одной из них. Распознавание ситуации ведется программно, при последовательной выдаче кодов 001, 010 и 100 на выходы Q1—Q3 и считывании ответных реакций с входа A1. При отсутствии всех перемычек S1—S4 на входе A1 постоянно присутствует сигнал лог. 1. Если установлена перемычка S4, то постоянно считывается сигнал лог. 0. При установке, например, перемычки S2 сигнал лог. 1 считывается с входа A1 только при выдаче на выходы кода 010 и т. д.

Схема, приведенная на рис. 3.33, позволяет уменьшить число входов за счет увеличения числа выходов микроЭВМ. Такое решение целесообразно в тех случаях, когда в наличии имеются незадействованные выходы, а входов мало. Отметим, что выходы Q1—Q3 могут использоваться не только для опроса перемычек S1—S4, но и для других целей. Например, эти выходы могут управлять обмотками реле через усилители. Опрос перемычек сопровождается кратковременным нарушением нормального режима управления,

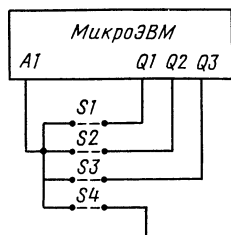


Рис. 3.33. Подключение перемычек к входному и выходному портам микроЭВМ. Использован только один вход, однако задействованы три выхода

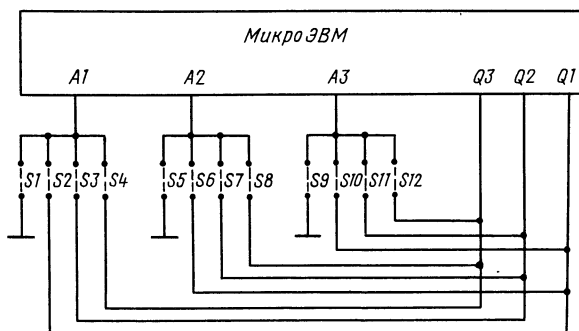


Рис. 3.34. Система с тремя независимыми группами перемычек.

С помощью перемычек S1—S12 можно задать одну из 125 кодовых комбинаций

однако в силу механической инерционности состояние реле (замкнуто—разомкнуто) остается неизменным.

Для повышения информативности или, иными словами, для увеличения емкости внешней постоянной памяти, выполненной на основе паянных перемычек, можно использовать схему, приведенную на рис. 3.34. В каждой группе  $S1—S4$ ,  $S5—S8$ ,  $S9—S12$  может присутствовать не более одной перемычки, чтобы не было замыканий выходов  $Q1—Q3$ . Опрос перемычек производится так же, как и в схеме, приведенной на рис. 3.33, подачей на выходы  $Q1—Q3$  кодов 001, 010 и 100. Число задаваемых пользователем кодовых ситуаций в каждой группе равно 5 (отсутствие перемычек или установка одной из них). Так как все три группы независимы, возможны  $5^3=125$  ситуаций, одна из которых распознаётся процессором микроЭВМ после проведения не более чем трех шагов опроса.

Число  $125_{10}$  можно представить 7 двоичными разрядами. Поэтому схема, показанная на рис. 3.34, по существу эквивалентна традиционной, содержащей 7-разрядный входной порт, каждый разряд которого с помощью индивидуальной перемычки может быть соединен с шиной лог. 0. Сравним информативность традиционного и предлагаемого решений.

Если число выходов  $Q_i$  (рис. 3.34) равно  $N$ , число входов  $A_i$  равно  $M$ , то пользователь может задать  $(N+2)^M$  различных процессором комбинаций, как было показано ранее на примере ( $N=3$ ,  $M=3$ ). В обычной схеме необходим эквивалентный  $L$ -разрядный порт, где  $L$ —минимальная разрядность двоичного кода, отображающего число  $(N+2)^M$ . На входах эквивалентного порта с помощью перемычек можно задать  $2^L$  кодовых комбинаций. Поэтому предлагаемую и традиционную схемы можно (с некоторой точностью) считать равноинформативными, если  $2^L \geq (N+2)^M$ , откуда следует, что  $L \geq M \log(N+2)/\log 2$ .

Дальнейшее увеличение информативности достигается введением интегрирующих RC-цепей и вычислением задержки между выходным и входным сигналами. В схеме, показанной на рис. 3.35, выход  $Q1$  соединен с входом  $A1$  через резисторы  $R2—R4$  и перемычки  $S1—S3$ , которые могут не устанавливаться или устанавливаться в любых сочетаниях. Резисторы  $R2$ ,  $R3$  и  $R4$  должны иметь разное сопротивление. Вход  $A1$ , в отличие от предыдущих примеров, не подключен внутри микроЭВМ к шине питания через резистор, т. е. он «взвешен». Для доопределения сигнала на этом входе в отсутствие перемычек  $S1—S3$  использован высокоомный резистор  $R1$ , сопротивление которого не принимается во внимание при расчете временных параметров сигнала на входе  $A1$ .

В исходном состоянии  $Q1=0$ , конденсатор  $C1$  разряжен. После формирования сигнала  $Q1=1$  и при наличии хотя бы одной перемычки  $S1—S3$  конденсатор заряжается через один, два или три параллельно включенных резистора  $R2—R4$ . В некоторый момент напряжение на входе  $A1$  достигает порога срабатывания элемента—приемника, микроЭВМ фиксирует этот момент и вычисляет время задержки между посылкой сигнала  $Q1=1$  и получением сигнала лог. 1 на вход  $A1$ . По результату вычислений определяется число и позиции имеющихся перемычек. Если сигнал лог. 1 не получен в течение максимально допустимого интервала времени ожидания (этот интервал вычисляется исходя из параметров самой «медленной» RC-цепи), то это означает, что ни одна перемычка  $S1—S3$  не установлена.

Для надежного распознавания позиций перемычек, установленных на плате, соотношения сопротивлений резисторов должны быть выбраны из некоторого оптимального ряда. Для схемы, приведенной на рис. 3.35, таковым является ряд 1,  $\sqrt{2}$ , 2. Это означает, например, что можно установить резисторы  $R2—R4$  сопротивлением соответственно 1, 1,41 и 2 кОм. В зависимости от кода, заданного перемычками, получим дискретизированный ряд эквивалентных сопротивлений (в порядке возрастания): 0,45, 0,59, 0,67, 0,83, 1, 1,41, 2 кОм.

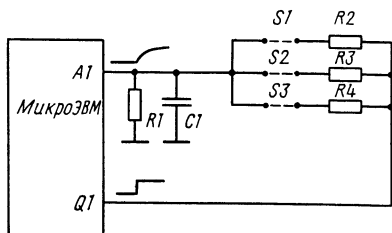


Рис. 3.35. Подключение к микроЭВМ RC-цепи с изменяемой перемычками постоянной времени

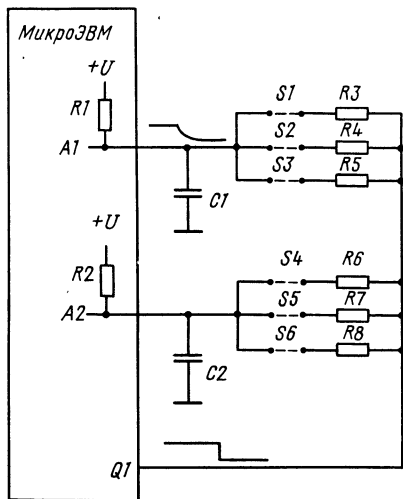


Рис. 3.37. Схема, управляемая сигналом лог. 0 на выходе Q1

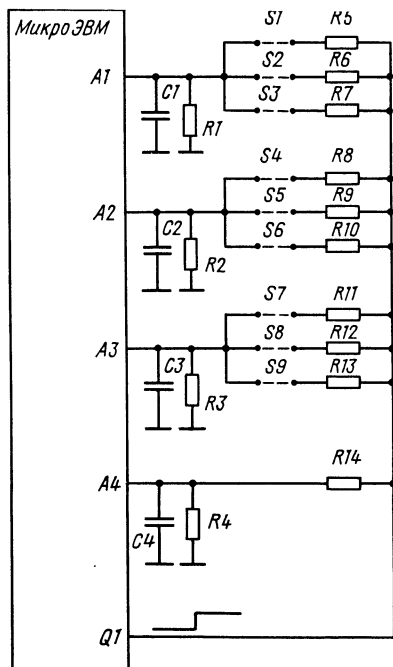


Рис. 3.36. Схема с эталонной RC-цепью (R14C4).

Компенсируется температурный дрейф параметров. Число возможных кодовых комбинаций, задаваемых перемычками S1—S9, равно 512

При изменении температуры пороговое напряжение входа A1 меняется. В меньшей степени меняются параметры RC-цепей. Однако соотношение задержек остается примерно постоянным. Чтобы учесть температурный дрейф параметров, в схему (рис. 3.36) вводится цепь R14C4 с известными параметрами (взята из той же партии компонентов, в которую входят и остальные RC-цепи). Эта цепь опрашивается одновременно с остальными, результат ее опроса служит для вычисления откорректированной в соответствии с температурой условной единицы времени. Использование нескольких независимых входов (A1—A3) позволяет применить резисторы с тремя значениями сопротивлений, как и в схеме рис. 3.35 (хотя возможны  $2^3$ , а не  $2^3$  кодовых ситуаций). Например, можно выбрать сопротивления так:  $R5=R8=R11=1\text{ кОм}$ ,  $R6=R9=R12=1,41\text{ кОм}$ ,  $R7=R10=R13=2\text{ кОм}$ ,  $R14=1,41\text{ кОм}$ .

В схеме, приведенной на рис. 3.37, конденсаторы C1 и C2 исходно заряжены через высокоомные резисторы, Q1=1. При выдаче сигнала Q1=0 и наличии хотя бы одной перемычки в каждой из групп S1—S3 и S4—S6 конденсаторы разряжаются с соответствующими постоянными времени. В остальном схема аналогична рассмотренным ранее.

### 3.13. Способ адресации контроллеров [56]

В системе (рис. 3.38, а) микроЭВМ соединена с четырьмя блоками расширения, каждый из которых содержит до восьми плат контроллеров внешних устройств (эти устройства и соответствующие кабели на рисунке не показаны). Пользователь может устанавливать любую плату в любой блок, между платами могут оставаться не занятые соединители, один или несколько блоков могут вообще не содержать плат. С точки зрения пользователя блоки равноценны и могут подключаться к микроЭВМ и друг к другу в произвольном порядке.

Собственный адрес блока определяется его физическим положением в последовательной цепи (рис. 3.38, б). Блок 1 имеет собственный адрес 11<sub>2</sub>, последующие блоки — адреса 10, 00 и 01. Действительно, если из микроЭВМ на вход цепи поступает код выбора блока  $E1\ E0=11$ , то сигнал опознавания собственного адреса (лог. 1) формируется элементом И первого блока, в остальных блоках элементы И закрыты по крайней мере по одному входу. Аналогично при  $E1\ E0=10, 00$  и 01 сигналы опознавания собственного адреса формируются в блоках 2, 3 и 4.

Блок  $i$  (рис. 3.39) соединен с предыдущим блоком ( $i-1$ ) или микроЭВМ кабелем, содержащим пять адресных линий, по которым слева направо передаются адрес блока ( $F1, F2$ ) и адрес платы в блоке ( $F3-F5$ ), две линии для передачи информационных сигналов  $R$  и  $T$  (направления передачи этих сигналов показаны стрелками) и линию для оповещения соседа слева о присутствии блока  $i$  передачей ему сигнала  $G=0$ . Аналогичные сигналы передаются по кабелю, соединяющему блоки  $i$  и  $i+1$ ; эти сигналы помечены знаками «штрих». Если есть соседний блок справа, то  $G'=0$ , в противном случае  $G'=1$ . (Кабели выполняются в виде наборов витых пар проводов, каждая пара содержит сигнальный и «земляной» провода, однако эти детали на рисунке не показаны для его упрощения.)

Платы 1—8 принимают и передают данные через соответствующие контакты соединителей. Данные могут быть представлены, например, «телеграфным» кодом со стартовым и стоповыми битами. Один из контактов на каждом соединителе выделен для оповещения системы о наличии платы: если она установлена, то  $M_j=0$ , если соединитель свободен, то  $M_j=1$ .

Сигналы  $F1$  и  $F2$  служат для адресации одного из четырех блоков, как было показано на рис. 3.38, б. При обращении к блоку  $i$  на входе выбора

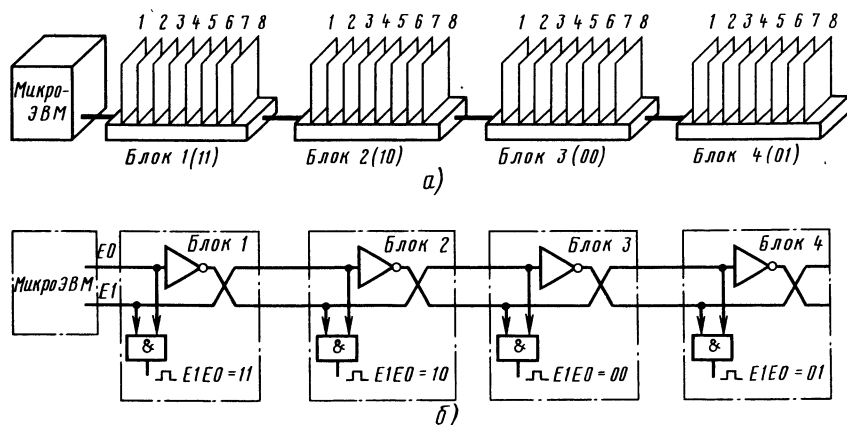
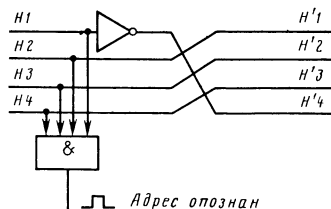


Рис. 3.38. Микропроцессорная система (а) и схема адресации (б) входящих в нее блоков расширения



Рис. 3.40. Схема передачи и опознавания адреса  $i$ -го блока ( $i = 1, 2, \dots, 8$ )



Если блок  $i$  и все его соседи слева не выбраны, а соседи справа нет, то  $G' = 1$ , путь для сквозной передачи сигнала  $R' \rightarrow R$  закрыт. При этом цепь обратной связи замкнута, поэтому сигнал  $T$  проходит через двухвходовые элементы И, ИЛИ, элемент 2И—ИЛИ и возвращается к источнику в виде сигнала  $R$ . Это означает, что микроЭВМ способна выявить отсутствующий блок.

После включения напряжения питания микроЭВМ исследует конфигурацию системы. Для этого она последовательно обращается ко всем 32 возможным адресатам (4 блока по 8 мест в каждом) и по результатам обращений составляет таблицу имеющихся в наличии контроллеров. При обращении к каждому контроллеру микроЭВМ формирует соответствующий 5-разрядный адрес  $F1-F5$  и посылает в линию для передаваемых данных последовательность сигналов  $T$ , отображающую код команды «Кто Вы?». Если по данному адресу размещен некоторый контроллер, то он отвечает на эту команду выдачей комбинации сигналов  $R$ , соответствующей его имени. Если разъем пуст или нужного блока вообще нет, то, как было показано, передаваемая посылка в неизменном виде возвращается к источнику — микроЭВМ получает в ответ эхо-сигнал. После выявления конфигурации системы микроЭВМ приступает к выполнению рабочей программы с учетом имеющихся ресурсов.

Схема передачи и опознавания адреса (рис. 3.40) позволяет создать цепь из восьми контроллеров со следующими собственными адресами: 1111<sub>2</sub>, 1110, 1100, 1000, 0000, 0011, 0111. Введение пятой линии дает возможность адресовать 10<sub>10</sub> контроллеров, шестой — 12 и т. д. Чтобы повысить эффективность использования адресных линий, можно применить более сложные преобразования входного кода в выходной.

### 3.14. Быстрое сохранение — восстановление содержимого регистров процессора при прерываниях программы [57]

Переход от фоновой к прерывающей программе сопровождается сохранением в ОЗУ параметров возврата: содержимого счетчика адреса команды, регистра состояния процессора, регистров общего назначения. При возврате к фоновой программе сохраненные в ОЗУ параметры вновь размещаются в соответствующих регистрах. Подобные процессы протекают также при вызове подпрограмм.

Сохраняемые параметры располагаются в последовательных ячейках памяти (области стека), поэтому адреса этих ячеек можно формировать с помощью аппаратного счетчика, исходно загрузив в него начальный адрес. Тогда шина адреса освобождается и ее можно использовать для расширения шины данных, чтобы ускорить прохождение сохраняемой (восстанавливаемой) информации между процессором и памятью (рис. 3.41).

В данном примере микроЭВМ содержит процессор D1, блок управления D2, приемопередатчики D3, D6, формирователь адреса D4, два блока ОЗУ D5 и D7, а также иные устройства: ПЗУ, контроллеры и т. д. (на рисунке

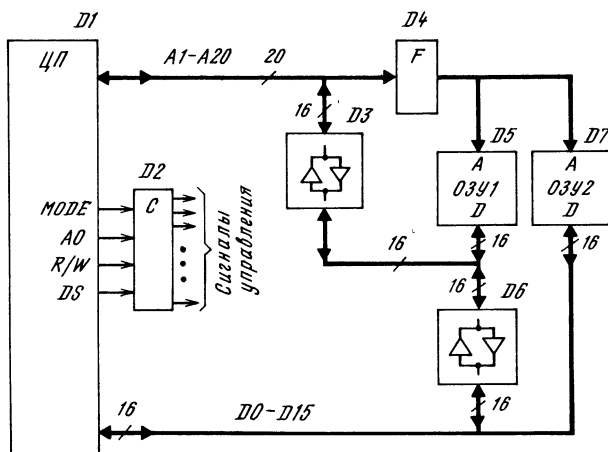


Рис. 3.41. Подключение памяти к процессору

не показаны). Сигнал **MODE** определяет режим работы подсистемы памяти: «нормальный» или стековый.

Первый режим (рис. 3.42) сопровождается отключением приемопередатчиков **D3** (считаем, что их нет) и переводом формирователя адреса **D4** в состояние, при котором он транслирует входные сигналы на выходы без каких-либо преобразований. Приемопередатчики **D6** транслируют данные в нужном направлении, которое задается сигналом **R/W** (чтение—запись). Поэтому можно считать, что этих приемопередатчиков нет, а шина данных просто проходит сквозь них. Каждый цикл обращения процессора к памяти начинается по сигналу **DS**.

Блоки памяти **O3Y1** и **O3Y2** выполнены на микросхемах динамического типа, например имеющих структуру  $1M \times 1$  бит ( $1M = 2^{20}$ ). Каждый блок содержит 16 таких микросхем и имеет адресные мультиплексоры (не показаны) для преобразования 20-разрядного адреса в последовательность из двух 10-разрядных, определяющих строку и столбец накопителя, на пересечении которых лежит выбираемая ячейка. (Более полные сведения о работе микросхем динамической памяти имеются в справочниках, поэтому здесь не приводятся.)

Младший разряд **A0** полного 21-разрядного адреса, поступающего из процессора, воспринимается блоком управления **D2**. Если **A0=0**, то выбирается **O3Y1**, если **A0=1** — то **O3Y2**. Таким образом, 16-разрядные слова с четными адресами хранятся в **O3Y1**, с нечетными — в **O3Y2**. Это известный способ «расслоения» адресов, позволяющий частично распараллелить работу **O3Y1** и **O3Y2** при обращении к ячейкам памяти с последовательно возрастающими (убывающими) адресами. Пока **O3Y1** завершает обмен (этот процесс при использовании микросхем динамической памяти требует определенных затрат времени), **O3Y2** включается в работу, и наоборот.

Второй режим обеспечивает ускоренный обмен данными между процессором и областью стека, выделенной в **O3Y**. Начальный или конечный адрес области (при записи или считывании), соответствующий первому обращению к стеку (за исключением разряда **A0**), запоминается в формирователе **D4** (рис. 3.43). Предположим сначала, что начальный адрес области стека четный, число 16-разрядных ячеек, образующих стек, также четно, а формирователь **D4** представляет собой двоичный реверсивный счетчик.

После фиксации в счетчике **D4** граничного адреса области стека блок управления **D2** формирует последовательность сигналов, в соответствии

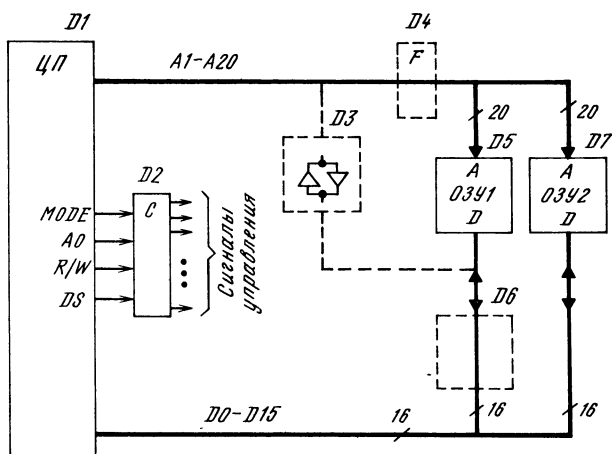


Рис. 3.42. Структура микроЭВМ при работе в «нормальном» режиме.  
Шина адреса используется по прямому назначению

с которыми вступают в действие приемопередатчики D3, а элементы D6 фактически исключаются из схемы. Блоки ОЗУ1 и ОЗУ2 в каждом цикле обмена активизируются одновременно (а не поочередно, как в «нормальном» режиме); адрес у них общий, а данные передаются по индивидуальным шинам. Иными словами, шина адреса используется в качестве шины данных блока ОЗУ1, а «штатная» шина данных обслуживает блок ОЗУ2.

Благодаря параллельному включению блоков ОЗУ1 и ОЗУ2 процессор в каждом цикле записи передает в память два 16-разрядных слова (по шинам адреса и данных); счетчик D4 последовательно увеличивает адресный код на единицу младшего разряда. Считывание массива данных из области

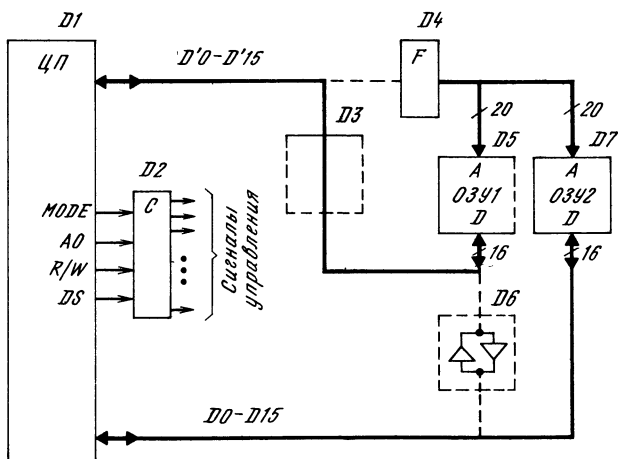


Рис. 3.43. Структура микроЭВМ при сохранении — восстановлении содержимого регистров процессора.

По шине адреса передаются данные, ОЗУ1 и ОЗУ2 включены параллельно

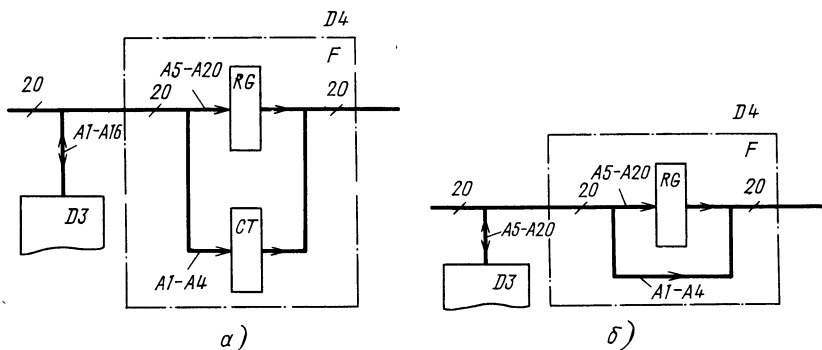


Рис. 3.44. Варианты построения формирователя F (D4).  
Линии управления не показаны

стека протекает аналогично, счетчик D4 работает в режиме вычитания единицы из текущего содержимого. Таким образом, сохранение и восстановление содержимого регистров при прерываниях программы выполняется вдвое быстрее, чем обычно.

Если начальный адрес области стека нечетный (первая ячейка этой области находится в ОЗУ2), то, прежде чем перейти к ускоренной записи, процессор выполняет одну пересылку в обычном режиме, чтобы затем стартовать с четного адреса. Примерно то же относится и к конечному адресу—если он четный, т.е. последняя ячейка области стека находится в ОЗУ1, то завершающая запись выполняется только в это ОЗУ в обычном режиме, чтобы не искалось содержимое одноименной ячейки в ОЗУ2. Эти особенности несколько снижают эффективность данного решения, поэтому при разработке программного обеспечения целесообразно выбирать начальный адрес области стека четным, а конечный—нечетным.

Чтобы упростить структуру формирователя D4, можно принять следующее соглашение между разработчиками аппаратуры и программного обеспечения: во-первых, начальный и конечный адреса области стека должны быть соответственно четным и нечетным (этот вопрос только что обсуждался); во-вторых, при последовательном переборе адресов области стека не должно быть переноса (заема) между разрядами A4 и A5. Тогда разряды A5—A20 в процессе счета остаются неизменными и вместо 20-разрядного можно применить 4-разрядный счетчик и 16-разрядный регистр-защелку (последний по сравнению со счетчиком имеет значительно более простую структуру), как показано на рис. 3.44, а. В данном примере в качестве шины данных для ОЗУ1 используются линии A1—A16 шины адреса.

В схеме, показанной на рис. 3.44, б, младшие разряды адреса A1—A4 генерируются процессором и передаются в память, минуя регистр RG; переноса (заема) между разрядами A4 и A5, как предполагаем, нет. В качестве шины данных для ОЗУ1 используются линии A5—A20 шины адреса.

### 3.15. Схема сопряжения между двумя микроЭВМ с использованием параллельных портов ввода—вывода [58]

В двухмашинной системе (рис. 3.45) микроЭВМ1 главная, а микроЭВМ2 подчиненная. Каждая из них содержит порт ввода—вывода для двунаправленной передачи данных D по многоразрядной шине. Ввод данных в микроЭВМ сопровождается управляющим сигналом чтения ( $RD=1$ ), вывод—

сигналом записи ( $WR=1$ ). Один из разрядов  $F$  выходного порта микроЭВМ2 соединен с одноименным разрядом входного порта микроЭВМ1. После окончания сигнала  $Сброс=0$  обе микроЭВМ приступают к выполнению своих программ, триггер  $TT$ , включенный в режиме делителя частоты на 2, установлен в 1.

Межмашинный обмен данными возможен только по инициативе микроЭВМ1. Сеанс связи начинается с двукратной записи слова данных из микроЭВМ1 в регистр  $RG1$  (рис. 3.46, 3.47). Точнее, в момент  $t_0$  выполняется первая команда записи, а в момент  $t_1$  — вторая (см. блок  $A1$ ). При выполнении каждой из этих команд формируется положительный импульс  $WR=1$ . По фронту этого импульса данные принимаются в регистр  $RG1$ . Первый импульс проходит через элемент ИЛИ и переводит триггер в 0, второй возвращает его в исходное состояние. Таким образом, на выходе триггера формируется короткий сигнал прерывания  $INT=0$ .

Процессор микроЭВМ2 реагирует на отрицательный фронт сигнала  $INT$  (переход из 1 в 0). Выполнение текущего задания (блок  $B1$ ) временно откладывается, процессор сохраняет в ОЗУ параметры возврата к нему и переходит к прерывающей программе (см. левую вертикальную стрелку на рис. 3.47). Первая команда этой программы (момент  $t_2$ ) предусматривает проверку состояния сигнала  $INT$ , который может опрашиваться программно, как обычный разряд входного порта (блок  $B2$ ). В данном случае  $INT=1$  — пока процессор микроЭВМ2 переходил к прерывающей программе, триггер успел вернуться в исходное состояние. Поэтому микроЭВМ2 принимает к сведению, что прерывание соответствует начальному взаимодействию, программный счетчик  $N$  устанавливается в 0 (блок  $B3$ ).

Далее микроЭВМ2 считывает данные из регистра  $RG1$  (блок  $B4$ ). При этом по сигналу  $RD=1$  выходные усилители регистра  $RG1$  переводятся из выключенного в активное состояние. Затем микроЭВМ2 записывает данные в регистр  $RG2$  сигналом  $WR=1$  (блок  $B5$ ), выполняет прочие действия (блок  $B6$ ), прибавляет единицу к счетчику  $N$  числа полученных сигналов прерывания (блок  $B7$ ), изменяет значение сигнала  $F$  (блок  $B8$ , момент  $t_3$ ) и возвращается к прерванному заданию (блок  $B9$ ).

МикроЭВМ1 после момента  $t_1$  выполняет фрагменты основной программы и периодически опрашивает состояние сигнала  $F$  (блоки  $A2$ ,  $A3$ ). После обнаружения изменения сигнала  $F$  она считывает данные из регистра  $RG2$  (блок  $A4$ , момент  $t_4$ ), при этом импульс  $RD=1$  переводит триггер в 0, и в момент  $t_5$  процессор микроЭВМ2 вновь приступает к выполнению прерывающей программы. На этот раз в момент проверки сигнал  $INT=0$ , поэтому счетчик  $N$  не устанавливается в 0 и отражает тот факт, что сеанс связи начат ранее, а сейчас он продолжается. МикроЭВМ1 вновь работает по основной программе и периодически опрашивает состояние сигнала  $F$  (блоки  $A5$ ,  $A6$ ).

В момент  $t_6$  сигнал  $F$  меняет значение в связи с окончанием работы микроЭВМ2 по прерывающей программе. МикроЭВМ1, обнаружив это изменение (блок  $A6$ ), записывает код в регистр  $RG1$  (блок  $A7$ , момент  $t_7$ ) и считывает код из регистра  $RG2$  (блок  $A8$ , момент  $t_8$ ). В результате сигнал  $INT$  дважды меняет значение, микроЭВМ2 вновь переходит в режим прерывания (момент  $t_9$ ) и по содержимому счетчика  $N$  определяет стадию развития сеанса связи. Далее процессы протекают аналогично, в соответствии с некоторым протоколом взаимодействия. В момент  $t_{15}$  сеанс связи завершается.

Порядок взаимодействия может, конечно, отличаться от описанного. Например, вместо двукратной записи можно использовать двукратное чтение, чтение — запись или запись — чтение. Во избежание «бесконечного» закликивания в блоках  $A2$ — $A3$  и  $A5$ — $A6$  при отсутствии ответа от микроЭВМ2 в программу вводятся счетчики циклов и т. п. Возможны также изменения в структуре прерывающей программы.

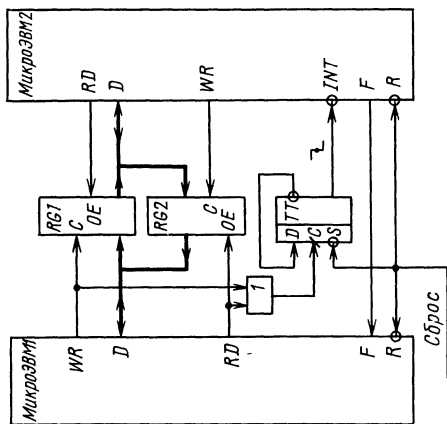


Рис. 3.45. Двухмашинная система

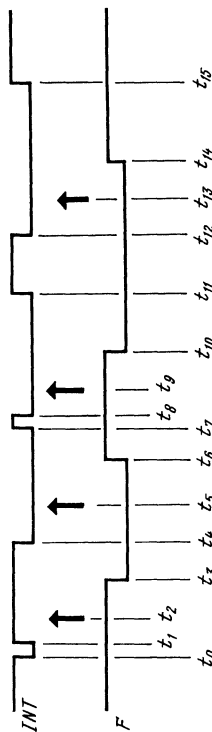


Рис. 3.47. Временные диаграммы работы системы (рис. 3.45). Стрелками показаны моменты опроса сигнала INT при вхождении микроЭВМ2 в прерывающую программу

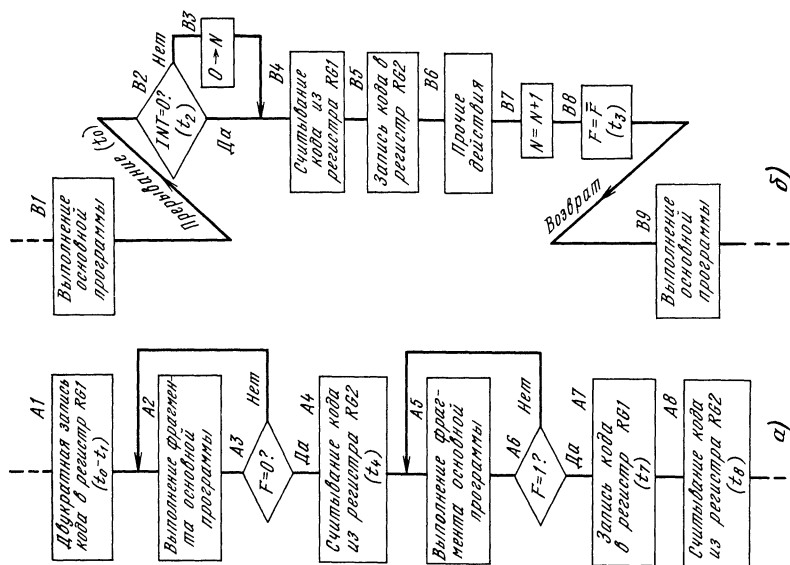


Рис. 3.46. Блок-схемы алгоритмов работы микроЭВМ1 (а) и микроЭВМ2 (б) при их взаимодействии. В скобках указаны моменты времени в соответствии с рис. 3.47

Предлагаемая схема сопряжения позволяет подчиненной микроЭВМ выделять признак начала сеанса связи без использования дополнительных управляющих линий. Кроме того, можно контролировать работу подчиненной микроЭВМ слежением за сигналом F.

### 3.16. Повышение производительности многопроцессорной системы при замене общей двунаправленной шины двумя однонаправленными [59, 60]

Многопроцессорная система (рис. 3.48) содержит процессоры ЦП<sub>1</sub>—ЦП<sub>N</sub> и блоки памяти ОЗУ<sub>1</sub>—ОЗУ<sub>M</sub>, объединенные однонаправленными шинами 1 и 2. Каждый процессор может обмениваться информацией с любым блоком памяти. Арбитр шины 1 устраняет конфликты при попытке одновременного обращения нескольких процессоров к этой шине. Аналогично арбитр шины 2 устраняет конфликты при попытке одновременного использования этой шины несколькими блоками памяти.

Обмен информацией между ЦП<sub>i</sub> и блоком памяти ОЗУ<sub>j</sub> начинается с формирования этим процессором адреса (номера) блока памяти ОЗУ<sub>j</sub> в линиях А<sub>1</sub>, соединенных с адресными входами арбитра шины 1. Одновременно с этим ЦП<sub>i</sub> формирует запросный сигнал, который также поступает в арбитр шины 1. Арбитр принимает запросный сигнал и проверяет, свободен ли блок памяти ОЗУ<sub>j</sub> (информация об этом поступает в арбитр в виде соответствующего сигнала занятости ОЗУ). Если блок ОЗУ<sub>j</sub> не занят, т. е. закончил выполнение ранее полученного задания, то запросный сигнал от процессора принимается арбитром к обработке. В противном случае обработка этого запросного сигнала откладывается до момента освобождения адресата — блока памяти ОЗУ<sub>j</sub>.

Проверка занятости адресата осуществляется арбитром в начале каждого цикла арбитража — ситуация может постоянно меняться, блок, ранее считавшийся свободным, может при очередной проверке оказаться занятым и наоборот.

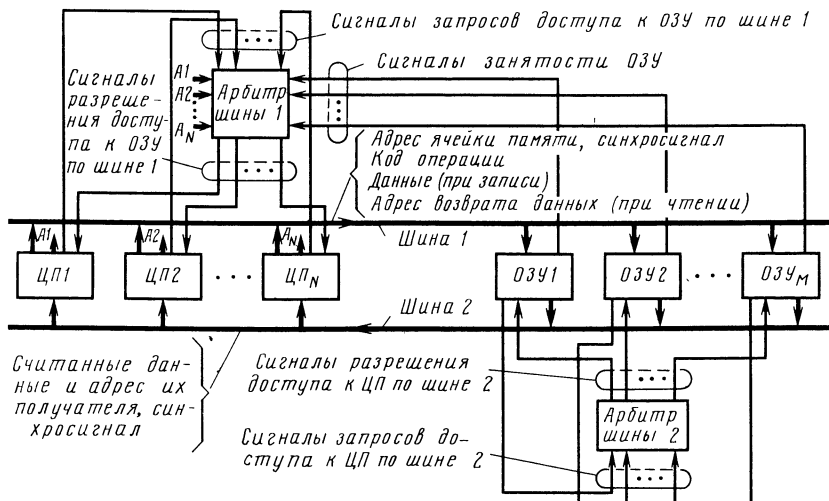


Рис. 3.48. Многопроцессорная система с шинами 1 и 2 прямой и обратной передачи данных при взаимодействии процессоров ЦП<sub>1</sub>—ЦП<sub>N</sub> с блоками памяти ОЗУ<sub>1</sub>—ОЗУ<sub>M</sub>

Запросные сигналы от разных процессоров, принятые к обработке в данном цикле арбитража, как обычно, подвергаются приоритетному отбору, чтобы одновременно несколько процессоров не могли овладеть шиной 1. Приоритеты процессоров могут быть одинаковыми (например, при равновероятном выборе «победителя») или разными— всё определяется требованиями к конкретной системе. В данном примере, возможно, запрос от ЦП<sub>i</sub> не будет удовлетворен в текущем цикле работы арбитра, однако (при правильно спроектированной системе) в дальнейшем должен наступить момент, когда в ответ на этот запрос арбитр сформирует на соответствующем выходе сигнал разрешения доступа к ОЗУ по шине 1.

Начиная с этого момента, ЦП<sub>i</sub> по праву считает себя единственным владельцем шины 1. Он выдает в эту шину следующую информацию: адрес ячейки памяти, к которой предстоит обращение, синхросигнал, подтверждающий истинность адреса, код операции (чтение, запись слова, запись байта, групповое чтение и т. п.), данные (при записи слова или байта) и адрес возврата данных (при чтении), т. е. номер процессора, который ожидает получения считанных данных.

Отметим, что номер процессора— приемника данных может совпадать или не совпадать с номером процессора— источника команды чтения. В последней ситуации процессор— источник команды чтения инициирует пересылку считанного из ОЗУ слова данных в другой процессор, который, как предполагается, находится в режиме ожидания этих данных. Это позволяет синхронизировать процессы, протекающие в системе.

Информация, выданная ЦП<sub>i</sub> в шину 1, пересылается по этой шине и запоминается во входном регистре блока памяти ОЗУ<sub>j</sub> (на рисунке эти регистры не показаны). Начиная с этого момента, блок памяти ОЗУ<sub>j</sub> считается занятым, что подтверждается соответствующим сигналом, передаваемым на вход арбитра шины 1. Таким образом, пересылка информации по шине 1 занимает достаточно короткий фиксированный интервал времени, например, длительностью 40 нс, после чего ЦП<sub>i</sub> освобождает шину 1, снимая сигнал запроса, а арбитр посылает сигнал разрешения доступа к ОЗУ процессору— очереднику (напомним, что абонент должен быть свободен).

Далее блок памяти ОЗУ<sub>j</sub> выполняет полученную команду в автономном режиме. Шина 1 используется в это время для пересылки информации от процессоров к незанятым блокам памяти. Если блок памяти ОЗУ<sub>j</sub> выполнял операцию записи данных в ячейку памяти, то после окончания этой операции сигнал занятости блока снимается, арбитр шины 1 принимает к сведению, что блок памяти ОЗУ<sub>j</sub> освободился, и если к нему адресованы запросы процессоров, то в ближайшем цикле работы арбитра эти запросы принимаются к приоритетной обработке, как было описано ранее.

Если блок памяти ОЗУ<sub>j</sub> выполнял операцию считывания данных из ячейки памяти, то считанное слово должно быть направлено в один из процессоров в соответствии с указанным в команде адресом возврата данных. Прежде чем воспользоваться шиной 2 для передачи данных в процессор-приемник, блок памяти ОЗУ<sub>j</sub> формирует сигнал запроса, который обрабатывается арбитром шины 2 и возвращается в блок памяти в виде сигнала разрешения доступа к процессору.

После этого считанное слово, адрес процессора— получателя данных и сопровождающий синхросигнал передаются по шине 2 адресату, затем сигнал запроса от блока памяти снимается, шина 2 освобождается, арбитр этой шины обслуживает запрос от очередного блока памяти.

В данном примере арбитр шины 2 не проверяет готовность процессора— адресата к приему данных; предполагается, что такая готовность имеется. Если возможна иная ситуация (например, когда адрес возврата данных не совпадает с адресом источника команды, как было описано ранее), то арбитр шины 2 должен анализировать состояние абонента, прежде чем принимать запрос к обработке, как это сделано в арбитраже шины 1 (иначе возникнут потери информации, которая просто отвергается приемником).

При выполнении команды групповой пересылки данных блок памяти ОЗУ<sub>j</sub> многократно захватывает шину 2 и пересылает абоненту некоторую группу слов, постоянно находясь в состоянии «занят». Длина группы оговаривается заранее при проектировании системы либо задается в команде, передаваемой по шине 1.

### 3.17. Бесконфликтный обмен данными в многопроцессорной системе с распределенным управлением [61]

В системе (рис. 3.49), содержащей  $N$  процессоров, обмен информацией между ними осуществляется по  $N$  линиям  $TR_1—TR_N$ . Разрядность линий может быть произвольной, в частности линия может быть представлена одним сигнальным проводом, при этом все процессоры объединены общей шиной нулевого потенциала, информация по линии передается последовательным кодом. Если линия многоразрядная, то информация передается по ней параллельным кодом. Возможны промежуточные варианты с последовательной передачей малоразрядных параллельных кодов, образующих результирующий многоразрядный код.

К каждой линии подключен один источник и  $N-1$  приемник информации. После включения напряжения питания все процессоры выполняют программы начального пуска, принимают к сведению собственные номера (в данном примере  $1—N$ ), заданные, например, микропереключателями на платах, и переходят к выполнению рабочих программ, хранимых в индивидуальных блоках памяти (на рисунке блоки памяти не показаны). В соответствии с распределением функций между процессорами в системе часть из них переходит в состояние пассивного ожидания информации от других, которые, в свою очередь, проявляют определенную активность и выражают намерение обменяться информацией с другими процессорами.

В дальнейшем при работе системы первоначально пассивные процессоры могут становиться инициаторами обмена информацией и наоборот—все определяется принятым алгоритмом функционирования. Рассмотрим меж-процессорное взаимодействие с позиции пассивного (рис. 3.50) и активного (рис. 3.51) процессоров.

Пассивный процессор с номером  $j$  в исходном состоянии циклически опрашивает входные линии  $TR_k$  ( $k=1, 2, \dots, N, k \neq j$ ), ожидая вызова со стороны какого-либо активного процессора. Это отражено блоками 1—3 на рис. 3.50. (Если линии одноразрядные, то процессор опрашивает не сами линии, а приемные сдвиговые регистры, в которых последовательные посылки по линии преобразуются в параллельные коды.) При обнаружении кода «Вызов процессора  $j$ » в линии  $TR_k$  процессор  $j$  посылает в линию  $TR_j$  код «Процессор  $k$ , вызывали меня?» для уточнения ситуации во избежание недоразумений.

Если вызов не был ложным, то в течение допустимого периода ожидания (блок 6) процессор  $k$  (инициатор обмена) устанавливает в линии  $TR_k$  код «Да, я вызывал Вас, процессор  $j$ » (блок 5). После получения этого кода осуществляется обмен информацией между процессорами  $j$  и  $k$  по линиям

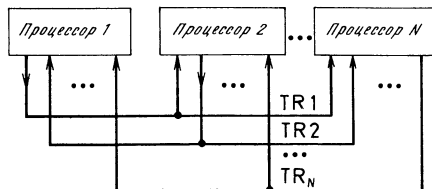


Рис. 3.49. Многопроцессорная система

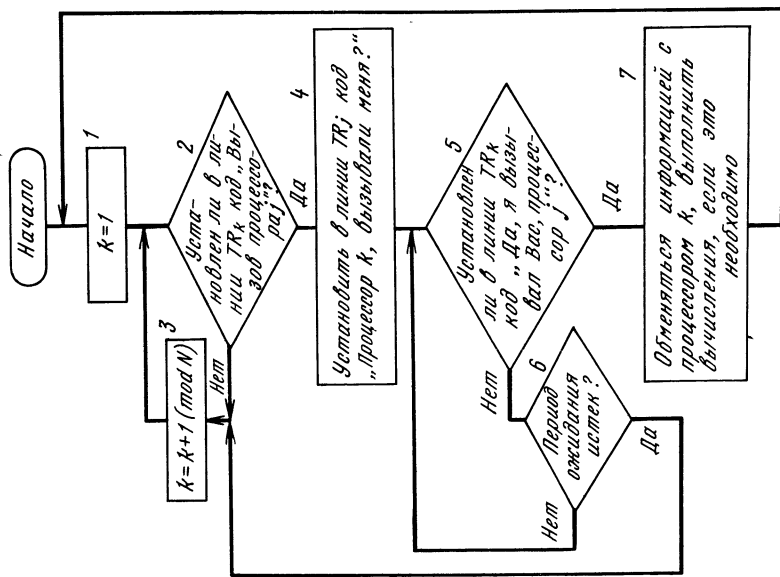


Рис. 3.50. Блок-схема алгоритма работы процессора j, вызывающего на связь процессором k

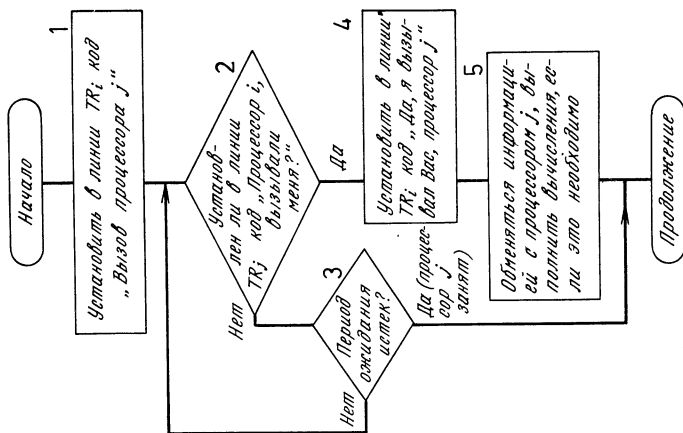


Рис. 3.51. Блок-схема алгоритма работы процессора i (инициатора обмена информацией) с процессором j

$TR_j$  и  $TR_k$  согласно принятому в системе протоколу обмена, который в данном случае не представляет для нас интереса. По завершении обмена процессор  $j$  может выполнить полученное от процессора  $k$  задание или предпринять иные программно-определяемые действия (блок 7), затем он возвращается к циклическому опросу входных линий. Если вызов оказался ложным или активный процессор «передумал» и преднамеренно отказался от сеанса связи (например, выполняя более приоритетное задание), то по истечении допустимого периода ожидания (блок 6) пассивный процессор продолжает циклический опрос входных линий.

Активный процессор  $i$  начинает сеанс связи с пассивным  $j$  (рис. 3.51), устанавливая в линии  $TR_i$  код «Вызов процессора  $j$ » (блок 1). После этого процессор  $i$  ожидает от процессора  $j$  передачи по линии  $TR_j$  ответного кода «Процессор  $i$ , вызывали меня?» (блок 2). Если в течение допустимого периода ожидания (блок 3) этот код получен, то процессор  $i$  устанавливает в линии  $TR_i$  код «Да, я вызывал Вас, процессор  $j$ » (блок 4). Если период ожидания истек, то процессор  $i$  считает, что процессор  $j$  занят, и продолжает работу, откладывая сеанс связи с процессором  $j$  на будущее. После установления связи (блок 4) процессоры  $i$  и  $j$  обмениваются информацией (блок 5), затем процессор  $i$  продолжает работу или переходит в пассивное состояние и следит за сигналами во входных линиях (см. рис. 3.50). В системе возможно одновременное взаимодействие между процессорами, временно объединившимися в независимые пары.

### **3.18. Арбитраж магистрали с предоставлением права исключительного пользования ею устройствам, проявляющим повышенную активность [62]**

Рассмотрим микропроцессорную систему, показанную на рис. 3.52, предполагая сначала, что формирователь импульса  $F$  исключен из схемы. Система, как обычно, построена из объединенных общей магистралью активных и пассивных устройств. Для устранения конфликтов, связанных с попыткой использования магистрали одновременно несколькими активными устройствами, применен арбитр. Он принимает сигналы запросов от активных устройств  $A_1—A_N$ , анализирует их и формирует сигнал разрешения работы с магистралью, адресуя его одному из этих устройств.

Выбор устройства-победителя в конкурентной борьбе осуществляется арбитром в соответствии с некоторым алгоритмом. Например, приоритеты запросов могут убывать, возрастать (или попеременно и то, и другое) с увеличением номера устройства ( $1—N$ ). Однако, каким бы ни был алгоритм, он однозначно определяет активное устройство  $A_1—A_N$ , которому в данном цикле предоставляется возможность захватить магистраль и воспользоваться ею для передачи одного информационного слова пассивному устройству  $B_1—B_M$  или от него. После получения сигнала Ответ от адресуемого абонента активное устройство—инициатор обмена информацией—освобождает магистраль, арбитр вновь анализирует группу запросных сигналов и т. д. Рассмотрим работу системы подробнее, чтобы стали понятны ее недостатки и проявилась сущность предлагаемого далее усовершенствования.

В момент  $t_0$  (рис. 3.53) активное устройство  $A_1$  формирует сигнал Запрос  $A_1$  (напряжение низкого уровня, лог. 0), желая получить разрешение на использование магистрали. Арбитр, обнаружив поступление запроса, запоминает во внутреннем регистре (на рисунке регистр не показан) все входные запросные сигналы и приступает к их анализу. Новые («опоздавшие») запросные сигналы не принимаются к рассмотрению до момента  $t_3$ . К моменту  $t_1$  арбитр заканчивает анализ группы входных запросных сигналов и формирует сигнал Разрешение  $A_1$ .

Получив этот сигнал, активное устройство  $A_1$  занимает магистраль и обменивается информацией с выбранным по адресу пассивным устройством—

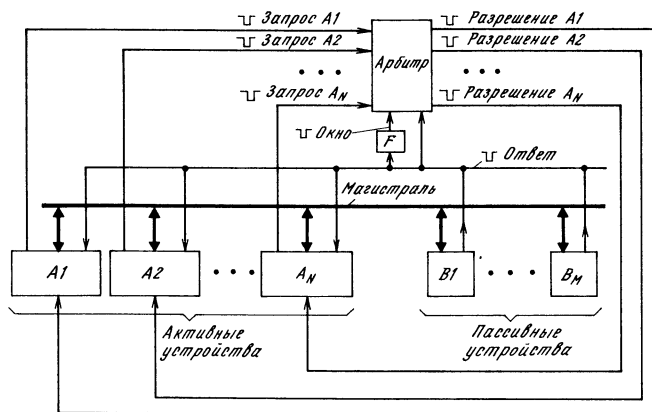


Рис. 3.52. Микропроцессорная система

абонентом (см. заштрихованную область  $t_1—t_3$  на нижней диаграмме, рис. 3.53). В момент  $t_2$  устройство A2 устанавливает сигнал Запрос A2=0, однако, как отмечалось, арбитр не воспринимает его до окончания текущего цикла работы. В момент  $t_3$  активное устройство A1 освобождает магистраль (о чем свидетельствует окончание сигнала Ответ от абонента) и снимает сигнал Запрос A1. Арбитр снимает сигнал Разрешение A1, открывается по входам, обнаруживает новый запрос (старый только что снят) на входе, запоминает все входные сигналы во внутреннем регистре и приступает к их анализу, как и в период  $t_0—t_1$ .

К моменту  $t_5$  арбитр заканчивает анализ группы входных запросных сигналов и формирует сигнал Разрешение A2. В момент  $t_4$  активное устройство A1 вновь сформировало сигнал Запрос A1, однако этот сигнал не успел восприняться арбитром, так как он закрылся по входам на время проведения второго цикла работы (период  $t_3—t_6$ ).

В период  $t_5—t_6$  магистраль используется устройством A2, в период  $t_6—t_7$  арбитр анализирует новую ситуацию на запросных входах, в период  $t_7—t_9$  магистраль занята устройством A1. В момент  $t_8$  формируется сигнал Запрос A2, который воспринимается арбитром после момента  $t_9$ , в момент  $t_{10}$  формируется сигнал Запрос A1 и т. д.

Из нижней временной диаграммы (рис. 3.53) следует, что в течение относительно больших периодов времени  $t_3—t_5$ ,  $t_6—t_7$  магистраль не занята, так как арбитр передает управление стоящему на очереди активному устройству. С другой стороны, из верхней временной диаграммы видно, что устройство A1 намерено осуществить серию обменов информацией с абонентом без сколько-нибудь значительных внутренних задержек — сняв текущий запрос в момент  $t_3$ , оно в момент  $t_4$  формирует новый (то же — в моменты  $t_9$  и  $t_{10}$  и далее). Такую серию быстрых обменов информацией обычно называют блочными или групповыми пересылками данных. При блочных пересылках целесообразно предоставить активному устройству возможность монопольного владения магистралью, чтобы при передаче каждого слова не тратить время на ожидание предоставления разрешения. Это в конечном счете увеличивает интенсивность потока данных по магистрали, как будет показано далее.

В предлагаемой схеме (рис. 3.52 с учетом подключения формирователя импульса F) арбитр, наблюдая за поведением активного устройства A1—A<sub>N</sub>, анализирует его стремление к осуществлению блочной пересылки данных. Если, по мнению арбитра, активное устройство готово работать в режиме



работу в режиме блочной пересылки данных. Если устройство не успело сформировать новый запрос в период, когда сигнал Окно=0, то оно считается «медленным» и обслуживается в следующем цикле на общих основаниях в соответствии с принятой приоритетной структурой. Иными словами, арбитр оценивает возможность экономии магистрального времени и, если такая возможность имеется, передает магистраль в исключительное пользование «быстрому» устройству.

В примере, приведенном на рис. 3.54, устройство А1 успевает сформировать повторный запрос в момент  $t_4$ , когда сигнал Окно=0. Поэтому арбитр прекращает начатый «нормальный» анализ имеющихся конкурирующих запросов и формирует «льготный» статический сигнал разрешения работы устройства А1 (момент  $t_6$ ). В момент  $t_4$  арбитр формирует внутренний сигнал Монопольный режим=0, отмечая факт обнаружения «быстрого» устройства.

В момент  $t_6$  устройство А1 занимает магистраль и проводит второй обмен данными с адресуемым абонентом. В момент  $t_7$  формируется сигнал Ответ, в момент  $t_8$  магистраль освобождается, запрос от устройства А1 снимается, формируется второй сигнал Окно=0. В момент  $t_9$  устройство А1 формирует третий запросный сигнал и, обнаружив, что разрешение имеется, приступает к работе без потерь времени (наконец-то проявилась суть предлагаемого решения!). Далее (моменты  $t_{10}$ — $t_{16}$ ) процессы развиваются аналогично.

В момент  $t_{16}$  формируется очередной сигнал Окно=0. На этот раз устройство А1 не подтвердило желание работать на льготных условиях (блочный обмен завершился, началась нормальная будничная работа). Поэтому в момент  $t_{17}$  арбитр, не дождавшись формирования очередного сигнала Запрос А1=0, снимает статический сигнал разрешения работы устройства А1 и переходит из монопольного режима в обычный. Сравнивая временные диаграммы на рис. 3.53 и 3.54, можно отметить, что после вхождения в установившийся режим блочных пересылок периоды занятости магистрالی следуют друг за другом с меньшими паузами, чем при обычном режиме с поочередной передачей управления.

### **3.19. Распределенный арбитраж магистрали многомашинной системы с переменными приоритетами микроЭВМ [63]**

Многомашинная система (рис. 3.55) содержит микроЭВМ (1—4), объединенные общей магистралью, которая включает шины адреса, данных, управления, а также дополнительно введенную шину S распределенного арбитража.

В установившемся режиме каждая микроЭВМ (1—4) в основном работает автономно и пользуется общей магистралью сравнительно редко. При необходимости обмена данными с абонентом, подключенным к общей магистрали, микроЭВМ с разрешения распределенного арбитра захватывает магистраль и пересылает один или несколько байтов по шине данных в ту или иную сторону в зависимости от типа операции (чтение—запись). Арбитраж магистрали при одновременном формировании запросов со стороны нескольких микроЭВМ основан на сравнении кодов приоритетов конкурирующих микроЭВМ и выборе из них максимального.

Для хранения кода текущего приоритета в каждую микроЭВМ введен счетчик СТ. Содержимое всех счетчиков синхронно увеличивается на единицу по окончании каждого сеанса арбитража. Коды в счетчике изменяются в определенном диапазоне, нижняя и верхняя границы которого задаются содержимым регистров RGL и RGH. В примере, приведенном на рисунке, диапазоны изменения кодов в счетчиках СТ следующие: в микроЭВМ1 он

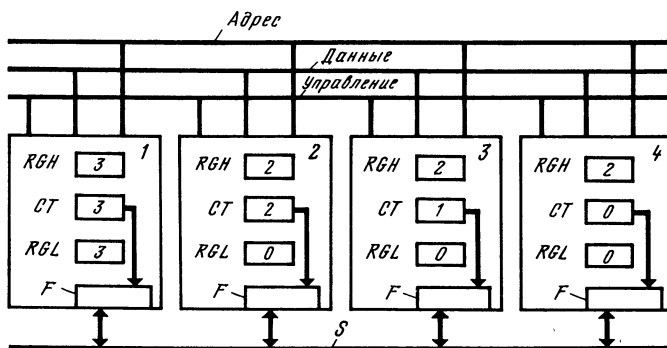


Рис. 3.55. Многомашинная система с переменными приоритетами микроЭВМ (1—4)

состоит из одного кода 3, в каждой из оставшихся—составляет три кода: 0, 1 и 2.

В исходном состоянии в счетчики СТ загружены разные коды (в данном примере 3, 2, 1, 0) для однозначного выбора «победителя» при конфликтах. Чем больше код в счетчике, тем выше приоритет, поэтому микроЭВМ1 поставлена в исключительно благоприятные условия, а остальные «в среднем» равноприоритетны, но всегда проигрывают при взаимодействии с ней. В этом можно убедиться, рассматривая процесс изменения кодов в счетчиках по мере проведения циклов арбитража, показанных в табл. 3.4.

Таблица 3.4

Номер цикла арбитража	Содержимое счетчиков СТ микроЭВМ (1—4)			
	1	2	3	4
1	3	2	1	0
2	3	0	2	1
3	3	1	0	2
4	3	2	1	0
5	3	0	2	1
...	...	...	...	...

Предположим, что в первом цикле арбитража в борьбу за право использования магистрали включились микроЭВМ 2 и 3 (соответствующие приоритеты показаны в первой строке таблицы). Эти микроЭВМ через логические блоки F взаимодействуют между собой по шине S; в результате после окончания переходных процессов в этой шине устанавливается максимальный код 2 из числа поступивших на конкурс (2 и 1). МикроЭВМ2 обнаруживает совпадение выданного и принятого из шины S кодов и по праву считает себя победителем; микроЭВМ3 в данном цикле арбитража проигрывает, так как код 1 в ее счетчике СТ меньше кода 2, установленного в шине S.

МикроЭВМ2 приступает к обмену данными по общей магистрали с выбранным абонентом—с другой микроЭВМ или некоторым общим ресурсом: блоком памяти, контроллером устройства ввода-вывода и т. п. (общие ресурсы на рисунке не показаны). В это время содержимое счетчиков СТ микроЭВМ (2—4) изменяется в соответствии с заданными диапазонами; счетчик микроЭВМ1 остается в состоянии 3, так как верхняя и нижняя

границы диапазона счета совпадают. Система готова к следующему циклу арбитража; состояние счетчиков отражено во второй строке таблицы.

Если во втором цикле арбитража в конкурентную борьбу вступают те же микроЭВМ (2, 3), то на этот раз побеждает микроЭВМ3, так как ее приоритетный уровень (2) теперь более высокий, чем у микроЭВМ2 (0). По окончании цикла арбитража микроЭВМ3 приступает к обмену данными по общей магистрали, а счетчики СТ переходят в состояния, показанные в третьей строке таблицы, и т. д.

Запрос, поступающий от микроЭВМ1, имеет постоянный максимальный приоритетный уровень, равный 3, поэтому обслуживается безусловно. Приоритетные уровни остальных запросов изменяются в зависимости от числа выполненных циклов арбитража (т. е. фактически псевдослучайны в заданных диапазонах), так что в целом эти запросы равноправны. Таким образом, рассмотренный пример показывает, что можно комбинировать жесткие приоритеты с мягкими, обеспечивающими равноправное (демократичное) обслуживание запросов, принадлежащих некоторой группе.

Можно создать полностью жесткую структуру, в которой, например, приоритетный уровень запроса убывает с увеличением номера микроЭВМ. Для этого следует откорректировать начальное содержимое регистров: в регистр RGL микроЭВМ2 записать код 2, в регистры RGH и RGL микроЭВМ3 — код 1, в регистр RGH микроЭВМ4 — код 0, коды в остальных регистрах остаются прежними (см. рисунок). Иными словами, за каждой микроЭВМ закрепляется неизменный код, который предьявляется распределенному арбитражу при возникновении соответствующего запроса.

Для создания полностью равноприоритетной структуры достаточно выделить всем счетчикам (исходно загруженным разными кодами) один и тот же диапазон изменения кодов: от 0 до 3. Можно реализовать и иные структуры, например закрепить за некоторой микроЭВМ минимальный приоритетный уровень, а остальным предоставить большие, но равные возможности и т. п.

Как отмечалось, логические блоки F совместно с шиной S образуют распределенное устройство для вычисления максимального кода из числа предъявленных на конкурс. Это устройство представляет собой комбинационную схему с поразрядным «отсевом» претендентов, чей приоритетный уровень невысок. Схемное решение напоминает комбинационный сумматор с последовательным, параллельным или групповым переносом (в зависимости от требований к бездействию) и впервые предложено в [64]. Описание его работы можно найти в [65].

### **3.20. Устранение конфликтных ситуаций в многомашинной системе [66]**

Обмен информацией между центральными ЭВМ 1—N (рис. 3.56) осуществляется по магистрали через соответствующие блоки связи 1—N. Магистраль включает одну или несколько линий для передачи данных D последовательным или параллельным кодом и линию управления, по которой передается сигнал C занятости магистрали.

Если  $C=0$  (напряжение низкого уровня), то магистраль занята. Если  $C=1$ , то магистраль свободна. При  $C=1$  транзисторы VT1 во всех блоках связи выключены, напряжение высокого уровня на управляющей линии поддерживается благодаря согласующим резисторам R, подключенным между ее физическими концами и шиной питания. Сигнал  $C=0$  формируется в тех случаях, когда хотя бы один транзистор VT1 включен. Одновременное включение нескольких транзисторов VT1 возможно только на этапе конкурсного отбора претендента на использование магистрали, как будет показано далее.

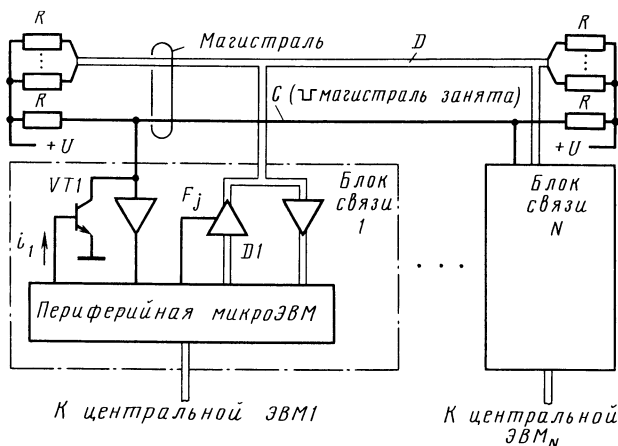


Рис. 3.56. Многомашинальная система.

Арбитраж магистральной осуществляется с использованием сигнала С на управляющей линии

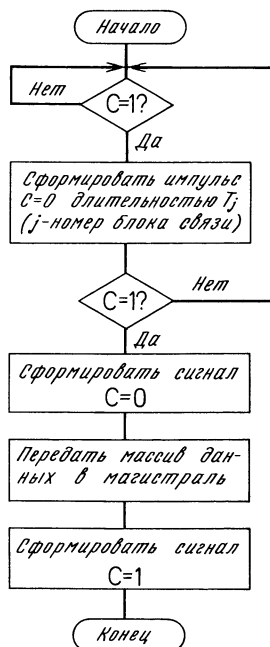


Рис. 3.57. Блок-схема алгоритма выявления победителя в конкурентной борьбе.

Каждый блок оценивает шансы на победу, и только одному из них удастся воспользоваться магистралью

Линии магистральной для передачи данных D так же, как и управляющая линия, выполнены по схеме с открытым коллектором, физические концы линий соединены с шиной питания через согласующие резисторы R.

Блок связи выполнен на основе периферийной микроЭВМ. При наличии тока  $i_j$  ( $j$  — номер блока связи) в базу транзистора VT1 ( $i_j=1$ ) этот транзистор насыщен, при отсутствии тока ( $i_j=0$ ) — выключен. Сигнал С с управляющей линии магистральной через приемник поступает в периферийную микроЭВМ и используется при арбитраже, как будет показано далее.

Сигнал  $F_j=1$  служит для разрешения работы передатчиков с открытым коллектором, через которые информационные сигналы  $D_j$  поступают в магистраль. Приемники транслируют данные из магистральной в периферийную микроЭВМ.

Периферийная микроЭВМ, получив задание от центральной, захватывает магистраль и передает массив данных к одному из блоков связи или, возможно, одновременно к нескольким или ко всем блокам. Периферийная микроЭВМ-абонент распознает на магистральной собственный адрес, принимает массив информации, проверяет контрольную сумму, передает массив к центральной ЭВМ. В протоколе обмена может предусматриваться посылка ответных кодов к микроЭВМ — источнику информации. Эти коды, например, подтверждают правильное завершение сеанса связи или указывают на ошибку и т. п. Вопросы построения протокола обмена далее не рассматриваются, как не существенные для описания идеи построения распределенного арбитра.

Для правильной работы системы необходимо, чтобы в процессе передачи данных активный блок связи был единственным источником информации. Иными словами, недопустимо одновременное использование информационных линий магистральной несколькими блоками, желающими провести сеанс связи. Поэтому в системе предусмотрена процедура арбитража, позволяющая при

наличии нескольких блоков-конкурентов провести их приоритетный отбор и предоставить магистраль в распоряжение только одного из этих блоков.

Если блок связи  $j$  желает воспользоваться магистралью, то он анализирует состояние сигнала  $C$  (рис. 3.57). При  $C=0$  (магистраль занята другим блоком) блок  $j$  переходит в состояние ожидания сигнала  $C=1$ . При  $C=1$  магистраль свободна, блок связи  $j$  (а возможно, и не только этот блок) формирует в управляющей линии импульсный сигнал  $C=0$  «уникальной» длительности  $T_j$ , зависящей от номера блока. После окончания выдачи «своего» импульса блок  $j$  вновь проверяет значение сигнала  $C$ . Импульсы от разных блоков суммируются в общей управляющей линии. Если  $C=0$ , то это означает, что по крайней мере один конкурент с большим приоритетом пытается завладеть магистралью (приоритет возрастает с увеличением длительности импульса). Поэтому блок  $j$  возвращается к ожиданию выполнения условия  $C=1$ .

Если  $C=1$ , то блок  $j$  по праву считает себя победителем в конкурентной борьбе, формирует «статический» сигнал  $C=0$  и приступает к передаче информации (адресной, числовой, управляющей и т. п.) по линиям  $D$ . Возвращаясь к моменту последней проверки сигнала  $C$ , отметим, что с появлением сигнала  $C=1$  блок  $j$  поддерживает сигнал  $C=0$  в течение длительного времени передачи массива данных. После окончания передачи массива блок  $j$  выключает транзистор  $VT1$ , в результате формируется сигнал  $C=1$  и блоки-конкуренты вновь вступают в борьбу за право пользования магистралью.

В качестве примера рассмотрим взаимодействие трех активных блоков с номерами 1—3 (рис. 3.58). В исходном состоянии, до момента  $t_0$ , магистраль занята блоком 3 ( $C=0$ ), который передает данные  $D$  одному из пассивных абонентов. Блоки 1 и 2 желают воспользоваться магистралью и ожидают ее освобождения. В момент  $t_0$  блок 3 по собственной инициативе прекращает передачу данных  $D$  и выключает ранее насыщенный транзистор  $VT1$ , магистраль освобождается ( $C=1$ ). Блоки 1 и 2, обнаружив сигнал  $C=1$ , выждав небольшое время, одновременно или почти одновременно включают транзисторы  $VT1$  токами  $i_1$  и  $i_2$ , оповещая возможных конкурентов о намерении захватить магистраль (момент  $t_1$ ), в результате  $C=0$ .

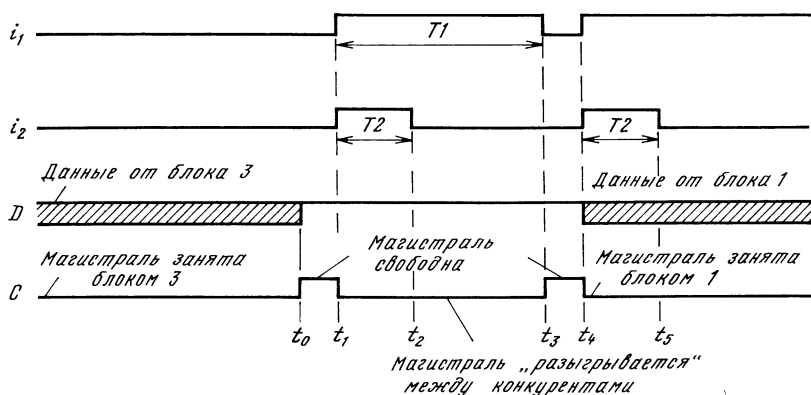


Рис. 3.58. Временная диаграмма взаимодействия трех блоков-конкурентов. Блоки 1 и 2 ожидают освобождения магистрали, а затем вступают в борьбу за право ее использования, в результате побеждает блок 1

В момент  $t_2$ , после окончания импульса длительностью  $T_2$ , блок 2 выключает транзистор VT1 ( $i_2=0$ ) и проверяет состояние сигнала  $C$ . Так как транзистор VT1 блока 1 все еще насыщен ( $T_1 > T_2$ ),  $C=0$ , блок 2 узнаёт о своем поражении в конкурентной борьбе и ожидает появления сигнала  $C=1$ , чтобы включиться в новый тур отбора (см. рис. 3.57).

В момент  $t_3$  блок 1 прекращает подачу тока  $i_1$  в базу транзистора VT1, этот транзистор выключается, формируется сигнал  $C=1$ . Блок 1 принимает к сведению факт формирования сигнала  $C=1$  при выключении тока  $i_1$  и расценивает его как разрешение работы с магистралью. Блок 2, обнаружив, что  $C=1$ , в момент  $t_4$  формирует импульс тока  $i_2$  и в момент  $t_5$  его окончания проверяет состояние сигнала  $C$ . Блок 1 в момент  $t_4$  приступает к передаче данных и сопровождает передачу постоянным включением транзистора VT1 ( $i_1=1$ ). Поэтому в момент  $t_5$  блок 2 обнаруживает, что  $C=0$ , т.е. очередная попытка завладеть магистралью оказалась неудачной.

В отличие от традиционных систем связи (например, типа Ethernet), в предлагаемой системе исключены наложения данных при конфликтах, что упрощает схемные решения и алгоритмы взаимодействия, исключает необходимость повторной передачи данных по магистрали.

### 3.21. Структура блоков кэш-памяти в многопроцессорной системе с общим ОЗУ [67]

В многопроцессорной системе (рис. 3.59) каждый процессор ЦП1—ЦП<sub>N</sub> подключен к магистрали через быстродействующий блок статической кэш-памяти с временем доступа к информации порядка 50 нс. Объем кэш-памяти обычно небольшой, например 2К ячеек. В этой памяти по мере работы процессора автоматически (без использования каких-либо дополнительных программных средств) накапливается наиболее актуальная информация, так что эта информация оказывается «приближенной» к процессору и ему нет необходимости обращаться за ней через магистраль в общее ОЗУ. Время доступа к информации в общем ОЗУ примерно 300 нс, его объем может быть равен, например, 4М ячеек.

Блоки кэш-памяти уменьшают интенсивность обращений к общему ОЗУ по магистрали со стороны разных процессоров, так как эти блоки создают своим процессорам благоприятные условия для более или менее автономной работы. Кроме того, процессоры работают с большей производительностью в силу отмеченного ранее различия быстродействия кэш-памяти и общего ОЗУ. (Выигрыш достигается лишь при чтении ячеек, копии которых имеются в кэш-памяти.) Структура кэш-памяти описана, например, в [65, с. 446], поэтому далее рассмотрим лишь те ее особенности, которые связаны с взаимодействием процессоров. Отметим только, что содержимое ячейки

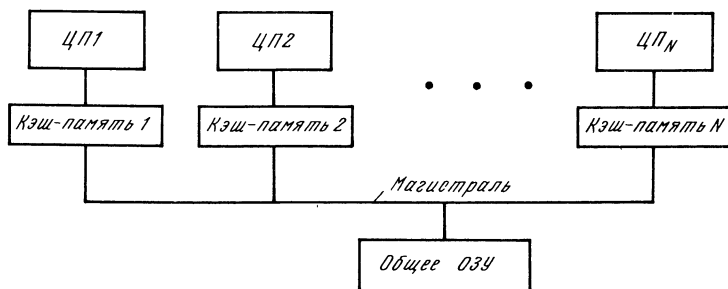


Рис. 3.59. Многопроцессорная система.

Проблема состоит в обеспечении соответствия между копиями и оригиналами ячеек, выполняющих функции «почтовых ящиков»

общего ОЗУ копируется в кэш-памяти при первоначальном считывании этой ячейки соответствующим процессором, а далее процессор использует эту быстродоступную копию. При обновлении копии процессор одновременно обновляет и оригинал, лежащий в общем ОЗУ.

Взаимодействие процессоров осуществляется только через общее ОЗУ, в котором ряд ячеек выполняет функции «почтовых ящиков». Например, ЦП<sub>1</sub> может быть назначен центральным, а ЦП<sub>2</sub>—ЦП<sub>N</sub>—периферийными (подчиненными). Процессор ЦП<sub>1</sub> по мере необходимости формирует в общем ОЗУ задания для процессоров ЦП<sub>2</sub>—ЦП<sub>N</sub>. После формирования задания для ЦП<sub>i</sub> процессор ЦП<sub>1</sub> устанавливает в определенной ячейке общего ОЗУ признак готовности задания. Процессор ЦП<sub>i</sub> (исходно не занятый выполнением какого-либо задания) периодически проверяет содержимое указанной ячейки общего ОЗУ («почтового ящика») и, обнаружив в ней определенный код, приступает к считыванию, анализу и выполнению задания, сформулированного процессором ЦП<sub>1</sub>.

После выполнения задания ЦП<sub>i</sub> помещает в общее ОЗУ результаты работы и записывает в «почтовый ящик» (тот же самый, что и упомянутый ранее, или иной, по договоренности разработчиков системы) ответный код—признак окончания работы. Процессор ЦП<sub>1</sub>, обнаружив этот признак при очередной проверке содержимого «почтового ящика», принимает к сведению факт выполнения задания. Далее процессы протекают аналогично. (Поведение системы может быть более сложным, однако для понимания предложенной идеи достаточно ограничиться описанным примером.)

Проблема состоит в обеспечении идентичности информации в «почтовом ящике» и соответствующей ячейке одного или нескольких блоков кэш-памяти, где сформирована копия. Иными словами, кэш-память, содержащая копию ячейки—«почтового ящика» общего ОЗУ, должна своевременно узнавать об изменениях содержимого ячейки—оригинала и при обнаружении такого изменения ликвидировать или обновлять устаревшую копию. В противном случае соответствующий процессор будет циклически запрашивать из своей кэш-памяти копию ячейки—«почтового ящика», которая устарела и уже не соответствует изменившейся ситуации. Так, в данном примере процессор ЦП<sub>1</sub>, работая со своей кэш-памятью, никогда не дожидется получения извещения о готовности задания, сформированного ЦП<sub>1</sub>, если кэш-память ЦП<sub>i</sub> не отреагирует на передачу этого извещения в общее ОЗУ уничтожением или коррекцией старой рабочей копии ячейки «почтового ящика». (Старая копия свидетельствует об отсутствии готовности задания, хотя готовность появилась.)

Чтобы кэш-память реагировала на смену содержимого определенных ячеек общего ОЗУ, можно с помощью дешифратора, подключенного к магистралам, выявлять соответствующие адреса и при обнаружении таковых уничтожать в кэш-памяти соответствующие копии, если они были ранее созданы (а лучше—не уничтожать, а надлежащим образом корректировать). Однако такое решение оказывается слишком сложным и архитектурно жестким, так как изменение адресов ячеек общего ОЗУ, выбранных в качестве «почтовых ящиков», влечет за собой необходимость изменения схем дешифраторов, реагирующих на эти адреса.

Иногда применяют иное решение. Оно состоит в том, что определенные ячейки общего ОЗУ (соответствующие выбранным адресам «почтовых ящиков»), объявляются не копируемыми. Для осуществления этой идеи в общее ОЗУ вводится дополнительный разряд, указывающий, можно или нельзя создавать копию данной ячейки в кэш-памяти. Тогда центральный и периферийный процессоры вынуждены пользоваться одним и тем же оригиналом ячейки—«почтового ящика», поэтому проблем не возникает. Такое решение помимо подключения дополнительного разряда общего ОЗУ требует введения схем, обеспечивающих программную доступность этого разряда. Кроме того (и это, может быть, главное), нужны программы, обеспечивающие поддержание правильной информации в дополнительном разряде общего ОЗУ.

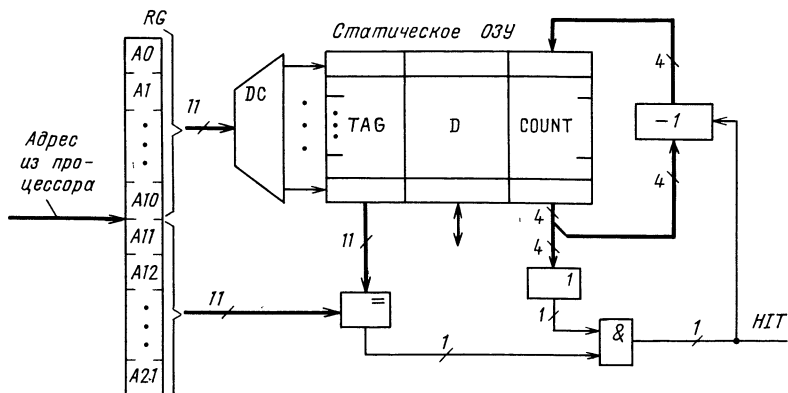


Рис. 3.60. Кэш-память с автоматическим обновлением копии после уменьшения кода COUNT до нуля.

При каждом обращении к копии код COUNT, первоначально равный 1111, уменьшается на единицу

В предлагаемом решении (рис. 3.60) не требуются дешифраторы, не нужен дополнительный разряд общего ОЗУ, имеющееся программное обеспечение не меняется. (О недостатках — позже.) Схема работает следующим образом. Адрес, поступающий из процессора при считывании ячейки общего ОЗУ или своей кэш-памяти (если повезет), запоминается в регистре RG. Младшие 11 разрядов адреса (A0—A10) поступают на входы дешифратора DC, который выбирает одну из  $2^{11}$  ячеек статического ОЗУ — собственно кэш-памяти. Выбранная ячейка содержит данные D, некоторый код TAG («ярлычок», «этикетка» — англ.) и код COUNT («счет»), установленный в состояние 1111<sub>2</sub> при создании копии. Содержимое ячейки (D, TAG, COUNT) считывается из статического ОЗУ. Старшие 11 разрядов адреса (A11—A21) из регистра RG сравниваются с разрядами кода TAG с помощью компаратора.

Если компаратор не зарегистрировал совпадения, то это означает, что копия ячейки с адресом, хранимым в регистре RG, отсутствует в кэш-памяти. В этом случае сигнал HIT=0 побуждает схему управления захватить магистраль, извлечь нужное слово данных из общего ОЗУ, переслать его в процессор и попутно сохранить в выбранной ячейке кэш-памяти. Код TAG при этом устанавливается равным коду в разрядах A11—A21 регистра RG, а код COUNT переводится в состояние 1111. Копия создана.

Если компаратор зарегистрировал совпадение, то это означает, что копия ячейки с адресом, хранимым в регистре RG, уже имеется в кэш-памяти. Если считанный код COUNT  $\neq 0$ , то формируется сигнал HIT=1, означающий, что нужное слово данных D считано из статического ОЗУ и готово для передачи в процессор. Таким образом, процессор получает интересное его слово данных из кэш-памяти, а не из общего ОЗУ.

При выдаче сигнала HIT=1 считанный из статического ОЗУ код COUNT арифметически уменьшается на единицу с помощью схемы вычитания («-1») и вновь записывается в выбранную ячейку. При следующем (когда-либо в дальнейшем) считывании этой ячейки и формировании сигнала HIT=1 код COUNT вновь уменьшается на единицу и т. д. В итоге наступает момент, когда считанный код COUNT=0, при этом срабатывает элемент ИЛИ (на его выходе сигнал лог. 0), запирая элемент И и препятствуя формированию сигнала HIT=1. Схема управления, обнаружив сигнал HIT=0, работает по описанному ранее алгоритму создания новой копии в кэш-памяти.

Иными словами, не разрешается пользоваться одной и той же копией более 15 раз во избежание потери ее соответствия оригиналу. При

шестнадцатом обращении копия не выдается, а затем уничтожается и заменяется более «свежей», совпадающей с только что считанным из общего ОЗУ оригиналом. Таким образом, предлагаемое решение снимает описанную ранее проблему рассогласования информации в кэш-памяти и общем ОЗУ.

Недостатком этого решения является относительно большая задержка копирования, так как в худшем случае устаревшая копия будет 15 раз использоваться процессором, прежде чем заменится новой. Уменьшение разрядности кода COUNT способствовало бы уменьшению задержки, однако следует учесть, что при этом снизилась бы эффективность использования кэш-памяти в целом. Действительно, смысл применения кэш-памяти как раз и состоит в том, чтобы однократно извлекать данные из «медленного» ОЗУ и многократно пользоваться ими, обращаясь к «быстрой» кэш-памяти; уменьшение же разрядности кода COUNT препятствует реализации этого процесса, т. е. сводит на нет все преимущества. Поэтому при выборе разрядности кода COUNT следует принимать компромиссное решение.

### 3.22. Повышение производительности микропроцессорной системы с кэш-памятью [68]

Кэш-память (рис. 3.61) представляет собой быстродействующее статическое ОЗУ небольшого объема, в котором по мере работы процессора автоматически (т. е. без помощи каких-либо программных средств) накапливается наиболее актуальная информация. Первоначально эта информация находится в главной памяти большого объема, выполненной на сравнительно «медленных» микросхемах динамических ОЗУ, а затем при считывании информации из главной памяти соответствующие копии откладываются в кэш-памяти, причем, поскольку кэш-память имеет малый объем (например, 8К ячеек), а поток считываемых кодов «бесконечен», старые копии вытесняются новыми, а новые — новейшими, которые, в свою очередь, со временем устаревают и также вытесняются. За время жизни «быстрой» копии процессор может неоднократно пользоваться ею, не обращаясь к «медленному» оригиналу; в этом, собственно, и заключается смысл использования кэш-памяти.

Кэш-память содержит группу накопителей информации (на основе статических ОЗУ) с общей схемой дешифрации адреса. Полный адрес, поступающий из процессора, содержит группу старших и группу младших разрядов (процессор «не знает» о подобной трактовке выдаваемого им адресного кода,

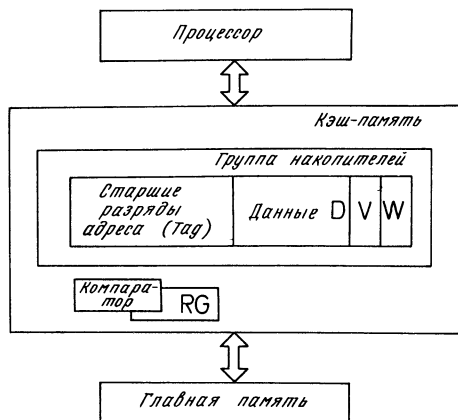


Рис. 3.61. Система с кэш-памятью. В отличие от традиционной схемы, в кэш-память введен буферный регистр RG для хранения вытесняемого из ячейки кэш-памяти старого слова, которое в дальнейшем пересылается в главную память, если это необходимо

равно как и об описанных далее «маленьких хитростях»). Группа младших разрядов через адресный дешифратор выбирает строку, состоящую из ячеек памяти всех накопителей, в результате считываются слово данных D, сопровождающий это слово код Tag и признаки V и W.

Код Tag содержит значения старших разрядов адреса ячейки — оригинала. Этот код после считывания из накопителя кэш-памяти поступает на компаратор и сравнивается с группой старших разрядов адреса, поступившего из процессора. Если зарегистрировано совпадение, то это означает, что копия содержимого ячейки памяти, к которой обратился процессор, присутствует в кэш-памяти. Если совпадения нет, то в выбранной ячейке кэш-памяти хранится «чужой» код, не имеющий отношения к текущему обмену информацией.

Разряд W несет информацию о том, соответствует ли копия оригиналу. При  $W=0$  слово данных D совпадает с исходным словом, хранимым в главной памяти, так как процессор, если и обращался к копии, то в режиме чтения. Условие  $W=1$  означает, что была, по крайней мере, одна запись в данную ячейку кэш-памяти по инициативе процессора, а оригинал из главной памяти не востребовался (чтобы не снижать темп работы). Поэтому при  $W=1$  схема управления кэш-памяти принимает к сведению, что копия не соответствует устаревшему оригиналу. (В частном случае запись может не изменить, а всего лишь подтвердить имеющийся в ячейке кэш-памяти код, тогда, несмотря на то, что  $W=1$ , копия будет оставаться совпадающей с оригиналом. Однако для упрощения схемных решений условие  $W=1$  рассматривается как показатель безусловного рассогласования копии с оригиналом.)

Признак V отражает истинность информации в данной строке. При  $V=0$  строка «выключена», при  $V=1$  строка включается в работу. Первоначально все признаки V установлены в 0, а затем по мере заполнения кэш-памяти копиями кодов, хранящихся в главной памяти, соответствующие признаки V переводятся в единичные состояния.

Забегая вперед, отметим, что смысл предлагаемого решения заключается в уменьшении потерь времени, связанных с коррекцией устаревшего оригинала. Для этого в кэш-память введен буферный регистр RG и изменен алгоритм работы схемы управления. Однако, чтобы понять предложенную идею, нам придется рассмотреть работу обычной системы с кэш-памятью, в которой регистр RG отсутствует. Возможные режимы работы системы отражены в табл. 3.5. Приведем описание этих режимов в соответствии с принятой в таблице нумерацией.

1. В этом режиме процессор выполняет чтение информации из некоторой ячейки памяти. В соответствии со значениями младших разрядов адреса, установленного процессором, из группы накопителей кэш-памяти считывается слово данных D, код Tag и признаки  $V=1$  и  $W=1$ . В данном случае компаратор не зарегистрировал совпадения старших разрядов адреса, установленного процессором, с разрядами кода Tag (копии нет). Процессор временно приостанавливается сигналом отсутствия готовности данных.

Прежде чем создать новую копию, необходимо освободить выбранную ячейку кэш-памяти от старой. Простое уничтожение старой копии привело бы к потере информации, так как эта копия обновлялась и уже не соответствует оригиналу ( $W=1$ ). Поэтому старое содержимое выбранной ячейки кэш-памяти пересылается в соответствующую ячейку главной памяти, где хранится старый (не откорректированный) оригинал. Адрес ячейки-оригинала формируется следующим образом. Старшие его разряды содержат код Tag, считанный из ячейки кэш-памяти, а младшие разряды совпадают с адресом выбранной ячейки кэш-памяти.

После коррекции старого оригинала из главной памяти считывается слово данных в соответствии с адресом, поступившим из процессора. Это слово записывается в освободившуюся после вытеснения старой информации ячейку кэш-памяти с соответствующим новой копии кодом Tag и признаками

Таблица 3.5

Номер режи- ма	Режим обра- щения про- цессора к памяти	Исходные условия		Последующая реакция системы	Ситуация в ячейках главной памяти
		Имеется ли в кэш-памяти со- держащегося кэш-памяти, к которой обращается процессор?	Если информация в ячейке кэш- памяти подлежит уничтожению, то была ли она ранее изменена процессором?		
1	Чтение	Нет	Да ( $W = 1$ )	Старое содержимое пересылается в старую ячейку главной памяти, затем принимается информация из считываемой процессором новой ячейки главной памяти	Старая ячейка принимает информацию из кэш-памяти, информация в новой ячейке остается неизменной
2	»	»	Нет ( $W = 0$ )	Содержимое ячейки кэш-памяти обновляется, т. е. создается копия содержимого ячейки главной памяти при пересылке информации из главной памяти в процессор; старая копия теряется	Информация в ячейках главной памяти не изменяется
3	»	Да	Информация в ячейке кэш-памяти не подлежит уничтожению	Информация из ячейки кэш-памяти передается в процессор	То же
4	Запись	Нет	То же	Информация в ячейке кэш-памяти не изменяется	Информация в старой ячейке не изменяется, новая ячейка принимает информацию из процессора
5	»	Да	Информация в ячейке кэш-памяти подлежит уничтожению, однако реакция системы не зависит от того, была ли эта информация ранее изменена процессором	Содержимое ячейки кэш-памяти обновляется, т. е. в эту ячейку принимается информация из процессора; признак $W$ устанавливается равным 1	Информация в ячейках главной памяти не изменяется (оригинал устарел и не соответствует новой копии)

$W=0$  и  $V=1$ . Слово данных передается также в процессор и подтверждается сигналом готовности данных, получив который процессор принимает слово и продолжает работу.

2. Этот режим отличается от предыдущего тем, что старая копия совпадает с оригиналом ( $W=0$ ), поэтому ее можно уничтожить при создании новой копии.

3. В данном режиме компаратор регистрирует совпадение кодов, информация с минимальной задержкой выдается из кэш-памяти в процессор, его приостановки не требуется. Признак  $W$  остается неизменным.

4. Процессор выполняет запись информации в ячейку главной памяти. Кэш-память не участвует в работе, так как она не содержит копию адресуемой ячейки. Копия не создается.

5. При записи кода в ячейку, копия которой содержится в кэш-памяти, сама эта ячейка не затрагивается. Корректируется только копия. Первая же коррекция вызывает установку в 1 признака  $W$ . В дальнейшем при вытеснении этой копии (см. п. 1) оригинал обновляется.

Поведение системы в режимах 1—3 отражено на блок-схеме алгоритма чтения (рис. 3.62). В соответствии с поступившим из процессора адресом (блок A1) проверяется присутствие в кэш-памяти копии запрашиваемого

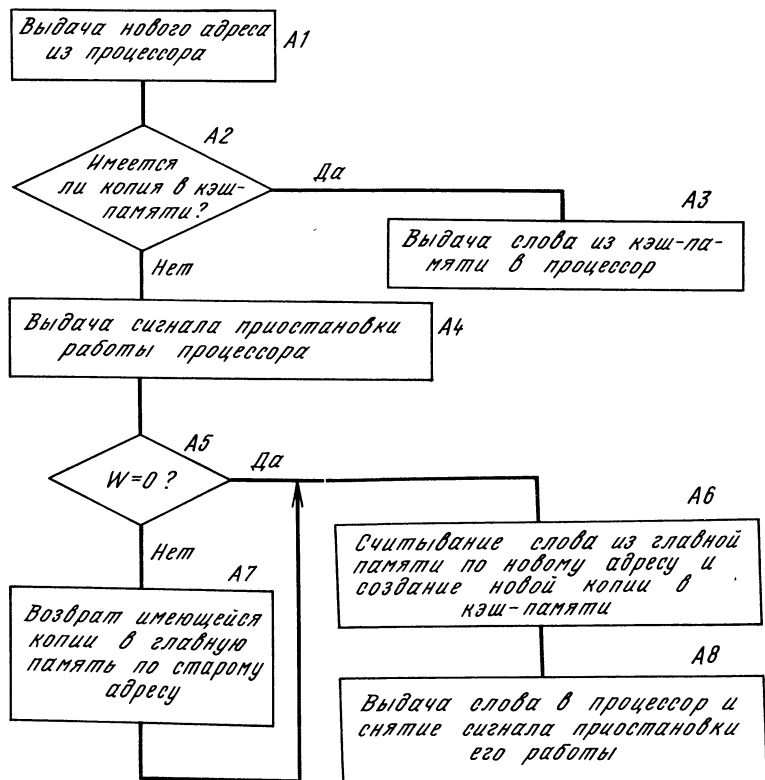


Рис. 3.62. Блок-схема алгоритма чтения в традиционной системе с кэш-памятью (см. рис. 3.61 без учета буферного регистра RG).

Возможны относительно длительные приостановки работы процессора, когда в главную память сначала возвращается старая копия, а затем извлекается новая

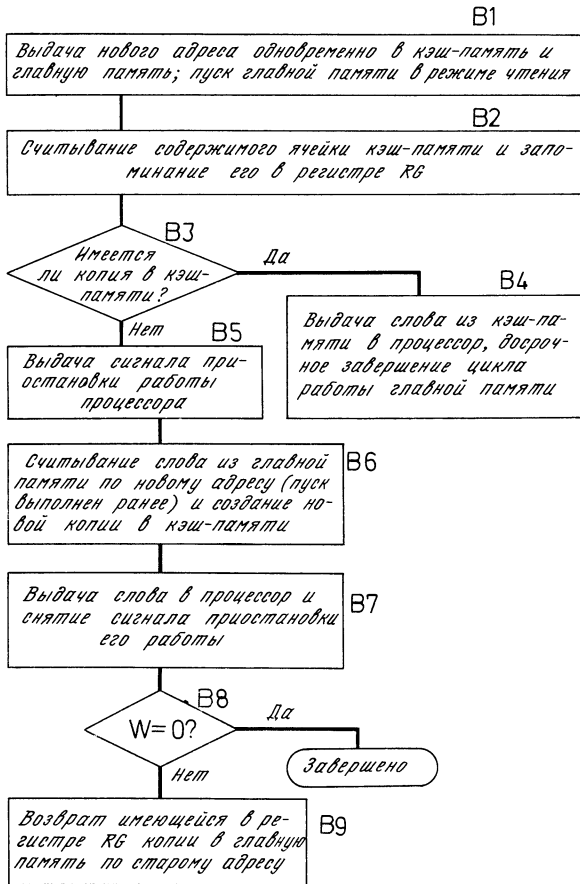


Рис. 3.63. Блок-схема алгоритма чтения в усовершенствованной системе с кэш-памятью (см. рис. 3.61 с учетом буферного регистра RG).

Приостановки работы процессора связаны только со считыванием новых копий из главной памяти; возврат старых копий совмещен во времени с работой процессора

слова (блок A2), и при наличии копии слово D передается в процессор (блок A3), цикл работы системы завершается.

Если копии нет, то процессор приостанавливает работу, т. е. переходит к выполнению циклов ожидания (блок A4). Если вытесняемая копия не была ранее изменена процессором (блок A5, условие  $W=0$  выполнено), то осуществляется пуск главной памяти по новому адресу, в результате на место старой копии помещается новая (блок A6). Считанное слово передается в процессор, которому разрешается дальнейшая работа (блок A8). Если вытесняемую копию необходимо сохранить в главной памяти (блоки A5, A7), то период ожидания данных процессором увеличивается на время записи этой копии в главную память по старому адресу.

Рассмотрим наконец работу усовершенствованной системы, пользуясь блок-схемой алгоритма чтения, приведенной на рис. 3.63.

Адрес, выданный процессором, поступает одновременно в кэш-память и главную память (блок В1). В главной памяти «на всякий случай» начинается цикл считывания слова данных по указанному процессором адресу — на микросхемы динамических ОЗУ подается группа адресных сигналов и управляющий сигнал RAS выбора строки.

В соответствии с младшими разрядами адреса, поступившего из процессора, из кэш-памяти извлекаются коды Tag, D и признаки V и W. Вся эта информация запоминается (дублируется) в регистре RG (блок В2). Благодаря этому выбранная ячейка кэш-памяти в дальнейшем может использоваться для приема новой копии без предварительного возврата в главную память старой копии, когда  $W=1$ .

Далее (блок В3) проверяется наличие копии нужного слова в кэш-памяти. Если копия имеется (компаратор сформировал сигнал совпадения кодов), то слово D передается в процессор (блок В4). Ранее начатый цикл считывания слова из динамического ОЗУ (главной памяти) завершается досрочно (сигнал выбора строки RAS, если его длительность достигла или превысила минимально допустимое значение, снимается, сигнал выбора столбца CAS не формируется), так как слово-оригинал уже заменено его точной копией, переданной в процессор. На этом цикл работы системы завершается.

Если копии нет, то работа процессора приостанавливается (блоки В3, В5), из главной памяти извлекается требуемое слово, которое с новыми разрядами кода Tag помещается в выбранную ячейку кэш-памяти и одновременно пересылается в процессор, которому разрешается дальнейшая работа (блоки В6, В7).

Теперь решается вопрос о необходимости сохранения старой копии (она находится в регистре RG) в соответствующей ячейке главной памяти. Если  $W=0$ , то копию сохранять не нужно, цикл работы системы завершается. Если  $W=1$ , то, прежде чем завершить работу, осуществляется коррекция оригинала (блок В9). Однако в отличие от традиционного алгоритма коррекция происходит на фоне работы процессора, который обрабатывает только что полученное слово и еще не сформировал новый запрос передачи данных из памяти или в память. Таким образом, время возврата старой копии в главную память исключается из цикла ожидания процессором слова данных, а пуск динамического ОЗУ осуществляется с упреждением (см. блок В1).

### 3.23. Схема быстрого выделения ресурсов по запросам текущих заданий [69]

В мультипрограммных и (или) мультипроцессорных системах используются общие для всех выполняемых заданий ресурсы: устройства ввода — вывода, блоки памяти, регистры и т. п. До начала или возобновления выполнения очередного задания система должна быть уверена в том, что все необходимые ресурсы имеются, т. е. они не заняты конкурирующими выполняемыми заданиями.

Поиск необходимых свободных ресурсов может осуществляться программно или с помощью специализированной аппаратуры, что предпочтительно при построении высокопроизводительных систем. Далее рассмотрена схема (рис. 3.64), которая следит за наличием свободных ресурсов и по запросу стоящего на очереди задания дает отрицательный или положительный ответ.

При отрицательном ответе вырабатывается сигнал  $HOLD=1$ , который означает, что свободных ресурсов нет или их недостаточно для начала (возобновления) выполнения задания. Поэтому задание откладывается. Положительный ответ сопровождается сигналом  $HOLD=0$  и означает, что достаточные ресурсы имеются. При этом на выходах М устанавливается информация, по которой можно определить, какие именно свободные ресурсы выделяются для выполнения задания. В дальнейшем выделенные ресурсы

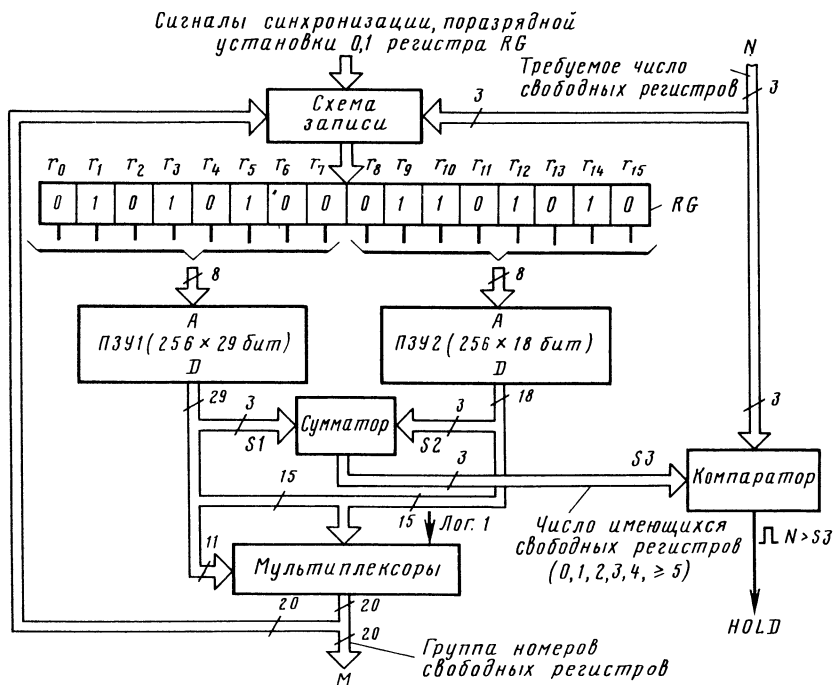


Рис. 3.64. Схема быстрого выделения ресурсов (регистров) по запросам текущих заданий

считаются занятыми до тех пор, пока задание не завершится или пока система принудительно не приостановит его выполнение в связи с более важными работами либо в соответствии с принятым расписанием.

Предположим для определенности, что в качестве общего ресурса используется группа из 16 регистров  $R_0$ — $R_{15}$  (на рис. 3.64 эти регистры не показаны). Каждое поступающее задание требует  $N$  регистров ( $1 \leq N \leq 5$ ), не обязательно соседних в группе. Текущее состояние ресурса отображается служебным регистром  $RG$ , в котором каждый разряд соответствует определенному регистру ресурса. Так, разряд  $r_0$  отображает состояние регистра  $R_0$  ресурса, разряд  $r_1$  — состояние регистра  $R_1$  и т. д. При  $r_i = 0$  регистр  $R_i$  занят, при  $r_i = 1$  — свободен для использования новым заданием. В исходном состоянии все регистры  $R_0$ — $R_{15}$  свободны, затем при работе системы ресурс используется текущими заданиями в той мере, насколько это необходимо. В примере, приведенном на рис. 3.64, текущая ситуация такова, что регистры  $R_0$ ,  $R_2$ ,  $R_4$ ,  $R_6$ ,  $R_7$ ,  $R_8$ ,  $R_{11}$ ,  $R_{13}$  и  $R_{15}$  заняты, остальные регистры свободны.

Предположим, что на входы схемы поступил запрос на выделение  $N=5$  свободных регистров. Как было показано, в данном примере имеется 9 занятых и 7 не занятых регистров, т. е. запрос должен быть удовлетворен. Выделение пяти свободных регистров осуществляется следующим образом.

Разряды  $r_0$ — $r_7$  и  $r_8$ — $r_{15}$  регистра  $RG$  поступают на адресные входы ПЗУ1 и ПЗУ2, из этих ПЗУ считываются 29- и 18-разрядный коды. В 3-разрядном коде  $S_1$  содержится информация о наличии свободных регистров в группе  $R_0$ — $R_7$ . Если  $S_1 = 0$ , то свободных регистров нет, при  $S_1 = 1, 2, 3, 4$  имеется соответственно 1, 2, 3 и 4 свободных регистра, при  $S_1 = 5$  имеется от пяти до восьми свободных регистров. Аналогично код  $S_2$

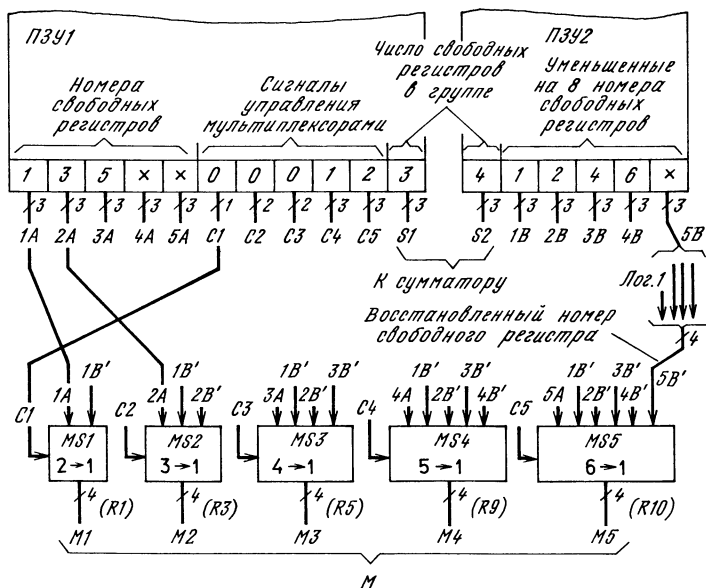


Рис. 3.65. Фрагмент схемы быстрого выделения ресурсов (рис. 3.64)

отображает ситуацию в группе регистров R8—R15. С помощью сумматора определяется общее число S3 свободных регистров в группе R0—R15. Сумматор выполнен так, что S3=0, 1, 2, 3, 4 при соответствующем общем числе свободных регистров и S3=5, если это число равно или превышает 5.

Код S3 сравнивается компаратором с кодом N. Если  $N > S3$ , то формируется сигнал HOLD=1. В данном примере  $N=5$ ,  $S3=5$  (имеется 7 свободных регистров), следовательно, HOLD=0. Оставшиеся разряды кодов, считанных из ПЗУ1 и ПЗУ2, поступают на управляющие и информационные входы мультиплексоров. С выходов мультиплексоров считывается упорядоченная в порядке возрастания группа M номеров свободных регистров. В данном примере это группа 1, 3, 5, 9, 10.

В соответствии с кодами N и M схема записи устанавливает в 0 нужные разряды регистра RG, отражая факт предоставления части общего ресурса в ответ на полученный запрос. В дальнейшем, по мере освобождения регистров R0—R15 соответствующие разряды регистра RG переводятся в состояние 1 с помощью внешнего источника сигналов (на рисунке не показан).

Процесс формирования кода M поясняется рис. 3.65. Как отмечалось, из ПЗУ1 считывается 29-разрядный код. Этот код содержит до 5 3-разрядных номеров свободных регистров в группе R0—R7, которые поступают на выходы 1A, 2A, 3A, 4A, 5A. В данном примере в группе R0—R7 три свободных регистра: R1, R3 и R5, что отражено числами 1, 3, 5, выдаваемыми на выходы 1A, 2A и 3A ПЗУ1. Других свободных регистров в группе нет, поэтому на выходы 4A и 5A выдаются произвольные (например, нулевые) коды, помеченные на рисунке знаками ×.

Аналогично из ПЗУ2 считываются номера свободных регистров в группе R8—R15 (до 5 номеров), причем для уменьшения разрядности выходов 1B, 2B, 3B, 4B и 5B счет номеров ведется с нуля, а не с восьми. Цифра 8 подразумевается в качестве слагаемого и учитывается добавлением сигнала лог. 1 в старший разряд передаваемого к мультиплексору кода (см. показанное на рис. 3.65 преобразование 5B→5B', остальные выходы 1B—4B преобразуются

Таблица 3.6

MS1		MS2		MS3		MS4		MS5	
C1	M1	C2	M2	C3	M3	C4	M4	C5	M5
0	1A	00	2A	00	3A	000	4A	000	5A
1	1B'	01	1B'	01	1B'	001	1B'	001	1B'
—	—	10	2B'	10	2B'	010	2B'	010	2B'
—	—	—	—	11	3B'	011	3B'	011	3B'
—	—	—	—	—	—	100	4B'	100	4B'
—	—	—	—	—	—	—	—	101	5B'

аналогично). В данном примере в группе R8—R15 свободны регистры R9, R10, R12 и R14, что соответствует цифрам 1, 2, 4 и 6 (номер регистра, уменьшенный на 8), показанным на выходах ПЗУ2. На входы 1B'—4B' подаются восстановленные цифры 9, 10, 12 и 14.

Сигналы с выходов C1—C5 ПЗУ1 подаются на управляющие входы мультиплексоров MS1—MS5. Работа мультиплексоров поясняется табл. 3.6. В таблице показано, какие входные сигналы передаются на выходы при разных управляющих кодах. Например, мультиплексор MS4 управляется 3-разрядным кодом с выходов C4. Если на управляющих входах код 000, то на выходы M4 пересылается код с линий 4A (старший разряд доопределяется нулем), коду 001 соответствует передача на выходы M4 кода с линий 1B' и т. д.

В данном примере с выходов C1, C2 и C3 ПЗУ считаны 0, с выходов C4 и C5 — коды 1 и 2. Это означает, что M1=1A=1, M2=2A=3, M3=3A=5, M4=1B'=9, M5=2B'=10, что и требовалось.

Если бы все 5 свободных регистров находились в группе R0—R7, то управление мультиплексорами MS1—MS5 задавалось так: C1...C5=00000. При наличии меньшего числа свободных регистров (4, 3, 2, 1, 0) управление задается соответственно так: C1...C5=00001, 00012 (рассмотренный ранее пример), 00123, 01234, 12345.

### 3.24. Схема учета и поиска свободных устройств [70]

Для распараллеливания обработки информации в вычислительных системах используют группы однотипных устройств: процессоров, контроллеров, блоков памяти и т. п. При работе системы каждое из этих устройств включается в действие, получив соответствующее задание, затем освобождается, получает новое задание и т. д. Координация процессов назначения исполнительных устройств, выдачи им заданий и приема результатов осуществляется центральным процессором вычислительной системы.

Чтобы следить за состоянием системы, ЦП должен вести оперативный учет имеющихся в данный момент свободных устройств. Использование чисто программного способа учета и поиска свободных устройств неэффективно из-за больших потерь времени. Поэтому применяют различные схемные решения, уменьшающие эти потери. Рассмотрим одно из таких решений.

Предположим, что имеется группа из 64 однотипных устройств с номерами 0—63<sub>10</sub>. Каждое устройство может находиться в двух состояниях — свободном и занятом. Для отображения состояний устройств воспользуемся графом, приведенным на рис. 3.66. Граф содержит узлы, размещенные на четырех уровнях (0—3), и ребра, соединяющие узлы соседних уровней. Он имеет древовидную структуру с общим корнем и разветвляющимся множеством ребер. Нумерация узлов, показанная на рисунке, тесно связана со схемным решением, рассмотренным позже.

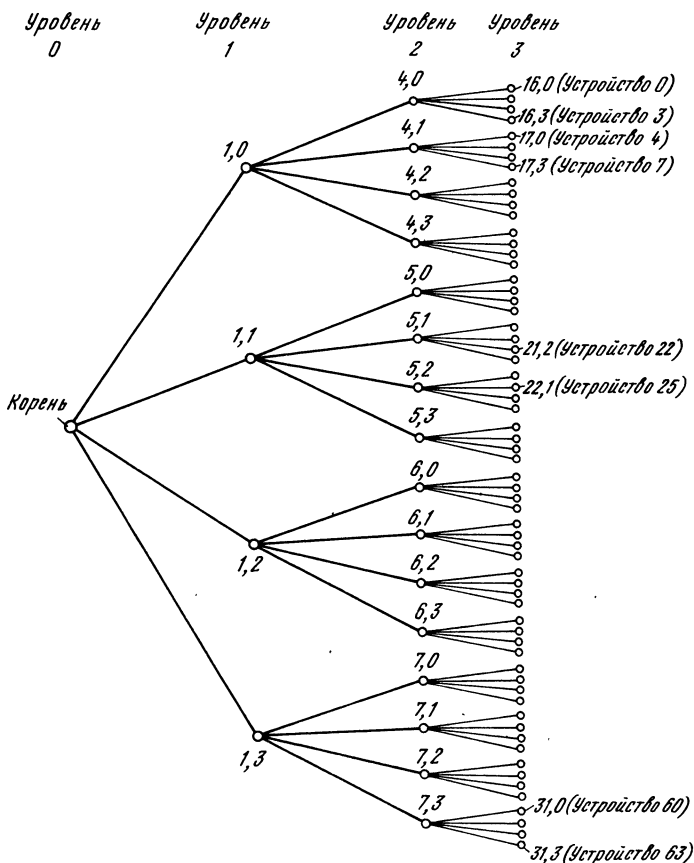


Рис. 3.66. Граф состояний группы из 64 однотипных устройств

Каждый узел уровня 3 отражает состояние одного из устройств в группе. Так, узел 16,0 отражает состояние устройства с номером 0, узел 16,1 — состояние устройства с номером 1 и т. д., узлы 31,2 и 31,3 соответствуют устройствам с номерами 62 и 63. Если устройство свободно (готово к работе), то соответствующий узел находится в состоянии лог. 1. Если устройство занято выполнением ранее полученного задания или не готово к работе по иным причинам, то узел находится в состоянии 0.

Каждый узел уровня 2 отражает состояние соответствующей четверки узлов уровня 3. Если хотя бы один из узлов четверки находится в состоянии 1 (устройство свободно), то узел уровня 2 также находится в состоянии 1. С другой стороны, нулевое состояние узла уровня 2 означает, что все четыре узла уровня 3 находятся в состоянии 0 (устройство занято). Аналогичные условия распространяются также на узлы низших уровней. Таким образом, если узел уровня 0 находится в состоянии 0, то свободных устройств в системе нет. Если узел уровня 0 находится в состоянии 1, то это означает, что имеется по крайней мере одно свободное устройство, которому ЦП может поручить работу.



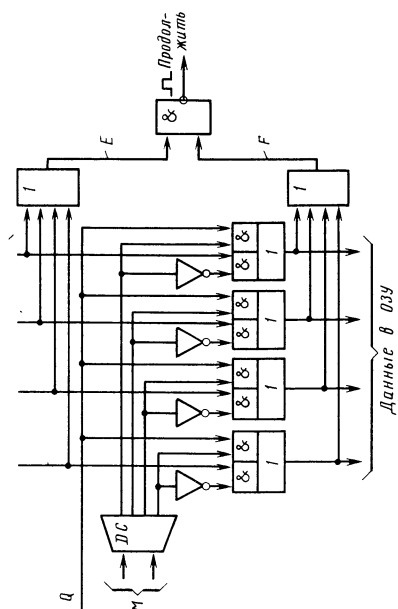


Рис. 3.68. Структура преобразователя данных (рис. 3.67)

Адрес: 0	Бит 0	Бит 1	Бит 2	Бит 3
1	Незанятая ячейка			
2	1,0	1,1	1,2	1,3
3	Незанятые ячейки			
4	4,0	4,1	4,2	4,3
5	5,0	5,1	5,2	5,3
6	6,0	6,1	6,2	6,3
7	7,0	7,1	7,2	7,3
8	Незанятые ячейки			
9				
14	Незанятые ячейки			
15				
16	16,0	16,1	16,2	16,3
17	17,0	17,1	17,2	17,3
18	18,0	18,1	18,2	18,3
19	19,0	19,1	19,2	19,3
20	20,0	20,1	20,2	20,3
21	21,0	21,1	21,2	21,3
22	22,0	22,1	22,2	22,3
23	23,0	23,1	23,2	23,3
30	30,0	30,1	30,2	30,3
31	31,0	31,1	31,2	31,3

Рис. 3.69. Размещение узлов графа (рис. 3.66) в ячейках ОЗУ (рис. 3.67)

не нашлось). На рисунке приведены также номера устройств (0—63), которым соответствуют узлы третьего уровня.

Рассмотрим процесс поиска номера свободного устройства. Последовательность событий следующая.

1. Блок управления (на рис. 3.67 не показан) формирует сигнал Установка кода 1, по которому в младшем разряде регистра RG1 устанавливается сигнал лог. 1, а в остальных шести разрядах — сигналы лог. 0. На адресные входы ОЗУ поступает код 00001<sub>2</sub> — указатель ячейки, соответствующей узлам 1,0—1,3 графа.

2. Из ОЗУ считывается 4-разрядный код и запоминается в регистре RG3 (управляющий вход этого регистра не показан).

3. Считанный из ОЗУ код анализируется приоритетным шифратором. Если код нулевой, то формируется признак  $P=1$ , означающий, что в системе нет ни одного свободного устройства. В этом случае поиск завершается. Если код, считанный из ОЗУ, содержит одну единицу, то на выходах приоритетного шифратора формируется номер ее позиции в ячейке памяти: 00, 01, 10 или 11. Если код содержит две, три или четыре единицы, то приоритетный шифратор выделяет крайнюю левую единицу (соответствующую самому верхнему из узлов данного уровня, установленных в 1, см. рис. 3.66). На выходах приоритетного шифратора формируется номер ее позиции.

4. Блок управления формирует сигнал Сдвиг влево, прием Н. По этому сигналу содержимое регистра RG1 за один такт сдвигается на два разряда влево, при этом в освободившиеся два младших разряда регистра принимается код Н. Фактически адрес умножается на 4 и складывается с кодом, установленным на выходах приоритетного шифратора. Это соответствует переходу к группе узлов более высокого уровня (см. рис. 3.66).

5. Если в старшем (левом) разряде регистра RG1 присутствует сигнал лог. 1 (Поиск завершен), то цель достигнута. Код W представляет собой номер искомого свободного устройства в диапазоне 0—63. Код в регистре RG3 показывает, имеются ли иные свободные устройства в выбранной четверке. (Эта информация может пригодиться в ближайшем будущем, когда вновь потребуются отыскать какое-либо свободное устройство.) Если в левом разряде регистра RG1 присутствует сигнал лог. 0 (число в регистре не превышает 64), то описанная процедура повторяется, начиная с шага 2.

Для коррекции состояний узлов графа при изменении состояния устройства (занято → свободно или свободно → занято) с заданным номером (0—63) используется следующая процедура.

1. На входы N схемы (рис. 3.67) подается 6-разрядный двоичный код номера устройства и сигнал лог. 1. Фактически на входах N устанавливается номер устройства, увеличенный на 64. По сигналу Прием N 7-разрядный код с входов N записывается в регистр RG1.

2. По сигналу Сдвиг вправо, прием F содержимое регистра RG1 за один такт сдвигается на два разряда вправо, при этом освободившиеся (старшие) разряды этого регистра заполняются нулями, а вытесненная старая информация (код F) из двух младших разрядов регистра RG1 запоминается в регистре RG2. Это соответствует делению числа на 4 с запоминанием остатка от деления в регистре RG2.

3. В соответствии с кодом, установленным на адресных входах ОЗУ, из него считывается 4-разрядное слово, которое запоминается в регистре RG3.

4. Код с выходов регистра RG3 обрабатывается преобразователем данных, так что значение одного из разрядов этого кода заменяется значением сигнала Q, поданного на вход преобразователя. При коррекции состояний узлов графа в связи с освобождением устройства  $Q=1$ , а при отображении перехода устройства в занятое состояние  $Q=0$ . Положение заменяемого разряда в слове определяется кодом M, поступающим из регистра RG2. Остальные разряды этого слова проходят сквозь преобразователь данных без изменения.

5. Код с выходов преобразователя данных записывается в ячейку ОЗУ по тому же адресу, который использовался при считывании. Иными словами,

корректируется состояние выбранного узла графа в четверке узлов, хранимой в ячейке ОЗУ.

6. Если сигнал Продолжить установлен в 1, то описанная процедура повторяется, начиная с шага 2. В противном случае коррекция состояний узлов графа завершается.

В схеме преобразователя данных (см. рис. 3.68) дешифратор DC выделяет позицию изменяемого разряда. При отображении перехода устройства из свободного в занятое состояние происходит гашение выбранной единицы в ненулевом 4-разрядном коде, проходящем сквозь преобразователь, так как  $Q=0$ . При этом возможны ситуации, при которых  $E=1$  и  $F=1$  (в группе остались узлы с ненулевым состоянием) или  $E=1$ ,  $F=0$  (погашена единственная единица в группе). Первая ситуация соответствует наличию сходящихся ветвей дерева, объединяющих узлы с единичными состояниями; ликвидируется одна из таких ветвей, в остальном дерево не претерпевает изменений, поэтому процедура завершается. Вторая ситуация соответствует уничтожению «длинной» единичной ветви дерева, точнее, продвижению по ней до тех пор, пока не будет обнаружено последующее объединение с другой единичной ветвью (если таковая имеется) или пока не будет достигнут уровень 1, после чего процедура завершается.

При отображении перехода устройства из занятого в свободное состояние происходит добавление единицы в выбранную дешифратором DC позицию 4-разрядного кода, проходящего через преобразователь, так как  $Q=1$ . При этом возможны ситуации, при которых  $E=1$  и  $F=1$  (ождается объединение единичных ветвей дерева) или  $E=0$ ,  $F=1$  (создается уникальная единичная ветвь). В первой ситуации процедура завершается, во второй продолжается до выявления объединения с другой единичной ветвью или до проведения коррекции состояния одного из узлов уровня 1. Число уровней графа и его иерархическая структура могут быть изменены при соответствующем изменении схемных решений.

### 3.25. Сжатие графической информации при ее передаче по последовательному каналу связи [71]

В двухмашинной системе (рис. 3.70, а) микроЭВМ1 и микроЭВМ2 обмениваются информацией по последовательному каналу связи. В частности, по этому каналу передается графическая информация для ее последующего отображения на экране дисплея, прорисовки на бумаге, редактирования и т. п.

Предположим, что микроЭВМ1 является источником, а микроЭВМ2 — приемником графической информации. Эта информация первоначально хранится в некоторой области ОЗУ микроЭВМ1, при этом каждый бит из этой области («экранной памяти») соответствует одной точке экрана дисплея. Будем считать, что нулевое значение бита экранной памяти соответствует цвету фона, а единичное — «рабочему» цвету. Например, фон может быть черным, а изображение — белым. Обычно каждой точке экрана соответствуют несколько бит экранной памяти, однако для описания предлагаемой идеи «сжатия» информации достаточно рассмотреть упрощенный пример.

Смысл предлагаемого решения состоит в том, чтобы вместо полного содержимого экранной памяти передавать по линии связи некоторую «свертку» изображения, в которой «плотно упакованы» повторяющиеся его фрагменты. Чем больше таких фрагментов, тем сильнее можно сжать информацию. Напротив, при попытке сжатия «случайной» информации, в которой нет или очень мало повторяющихся элементов, получим отрицательный результат, при котором экономичнее передавать по линии связи полное содержимое экранной памяти без какой-либо предварительной обработки. Однако согласно [71] в большинстве случаев графическая информация достаточно регулярна (редко содержит более 40% «несжимаемых» компонентов), что оправдывает применение рассмотренного далее метода.

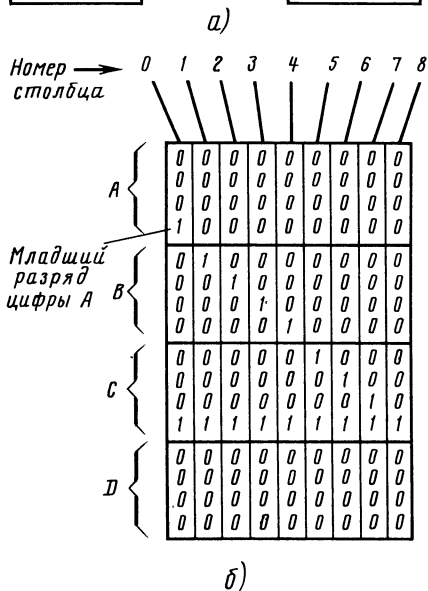
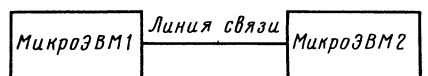


Рис. 3.70. Двухмашинная система (а) и фрагмент изображения (б), хранимого в экранной памяти микроЭВМ1

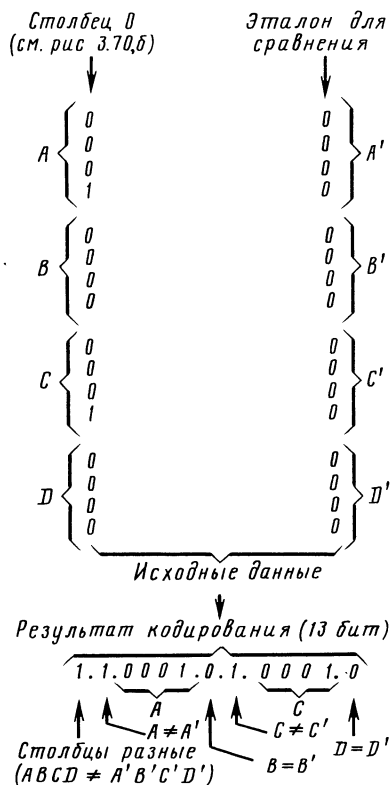


Рис. 3.71. Пример кодирования столбца 0 фрагмента изображения, показанного на рис. 3.70,б, при его сопоставлении с эталоном, содержащим нулевую информацию

Изображение делится на прямоугольные фрагменты. В данном примере (рис. 3.70,б) фрагмент содержит 9 столбцов и 16 строк. Каждый столбец представлен в экранной памяти четырьмя 4-разрядными числами: А, В, С, D. Столбец 0 отображается цифрами А, В, С, D, равными 1, 0, 1, 0 (здесь и далее цифры представлены в шестнадцатеричной записи в диапазоне 0—F), столбец 1—цифрами 0, 8, 1, 0, столбец 2—цифрами 0, 4, 1, 0 и т. д. Полный фрагмент изображения, показанный на рисунке, можно описать группой из  $9 \times 4 = 36$  4-разрядных чисел, если не применять сжатия информации.

При сжатии информации в зависимости от характера фрагмента изображения выбирается один из четырех способов его кодирования.

1. Все 9 столбцов последовательно сравниваются с эталонным, в котором  $A' = B' = C' = D' = 0$ . При выборе этого способа кодирования (подробности описаны далее) передаче фрагмента изображения в линию связи предшествует заголовки в виде двойной цифры 0.

2. Каждый столбец, за исключением столбца с номером 0, сравнивается с ближайшим соседним, размещенным слева, т. е. i-й столбец ( $i = 1, 2, \dots, 8$ ) сравнивается с (i-1)-м. Крайний левый столбец сравнивается с эталонным,

в котором  $A' = B' = C' = D' = 0$ . Если выбран этот способ кодирования, то заголовок принимается равным 10<sub>2</sub>.

3. Каждый столбец (с номером  $i$ ), за исключением столбцов с номерами 0 и 1, сравнивается с соседним слева, имеющим номер  $i-2$ . Столбцы с номерами 0 и 1 сравниваются с эталонным, в котором  $A' = B' = C' = D' = 0$ . При таком способе кодирования информация, выдаваемая в линию связи, начинается с заголовка 110<sub>2</sub>.

4. Если фрагмент изображения (9 столбцов, 16 строк) содержит только нулевые биты, то в линию связи выдается только заголовок 1110<sub>2</sub>, а какая-либо дополнительная информация об этом фрагменте не передается.

МикроЭВМ1 сопоставляет результаты кодирования, полученные разными способами, и выбирает наилучший, соответствующий наименьшей длине цепочки битов, которые должны быть переданы в микроЭВМ2 по линии связи. Рассмотрим условия кодирования по пп. 1—3 подробнее.

*Условие 1.* Если все четыре 4-разрядных числа  $A$ ,  $B$ ,  $C$ ,  $D$  данного столбца совпадают с соответствующими числами  $A'$ ,  $B'$ ,  $C'$ ,  $D'$  столбца, предъявленного для сравнения (столбец из нулей, столбец с номером  $i-1$  или  $i-2$  в зависимости от принятого способа кодирования), то в формируемую для передачи в микроЭВМ2 посылку включается бит, равный нулю.

*Условие 2.* Если имеется по крайней мере одно несовпадение (см. условие 1), то вместо бита, равного нулю, в формируемую посылку включается бит, равный 1.

*Условие 3.* При выполнении условия 2 числа  $A$ ,  $B$ ,  $C$  и  $D$  данного столбца последовательно сопоставляются с соответствующими числами  $A'$ ,  $B'$ ,  $C'$  и  $D'$  столбца, предъявленного для сравнения. При совпадении пары чисел ( $A=A'$ ,  $B=B'$ ,  $C=C'$ ,  $D=D'$ ) в формируемую посылку вводится бит, равный 0, при несовпадении — бит, равный 1, и вслед за ним — 4-разрядный двоичный код соответствующего числа  $A$ ,  $B$ ,  $C$  или  $D$ .

В качестве примера на рис. 3.71 показан процесс кодирования столбца 0 фрагмента изображения, приведенного на рис. 3.70, б. Этот столбец состоит из 16 бит; результат кодирования содержит 13 бит. Точки между разрядами результата введены лишь для удобства описания процесса кодирования — они разделяют код на части, каждая из которых соответствует одному из приведенных условий кодирования.

Так, левая единица результата свидетельствует о том, что столбец как целое не совпадает с предъявленным эталоном (см. условие 2). Следующая единица отражает тот факт, что число  $A$  не совпало с числом  $A'$ , а последующая группа из четырех разрядов (0001) отображает число  $A$  (см. условие 3) и т. д.

Вернемся к рассмотрению фрагмента изображения, показанного на рис. 3.70, б, и определим, какой из способов кодирования целесообразно применить для его сжатия. Сразу отметим, что способ 4 не подходит, так как фрагмент содержит как нулевые, так и единичные биты. Для удобства сопоставления разных способов кодирования представим наш фрагмент изображения более компактно, в виде матрицы из шестнадцатеричных чисел:

1	0	0	0	0	0	0	0	0
0	8	4	2	1	0	0	0	0
1	1	1	1	1	9	5	3	1
0	0	0	0	0	0	0	0	0

Применение способа 1 (сравнения столбцов с нулевой информацией) дает 14 несовпадений, по числу ненулевых чисел в матрице. Применение способа 2 (сравнения столбцов с ближайшими соседями слева) выявляет 12 несовпадений: 1—0, 1—0, 0—1, 8—0, 4—8, 2—4, 1—2, 0—1, 9—1, 5—9, 3—5, 1—3. При использовании способа 3 вновь получим 14 несовпадений: 1—0, 1—0, 8—0, 1—0, 0—1, 4—0, 2—8, 1—4, 0—2, 9—1, 0—1, 5—1, 3—9, 1—5. Таким образом, данный фрагмент изображения «сжимается» наиболее плотно, если использовать второй способ кодирования, в соответствии с которым получим следующий окончательный результат:

10:1.1.0001.0.1.0001.0:1.1.0000.1.1000.0.0:1.0.1.0100.0.0:1.0.1.0010.0.0:1.0.1.0001.0.0:1.0.1.0000.1.1001.0:1.0.0.1.0101.0:1.0.0.1.0011.0:1.0.0.1.0001.0:.

Напомним, что при использовании второго способа кодирования передаваемое сообщение о текущем фрагменте изображения начинается с заголовка  $10_2$ , затем следует информация о столбцах. Столбец с номером 0 кодируется группой из 13 бит (она уже рассматривалась на рис. 3.71), далее следуют группы из 13, 9, 9, 9, 13, 9, 9, 9 бит, описывающие столбцы с номерами 1—8. Границы групп отмечены в приведенной результирующей кодировке двоеточиями.

Сформированное сообщение (95 бит) передается по линии связи в микроЭВМ2, которая дешифрует заголовок и в соответствии с указанным в нем способом кодирования восстанавливает в собственной экранной памяти исходный фрагмент изображения ( $9 \times 16 = 144$  бит). В данном примере длина кодовой посылки по линии связи уменьшена на 34% по сравнению с длиной посылки, соответствующей непосредственной передаче фрагмента содержимого экранной памяти из микроЭВМ1 в микроЭВМ2.

### 3.26. Ускорение обмена данными между двумя микроЭВМ с раздельной синхронизацией [72]

Двухмашинная система (рис. 3.72, а) с общей синхронизацией от сигнала CLK позволяет главной микроЭВМ (1) обмениваться данными с адресуемыми элементами (ОЗУ, ПЗУ, регистрами и т. п.) подчиненной микроЭВМ (2) без дополнительных потерь времени, свойственных системам с раздельной синхронизацией. Чтобы показать происхождение таких потерь, рассмотрим сначала работу полностью синхронной системы на примере выполнения операции чтения (рис. 3.72, б).

В момент  $t_0$  микроЭВМ1 устанавливает в магистрали адрес A и признак чтения  $RD=1$ ; признак записи отсутствует, т. е.  $WR=0$ . В момент  $t_1$  из микроЭВМ2 в магистраль выдается искомое слово данных D и ответный сигнал ACK; в моменты  $t_2$  и  $t_3$  первая и вторая микроЭВМ возвращаются в исходные состояния. Считанное слово данных зафиксировано в микроЭВМ1.

Если микроЭВМ (1 и 2) синхронизируются разными сигналами — CLK1 и CLK2 (рис. 3.73), то нужна привязка «чужих» управляющих сигналов к «своим» синхросигналам. В данном примере для привязки используются цепи из двух последовательно включенных триггеров.

Напомним, что D-триггер с динамическим входом синхронизации принимает данные по положительному фронту синхросигнала. Для уверенной работы триггера необходимо, чтобы данные были стабильны в некотором интервале времени до момента поступления положительного фронта синхросигнала.

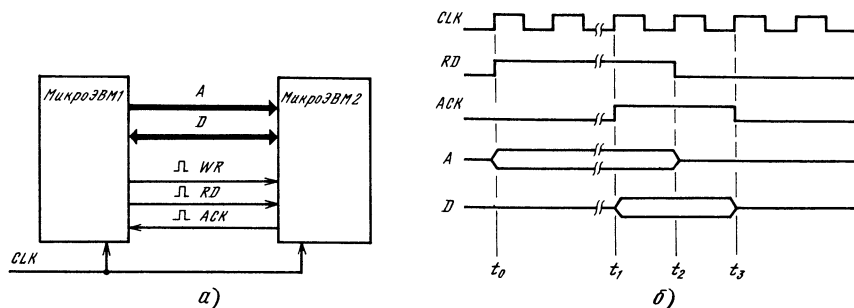


Рис. 3.72. Система из двух микроЭВМ с общей синхронизацией (а) и временные диаграммы считывания данных из микроЭВМ2 в микроЭВМ1 (б)

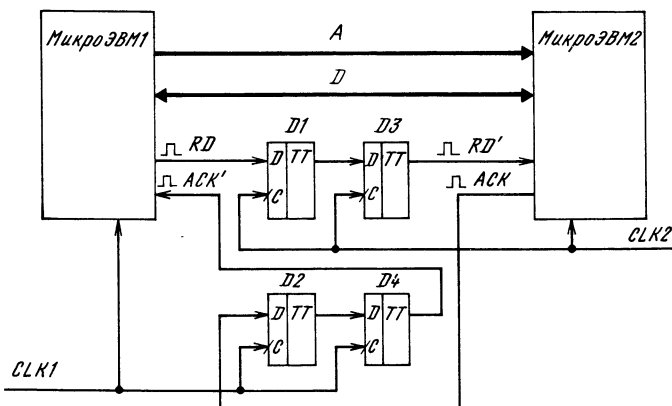


Рис. 3.73. Система из двух микроЭВМ с отдельной синхронизацией (первый вариант). Цепь передачи сигнала WR не показана

Если в этом интервале данные изменяются, то реакция триггера непредсказуема. Он может принять или не принять новый бит данных или даже сформировать на выходе очень короткий импульс, длительность и амплитуда которого недостаточны для «зашелкивания» триггера, т. е. для его перехода в новое устойчивое состояние.

Влияние этого импульса на схемы, подключенные к выходу триггера, может также оказаться непредсказуемым. Например, если к выходу триггера подключены два входа логических элементов одной серии, но размещенных в разных микросхемах, то из-за незначительных различий порогов срабатывания и разброса быстродействия один из этих элементов может воспринять короткий входной импульс, а другой останется в исходном состоянии.

Вернемся к схеме, показанной на рис. 3.73. Сигнал чтения RD вырабатывается синхронно с сигналом CLK1. Для правильной работы адресуемых элементов микроЭВМ2 сигнал чтения должен поступать синхронно с сигналом CLK2. Триггер D1 привязывает сигнал RD к фронту сигнала CLK2, однако, как отмечалось, на его выходе возможно появление кратковременных импульсов, когда триггер не успевает перейти в новое состояние. Чтобы устранить эти импульсы, применен триггер D3. Цепь D1—D3 образует обычный сдвиговый регистр. Короткий ложный импульс с выхода триггера D1 не может повлиять на состояние триггера D3, так как этот импульс опаздывает к моменту прохождения положительного фронта сигнала CLK2.

Цепи D1—D3 и D2—D4 привязки сигналов чтения и ответа (цепь записи не рассматриваем) вносят задержки в цикл обмена данными между микроЭВМ. В зависимости от текущего соотношения фаз синхросигналов задержка передачи данных в цепи D1—D3 (D2—D4) может составить от одного до двух периодов сигнала CLK2 (CLK1). Этим и вызваны упоминавшиеся потери быстродействия при обмене данными между микроЭВМ. Предлагаемое решение (рис. 3.74) позволяет в большинстве случаев исключить из цикла обмена задержку передачи ответного сигнала ACK.

В схему введен блок F предсказания момента готовности считанных данных. Этот блок может быть выполнен в виде комбинационной схемы, ПЗУ, ПЛМ и т. п. На его входы поступают старшие разряды адреса, с выходов снимаются сигналы Y0—Y3, прогнозирующие задержку поступления данных из микроЭВМ2 при считывании. Идея состоит в том, что разработчик системы, зная схемотехнику и элементную базу адресуемых блоков микроЭВМ2, может составить таблицу, в которой каждому блоку поставлено

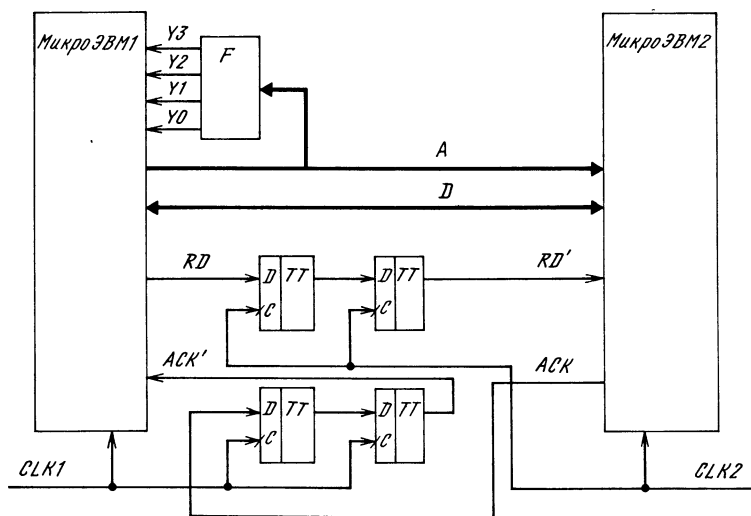


Рис. 3.74. Система из двух микроЭВМ с раздельной синхронизацией (второй вариант).  
Цепь передачи сигнала WR не показана

в соответствие время его срабатывания (единица измерения—период сигнала CLK2). Эта таблица «материализуется» в блоке F, так что микроЭВМ1, обращаясь по тому или иному адресу, заранее «знает», сколько времени нужно ждать получения данных.

Например, быстродействующее статическое ОЗУ, размещенное в некоторой (заранее известной) области адресного пространства микроЭВМ2, срабатывает за время, равное одному периоду сигнала CLK2 (начало отсчета—момент поступления сигнала RD'), динамическое ОЗУ—за время, равное трем периодам этого сигнала, и т. д. Анализируя старшие разряды адреса, блок F фактически сортирует адресуемые блоки по быстродействию. В данном примере блок F имеет 4 выхода, поэтому можно задать  $2^4=16$  кодов, характеризующих быстродействие адресуемых блоков.

Один из кодов, например нулевой, может выделяться для расширения системы, когда заранее не известно, какие платы будут устанавливаться в свободные разъемы микроЭВМ2. При обращении по адресам, выделенным этим платам, нулевой код с выхода блока F информирует микроЭВМ1 о том, что обмен должен протекать по обычной, «медленной» схеме, когда истинность считанных данных подтверждается сигналом ACK'. Однако, после того как конфигурация микроЭВМ2 «устоялась» (платы расширения одни и те же), можно дополнить кодировку блока F соответствующей информацией о быстродействии новых плат, чтобы отказаться от ожидания сигнала ACK' при работе с ними.

Момент готовности данных вычисляется аппаратурой микроЭВМ1. За исходную точку отсчета времени принимается момент формирования сигнала RD=1. Как было показано, сигнал RD'=1 формируется в худшем случае через интервал времени T, равный двум периодам сигнала CLK2. Поэтому в качестве первого слагаемого времени ожидания всегда выбирается T. Второе слагаемое зависит от быстродействия адресуемого устройства и задается сигналами Y0—Y3. Таким образом, из цикла обмена исключаются один—два периода сигнала CLK1, т. е. устраняются потери времени передачи сигнала ACK из микроЭВМ2 в микроЭВМ1.

### 3.27. Двусторонний обмен данными между главной и подчиненной микроЭВМ по общей информационной шине [98]

В схеме (рис. 3.75) микроЭВМ1 главная, микроЭВМ2 подчиненная. В исходном состоянии обе микроЭВМ работают автономно и не обмениваются между собой какой-либо информацией. Сеанс связи между ними возможен как по инициативе первой, так и второй микроЭВМ с использованием общей шины данных D. МикроЭВМ-инициатор обмена пересылает микроЭВМ-абоненту одно слово данных и освобождает шину. При возникновении конфликтов из-за одновременной попытки захвата общей шины обеими микроЭВМ всегда выигрывает микроЭВМ1. Рассмотрим межмашинный обмен подробнее.

Когда шина данных не занята,  $C0=0$ ,  $C1=C2=1$  (принята положительная логика). В момент  $t_0$  (рис. 3.76) микроЭВМ1 устанавливает в верхней управляющей линии сигнал  $C0=1$ , который передается в микроЭВМ2 и оповещает ее о предстоящем сеансе связи. Этот сигнал запрещает подчиненной микроЭВМ захватывать шину данных, что существенно при устранении конфликтов. Пока, однако, предполагаем, что конфликтов нет.

В момент  $t_1$  микроЭВМ1 посылает в микроЭВМ2 сигнал  $C1=0$ , отрицательный фронт которого рассматривается как запрос готовности микроЭВМ2 к приему данных. Если готовность есть, то микроЭВМ2 формирует ответный сигнал  $C2=0$  (момент  $t_2$ ), получив который микроЭВМ1 в момент

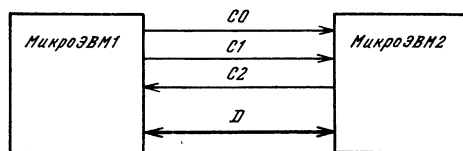


Рис. 3.75. Система из двух микроЭВМ

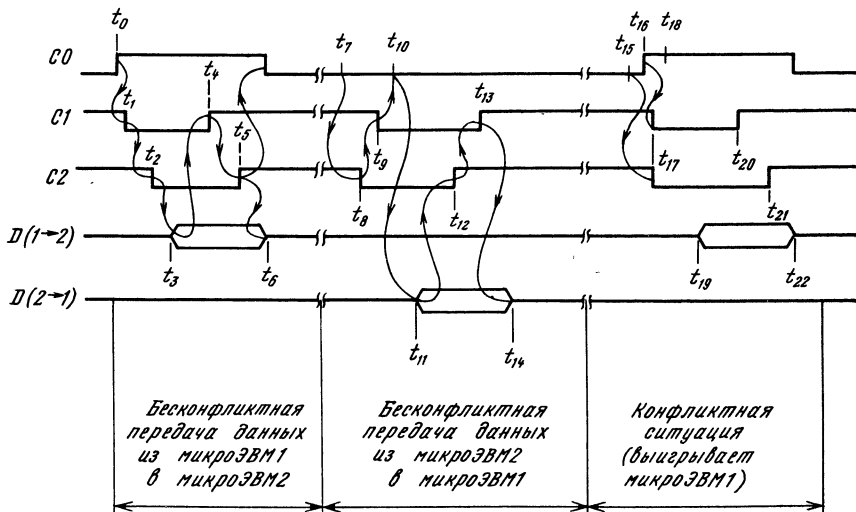


Рис. 3.76. Временные диаграммы пересылки данных между микроЭВМ

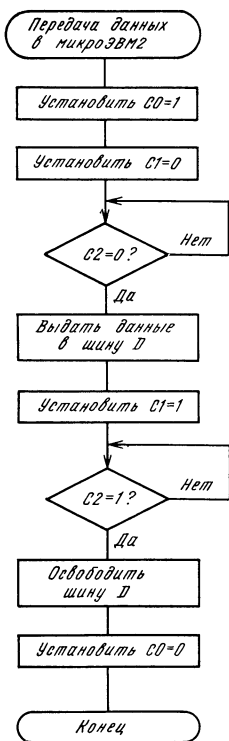


Рис. 3.77. Блок-схема алгоритма передачи данных из микроЭВМ1 в микроЭВМ2

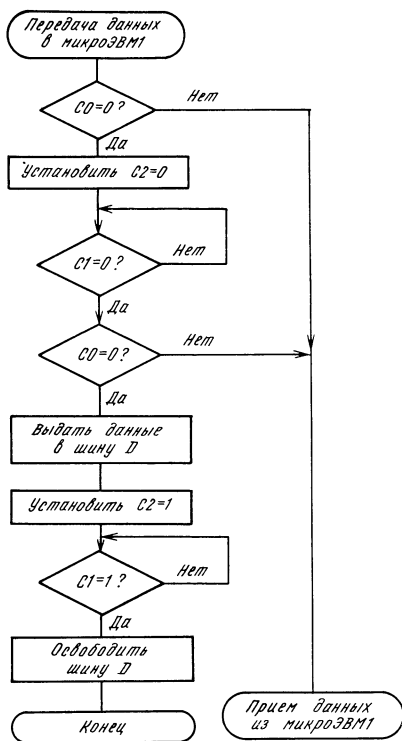


Рис. 3.78. Блок-схема алгоритма передачи данных из микроЭВМ2 в микроЭВМ1

$t_3$  устанавливает в шине данных передаваемый код  $D$  (1→2). После окончания переходных процессов в шине данных микроЭВМ1 подтверждает истинность кода  $D$  (1→2) положительным фронтом сигнала  $C1$  (момент  $t_4$ ).

Получив это подтверждение, микроЭВМ2 принимает код  $D$  (1→2) с шины данных и в момент  $t_5$  положительным фронтом сигнала  $C2$  уведомляет микроЭВМ1 о завершении приема. Восприняв уведомление, микроЭВМ1 проверяет состояние сигнала  $C0$  и снимает сигнал  $C0$  (момент  $t_6$ ). Рассмотренные процессы (период  $t_0$ — $t_6$ ) поясняются блок-схемой алгоритма передачи данных из микроЭВМ1 в микроЭВМ2, приведенной на рис. 3.77.

Передача данных в обратном направлении (из микроЭВМ2 в микроЭВМ1) протекает аналогично, однако микроЭВМ2 ведет себя менее «решительно» и не накладывает каких-либо ограничений на работу микроЭВМ1 (нет аналога сигнала  $C0$  со стороны микроЭВМ2). В момент  $t_7$  (см. рис. 3.76) микроЭВМ2 проверяет состояние сигнала  $C0$  и, поскольку запрета нет ( $C0=0$ ), в момент  $t_8$  устанавливает сигнал  $C2=0$ —запрос готовности микроЭВМ1 к приему данных. В момент  $t_9$  микроЭВМ1 сигналом  $C1=0$  подтверждает эту готовность.

В момент  $t_{10}$  микроЭВМ2 повторно проверяет состояние сигнала  $C0$ . Так как мы по-прежнему рассматриваем бесконфликтную передачу данных, повторная проверка показывает отсутствие запрета выдачи кода в общую шину ( $C0=0$ ). В момент  $t_{11}$  микроЭВМ2 устанавливает в общей шине

данных код  $D(2 \rightarrow 1)$ , в момент  $t_{12}$  истинность этого кода подтверждается положительным фронтом сигнала  $C2$ .

К моменту  $t_{13}$  микроЭВМ1 заканчивает прием данных с шины и сообщает об этом положительным фронтом сигнала  $C1$ . МикроЭВМ2, в свою очередь, в момент  $t_{14}$  освобождает общую шину, сеанс связи заканчивается. Процесс передачи данных в период  $t_7 - t_{14}$  поясняется блок-схемой алгоритма, приведенной на рис. 3.78.

Конфликтные ситуации возникают при одновременной (или почти одновременной) попытке обеих микроЭВМ воспользоваться общей шиной для выдачи данных. В момент  $t_{15}$  микроЭВМ2 проверяет состояние сигнала  $C0$ . Так как  $C0=0$ , эта микроЭВМ приступает к обмену и в момент  $t_{17}$  устанавливает сигнал  $C2=0$ . Однако незадолго до этого, в момент  $t_{16}$ , микроЭВМ1 решает провести сеанс связи с микроЭВМ2 и последовательно формирует сигналы  $C0=1$ ,  $C1=0$ .

МикроЭВМ2 убеждается в наличии сигнала  $C1=0$ . Пока, с ее точки зрения, процесс захвата общей шины развивается нормально, сигнал  $C1=0$  рассматривается как ответ от микроЭВМ1, свидетельствующий о готовности к приему данных, хотя ситуация изменилась и этот сигнал уже «не тот» — он отражает начальный этап сеанса связи, проводимого по инициативе микроЭВМ1. Истина выясняется в момент  $t_{18}$ , когда микроЭВМ2 повторно проверяет сигнал  $C0$ .

К этому моменту появилась полная определенность в оценке развития событий. Действительно, сигнал  $C1=0$  при  $C0=1$  соответствует встречному обращению со стороны микроЭВМ1, а при  $C0=0$  — пассивному ответу на запрос готовности. (При этом существенно, что выдача сигнала  $C0=1$  предшествует формированию сигнала  $C1=0$ , см. рис. 3.77. Поэтому микроЭВМ2 не может вторично ошибиться в оценке реальной ситуации.) В данном примере в момент  $t_{18}$   $C0=1$ , поэтому микроЭВМ2 прекращает попытку проведения сеанса связи и переходит к программе приема данных от микроЭВМ1.

МикроЭВМ1, по праву считая себя «главной», уверена в правомерности захвата ею шины данных и продолжает работу по рассмотренному ранее алгоритму (см. рис. 3.77). Так как  $C2=0$ , то в момент  $t_{19}$  через шину данных начинает передаваться код  $D(1 \rightarrow 2)$ , в момент  $t_{20}$  подтверждается его истинность, в момент  $t_{21}$  поступает информация о том, что код принят, в момент  $t_{22}$  шина данных освобождается и снимается сигнал  $C0$ . Конфликт устранен, выиграла микроЭВМ1.

## Глава 4

### Аппаратная поддержка вычислений, узлы микроЭВМ

#### 4.1. Выравнивание задержек каналов усиления синхросигналов [73]

При передаче синхронизирующего или иного сигнала в удаленные друг от друга точки печатной платы или многоплатного блока обычно используют группу магистральных усилителей с объединенными входами. Если усилители размещены в разных микросхемах, то из-за разброса задержек (например, от 5 до 20 нс) выходные сигналы будут смещены относительно друг друга (в худшем случае на  $20 - 5 = 15$  нс). Чем больше смещение, тем более

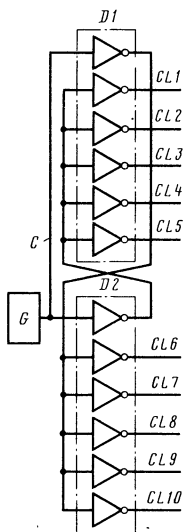


Рис. 4.1. Схема выравнивания задержек каналов усиления синхросигналов

«неопределенными» становятся временные диаграммы работы устройства, тем дольше в каждом такте нужно ждать установления правильных данных в расчете на наихудший случай и т. п.

Чтобы выровнять сигналы в разных каналах, обычно применяют дополнительные элементы задержки с подстроечными элементами — переменными сопротивлениями или конденсаторами. Это создает неудобства при изготовлении и эксплуатации устройства, так как нужна регулировка задержек как при первом включении, так и после замены отказавших микросхем усилителей либо в связи с «уходом» параметров элементов за счет старения.

В предлагаемой схеме (рис. 4.1) регулировок не требуется; разброс задержек выходных сигналов CL1—CL10 уменьшен по сравнению с рассмотренным вариантом использования усилителей с объединенными входами.

Предположим, что самый «быстрый» инвертор микросхемы D1 имеет задержку 5 нс. Тогда задержка самого «медленного» инвертора этой микросхемы не превысит 5,5 нс, так как разброс задержек инверторов, размещенных в одной микросхеме, не превышает 10%. Однако если самый «медленный» инвертор микросхемы D2 имеет задержку 20 нс, то задержка самого «быстрого» инвертора этой микросхемы окажется не менее 18 нс. Сигнал С с тактового генератора G при преобразовании в сигналы CL1—CL10 проходит через обе микросхемы, поэтому с точки зрения «преодоления задержек» обе группы сигналов (CL1—CL5 и CL6—CL10) находятся в равных условиях.

В данном примере взаимный разброс сигналов CL1—CL10 не превышает 2,5 нс, так как самая «быстрая» суммарная цепь из двух последовательно включенных инверторов срабатывает за  $5 + 18 = 23$  нс, а самая «медленная» — за  $5,5 + 20 = 25,5$  нс. Максимальный взаимный разброс сигналов CL1—CL10, равный 4 нс, может наблюдаться при установке на плату двух «медленных» микросхем D1 и D2 с задержками инверторов 18—20 нс.

## 4.2. Уменьшение потерь времени при формировании сигнала подтверждения истинности кода, выдаваемого в общую шину [74]

При работе вычислительных устройств, построенных с использованием общей магистрали, передаваемые по общим шинам коды обычно сопровождаются сигналами в управляющих линиях подтверждения истинности этих кодов. Далее для определенности рассматривается адресная шина общей магистрали и управляющая линия подтверждения истинности адреса.

В традиционной схеме, показанной на рис. 4.2, активное устройство (процессор, контроллер) захватывает общую магистраль согласно некоторому протоколу (протокол не рассматриваем) и приступает к адресному обмену с пассивным устройством — памятью, регистром и т. п. Подготовленный активным устройством 20-разрядный адресный код поступает на входы трех 8-разрядных магистральных усилителей (микросхемы D2—D4), выполненных по схеме с открытым коллектором. Шина адреса и управляющая линия подтверждения его истинности подключены через резисторы к источнику напряжения +U, поэтому в пассивном состоянии в шине и управляющей линии присутствуют сигналы лог. 1 — напряжения высокого уровня.

После подготовки адреса активное устройство формирует сигнал выборки SEL=0, который разрешает работу усилителей D2—D4. Этот же сигнал

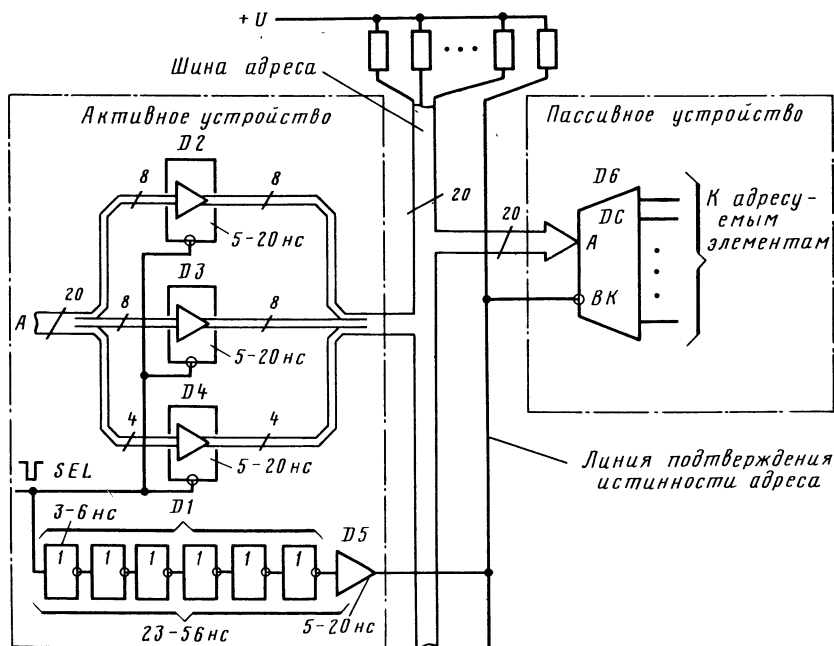


Рис. 4.2. Схема выдачи сигналов в шину адреса.

Сигнал подтверждения истинности адреса гарантирует правильную работу дешифратора  $D6$ , однако возможны значительные потери времени при неблагоприятном сочетании задержек микросхем  $D1-D5$

поступает в линию задержки, выполненную на шести инверторах (микросхема  $D1$ ) и усилителе (микросхема  $D5$ ) с открытым коллектором.

Через некоторое время после формирования сигнала  $SEL=0$ , а именно после срабатывания самого «медленного» магистрального усилителя из группы  $D2-D4$ , в шине адреса устанавливается правильный код, а затем в управляющей линии формируется сигнал лог. 0, подтверждающий истинность адресного кода. Чтобы не нарушалась указанная последовательность событий, суммарная задержка распространения сигнала  $SEL$  через микросхемы  $D1$  и  $D5$  не должна быть меньше, чем время реакции на этот сигнал со стороны самого «медленного» из усилителей  $D2-D4$ . В противном случае дешифратор адреса  $D6$  пассивного устройства получит сигнал лог. 0 на вход выбора кристалла ( $ВК$ ) слишком рано, когда в шине адреса еще не окончились переходные процессы. Это может привести к неправильному выбору адресуемого элемента пассивного устройства.

Пусть минимальная и максимальная задержка усилителей  $D2-D4, D5$  составляет 5 и 20 нс, а задержка одного инвертора микросхемы  $D1$  3—6 нс. В худшем случае, когда усилители  $D2-D4$  «медленные», а инверторы  $D1$  и усилитель  $D5$  «быстрые», адрес устанавливается в общей шине через 20 нс, а сигнал подтверждения истинности адреса — через 23 нс после формирования сигнала  $SEL=0$ . Поэтому соотношение задержек правильное. Однако если микросхемы  $D2-D4$  «быстрые», а микросхемы  $D1$  и  $D5$  «медленные», то создается ненужный запас времени. Действительно, код в шине адреса готов уже через 5 нс после выдачи сигнала  $SEL=0$ , а признак подтверждения истинности адреса формируется только через 56 нс. Иными словами,



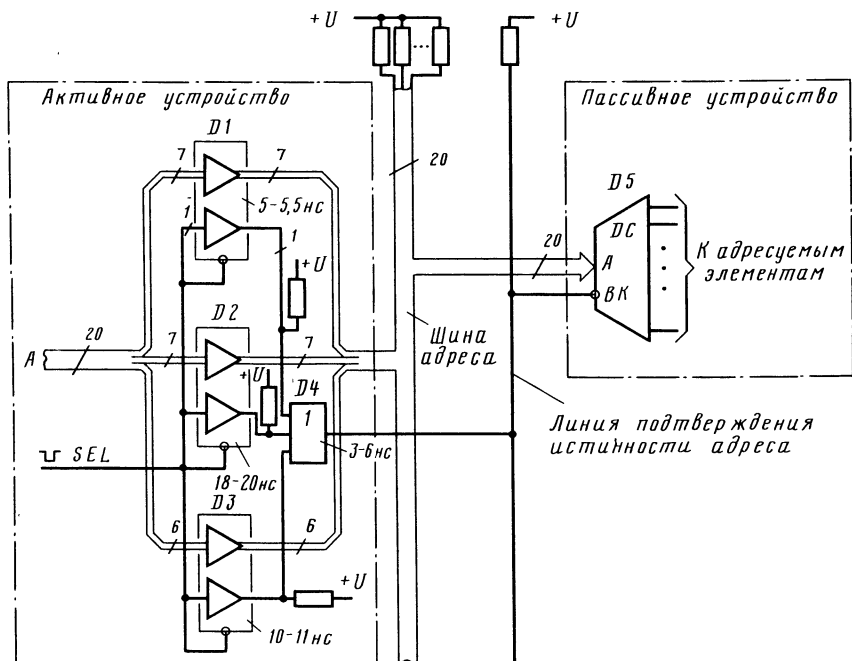


Рис. 4.4. Второй вариант усовершенствованной схемы передачи адреса.

Элемент ИЛИ D4 компенсирует разность задержек (а не абсолютную задержку) усилителей, размещенных в одной микросхеме, и выполняет функцию объединения управляющих сигналов

сигналы лог. 0. Эти сигналы поступают на входы ВК дешифратора D5 и открывают его. В данном случае сигналы подтверждения истинности адреса задержаны, как минимум, на 3 нс элементом И D1 и на 5, 18 и 10 нс элементами, размещенными в микросхемах D2—D4. Все три сигнала лог. 0 смогут одновременно присутствовать в линиях шины подтверждения истинности адреса не ранее чем через  $3 + 18 = 21$  нс после выдачи сигнала  $SEL = 0$ , что вполне удовлетворительно (напомним, что адрес устанавливается в общей шине не позже чем через 20 нс после формирования сигнала  $SEL = 0$ ).

Потери времени при неблагоприятном сочетании «быстрых» микросхем D2—D4 с «медленной» микросхемой D1 не столь велики, как в схеме, приведенной на рис. 4.2. Действительно, в данном случае адрес формируется на общей шине через 5 нс, а его подтверждение — через  $6 + 5,5 = 11,5$  нс, потери на ожидание составляют  $11,5 - 5 = 6,5$  нс, а не 51 нс, как в предыдущей схеме. Выигрыш достигается за счет того, что управляющие сигналы распространяются в той же «среде», что и адресные сигналы. Элемент задержки D1 компенсирует не абсолютную максимальную задержку микросхем D2—D4, а относительный разброс («перекося») задержек, не превышающий, как было показано, 2 нс.

В схеме, приведенной на рис. 4.4, элемент ИЛИ D4 с открытым коллектором выполняет логическую функцию объединения сигналов подтверждения истинности адреса и компенсирует относительный разброс задержек усилителей аналогично элементу И D1 в схеме, показанной на рис. 4.3. Это позволило вернуться к варианту построения схемы с единственной линией подтверждения истинности адреса.

### 4.3. Повышение пропускной способности магистрали с третьим состоянием [75]

При построении магистралей микроЭВМ наиболее часто используют линии с третьим состоянием или с открытым коллектором (эмиттером). Элемент-передатчик с третьим (выключенным) состоянием либо отключен от линии, либо формирует в ней напряжение низкого или высокого уровня (активный 0 или активная 1).

Такой элемент привлекателен тем, что он, во-первых, способен быстро перезаряжать паразитную емкость линии благодаря низкому выходному сопротивлению (от единиц до десятков ом) и, во-вторых, обеспечивает низкое энергопотребление из-за отсутствия внешнего нагрузочного резистора. В установившемся режиме передачи сигнала лог. 0 или лог. 1 открыт либо «нижний», либо «верхний» транзистор выходного каскада; в третьем состоянии оба транзистора закрыты.

Однако линия с третьим состоянием имеет недостаток, связанный с необходимостью строгого разделения во времени сигналов, поступающих от разных передатчиков. При наложении сигналов лог. 0 и лог. 1 от двух передатчиков возникают электрические перегрузки их выходных каскадов, когда в одном открыт «верхний», в другом — «нижний» транзисторы. В результате эти транзисторы работают в режиме, близком к короткому замыканию, что недопустимо; от источника питания в этот период отбирается значительный ток.

Чтобы исключить одновременную работу на линию двух (или более) передатчиков, нужно сначала выключить первый, а затем включить второй. Интервал времени между этими событиями должен гарантировать отсутствие наложений сигналов в наихудшем случае, когда выключается «медленный», а включается «быстрый» передатчик (соответствующие задержки могут составлять 20 и 5 нс, если передатчики размещены в разных микросхемах одного типа). Этот интервал времени добавляется к каждому циклу обмена (при смене источника сигнала) и, следовательно, снижает пропускную способность магистрали.

Линия с открытым коллектором через нагрузочный резистор подключена к шине питания и допускает наложение сигналов от разных источников. Поэтому нет потерь времени, связанных с необходимостью разделения сигналов, как в линии с третьим состоянием. Сигнал лог. 0 поддерживается в линии за счет насыщения хотя бы одного из транзисторов; сигнал лог. 1 формируется после заряда паразитной емкости линии через нагрузочный резистор, когда все передатчики выключаются. В установившемся состоянии лог. 1 ток через нагрузочный резистор незначителен.

Чтобы ускорить заряд паразитной емкости линии при переключении передатчика с открытым коллектором из 0 в 1, следует уменьшить сопротивление нагрузочного резистора, но при этом растет рассеиваемая на нем мощность в состоянии лог. 0. Таким образом, линии обоих типов (с третьим состоянием и открытым коллектором) далеко не совершенны; каждая имеет определенные преимущества и недостатки.

Предлагается улучшить временные параметры линии магистрали с третьим состоянием подключением к ней триггера на усилителе TR с резистором в цепи обратной связи (рис. 4.5). Усилитель имеет активный выход, т. е. он формирует сигналы лог. 0 и лог. 1 в виде напряжений низкого и высокого уровней. Резистор может быть достаточно высокоомным; он служит лишь для поддержания в линии напряжения, ранее установленного одним из передатчиков (а не для перезаряда паразитной емкости линии).

В исходном состоянии, до момента  $t_0$ , передатчики всех блоков 1—N микропроцессорной системы выключены соответствующими сигналами  $E_1 = E_2 = \dots = E_N = 0$ . В линии D присутствует поступающий с триггера сигнал лог. 0 или лог. 1.

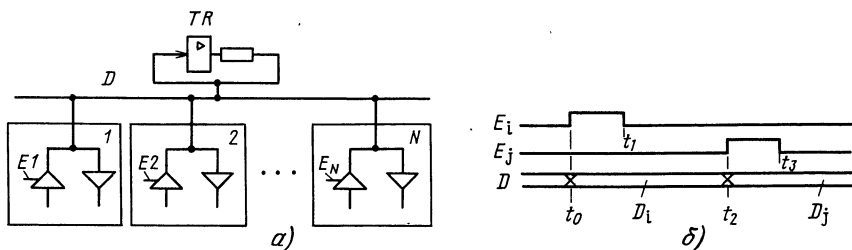


Рис. 4.5. Линия магистрали с увеличенной пропускной способностью (а) и временные диаграммы ее работы (б)

В момент  $t_0$  формируется сигнал  $E_i$  разрешения работы передатчика (с третьим состоянием), размещенного в блоке  $i$ . Сигнал активного 0 или 1 с выхода передатчика поступает в линию  $D$ . Триггер не может воспрепятствовать этому, так как его выходное сопротивление велико и он «терпит поражение» при возникающем конфликте с более «сильным» соперником — низкоомным источником сигнала  $D_i$  (при совпадении старого и нового сигналов в линии конфликтов нет). Поэтому триггер устанавливается в нужное состояние и «в меру своих сил» (через высокоомный резистор) помогает поддержанию в линии нужного напряжения, хотя необходимости в такой помощи пока нет.

В момент  $t_1$  сигнал  $E_i=1$  разрешения работы передатчика блока  $i$  снимается, хотя передача бита  $D_i$  абоненту не завершена, точнее, только начата; через некоторое время  $\tau(t_1 < \tau < t_2)$  этот передатчик выключается, т. е. его выход переходит в высокоомное состояние. Однако в отличие от традиционного решения данные  $D_i$  в линии сохраняются благодаря «поддержке» со стороны триггера — его низкая нагрузочная способность все же достаточна для привязки напряжения в линии к нужному уровню.

В момент  $t_2$  передача бита  $D_i$  между блоками завершается, в действие вступает новый передатчик с номером  $j$ , который «почти сразу же», в момент  $t_3$ , выключается и т. д. Таким образом, благодаря «подстраховке» данных триггером переход передатчика из активного в выключенное состояние происходит «скрытно», в тот период, пока идет обмен данными, а не в промежутках между обменами.

#### 4.4. Быстродействующая КМОП-магистраль [76]

В схеме (рис. 4.6) магистраль используется для передачи информационных и управляющих сигналов между платами 1— $N$ , размещенными в едином конструктивном блоке (показан один разряд магистрали). В общем случае каждая плата содержит передатчик и приемник типа КМОП. Каждая линия магистрали выполнена в виде печатного проводника на кроссплате. Платы 1— $N$  устанавливаются в разъемы, размещенные на кроссплате.

Элементы-передатчики выполнены по схеме с «третьим состоянием»: при  $F=0$  передатчик выключен, его выходное сопротивление велико (порядка 1 МОм), а при  $F=1$  включен, его выход можно представить генератором напряжения (0 или +5 В) с последовательно соединенным резистором сопротивлением от 2 до 10 Ом. Входное сопротивление элементов-приемников велико, и им можно пренебречь. В схеме использованы резисторы сопротивлением 150 и 30 Ом с разбросом в пределах 1%.

В отличие от традиционной схемы, в которой резисторы  $R_1$  и  $R_2$  отсутствуют, а остальные короткозамкнуты, данное решение имеет три основных преимущества.

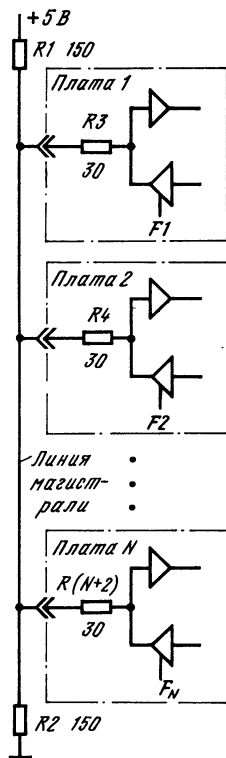


Рис. 4.6. Быстродействующая КМОП-магистраль (показан один разряд).

Резисторы R1 и R2 подключены к физическим концам линии магистрали

1. Полное сопротивление линии составляет 30—60 Ом, выходное сопротивление передатчика с последовательно соединенным резистором равно примерно 30 Ом, т. е. обеспечивается хорошее согласование источника сигнала с линией, способствующее уменьшению искажений формы сигнала.

2. Паузу между сигналом выключения передатчика платы  $i$  и сигналом включения передатчика платы  $j$  можно уменьшить и даже исключить. При этом может случиться так, что в течение некоторого времени передатчик платы  $i$  еще не выключился, а передатчик платы  $j$  уже включился. Однако благодаря резисторам в данном случае предотвращается короткое замыкание выходов, несущих противоположные сигналы, поэтому нет опасности выхода из строя передатчиков, как при использовании традиционной схемы. Нет и больших импульсных токов короткого замыкания, сопровождающих описанный режим, поэтому не генерируются помехи, связанные с этими токами. Иными словами, передаваемые в разных сеансах связи сигналы можно максимально сблизить во времени без появления каких-либо нежелательных «побочных эффектов», что равносильно повышению пропускной способности магистрали.

3. Благодаря резисторам  $R_3—R(N+2)$  скорость изменения тока ( $di/dt$ ) через передатчик при его включении—выключении снижается в 5—10 раз, поэтому в такой же степени уменьшается влияние паразитных индуктивностей сигнальной цепи, шин питания и «земли». Следовательно, подавляются

помехи, создаваемые передатчиком.

Недостаток схемы также связан с введением резисторов—уменьшен размах сигналов в линии магистрали. Сигналы лог. 0 и лог. 1 представлены соответственно уровнями 0,8 и 4,2 В (вместо 0 и 5 В в традиционной схеме), а в наихудшем случае—уровнями 1,3 и 3,7 В. Если приемник расценивает входные сигналы, меньшие 2 В и большие 3 В, как лог. 0 и лог. 1, то запас помехоустойчивости в расчете на наихудший случай составляет 0,7 В по каждому из уровней.

#### 4.5. Выходной каскад КМОП-элемента, обеспечивающий низкое потребление энергии при сопряжении с входами ТТЛ-и КМОП-элементов [77]

Элемент КМОП-типа (рис. 4.7, а) содержит входную цепь L и выходной каскад на р- и n-канальных МОП-транзисторах. При  $CS=1$  (активное состояние) входной сигнал D транслируется на выход элемента, при этом в статике открыт либо только верхний, либо только нижний транзистор; при  $CS=0$  (пассивное состояние) независимо от значения D верхний транзистор открыт, нижний закрыт.

Проблема состоит в том, что при сопряжении с ТТЛ-элементом при  $CS=0$  через р-канальный транзистор протекает ток I, т. е. потребляется энергия

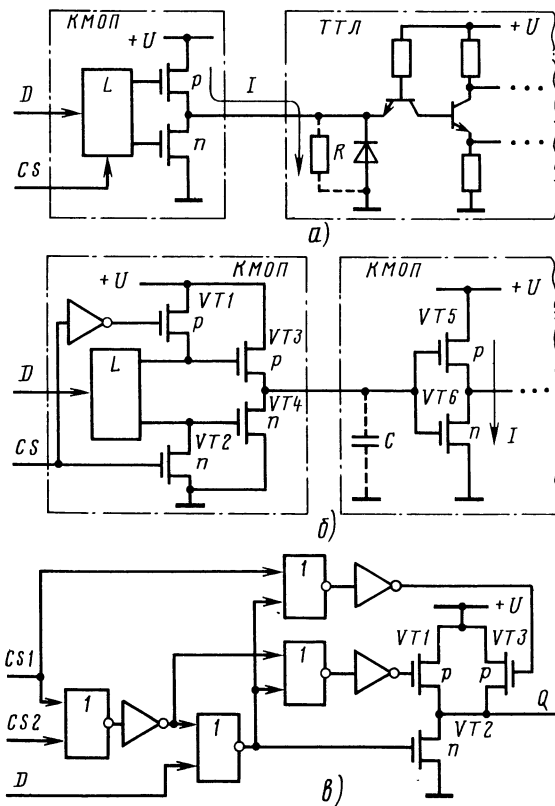


Рис. 4.7. Примеры неудачного сопряжения элементов КМОП—ТТЛ (а) и КМОП—КМОП (б); предлагаемая схема универсального выходного каскада (в)

от источника питания. (ТТЛ-элемент при подаче на его вход напряжения высокого уровня можно представить некоторым эквивалентным сопротивлением  $R$ .) Поэтому при сопряжении КМОП- и ТТЛ-элементов следует использовать выходной каскад с «третьим состоянием», при котором верхний и нижний транзисторы одновременно выключены и ток  $I$  практически отсутствует.

Однако КМОП-элемент с «третьим состоянием» также «небезупречен». Действительно, если в качестве приемника применен КМОП-элемент (рис. 4.7, б), то при  $CS=1$  транзисторы  $VT1$  и  $VT2$  включены,  $VT3$  и  $VT4$  выключены, паразитная емкость  $C$  хранит некоторый заряд, напряжение на ней может соответствовать пороговому, при котором транзисторы  $VT5$  и  $VT6$  находятся в активном режиме, через них протекает значительный «сквозной» ток  $I$ . Помимо увеличения энергопотребления этот режим может привести к отказу транзистора  $VT5$  или  $VT6$  из-за перегрева. Поэтому в данном случае следовало бы использовать выходной каскад КМОП-элемента, показанного на рис. 4.7, а,—паразитная емкость  $C$  была бы гарантированно заряжена до уровня лог. 1 и «сквозной» ток  $I$  отсутствовал.

Таким образом, каждый из выходных каскадов (рис. 4.7, а, б) по-своему хорош (плох). При сопряжении КМОП- с ТТЛ-элементом следует использовать выходной каскад с «третьим состоянием» (если допустима помеха на входе

ТТЛ-элемента, связанная с «оборванным» входом); при сопряжении с КМОП-элементом необходимо применять выходной каскад с гарантированным сигналом лог. 1 или лог. 0 на выходе в пассивном режиме. (Привязка выходного каскада с «третьим состоянием» через резистор к шине питания или «земли» — не лучшее решение, так как при нормальной работе это создает дополнительное потребление тока.)

Так как во многих ситуациях заранее не известно, какие периферийные устройства будут подключаться к выходному каскаду, предлагается спроектировать его так, чтобы он был пригоден для сопряжения как с ТТЛ-, так и КМОП-элементами (рис. 4.7, в).

При  $CS1=CS2=0$  сигнал D передается на выход Q, транзисторы VT1 и VT3 работают синхронно. Оба они включены, когда  $D=Q=1$ , и выключены, когда  $D=Q=0$ . Состояние транзистора VT2 противоположно состоянию VT1 (VT3).

При сопряжении этой схемы с ТТЛ-элементом пассивный режим сопровождается сигналами  $CS1=1$ ,  $CS2=0$ . Транзисторы VT1—VT3 выключены, выход Q находится в высокоомном состоянии, входная цепь ТТЛ-элемента (не показана) изолирована от шины питания +U.

Если в качестве нагрузки использован КМОП-элемент, то в пассивном состоянии на управляющие входы схемы подаются сигналы  $CS1=0$ ,  $CS2=1$ . Транзисторы VT1 и VT2 выключены, VT3 включен и выполняет функцию резистора, доопределяющего сигнал на выходе Q до гарантированного уровня лог. 1. Спротивление канала открытого транзистора VT3 может составлять несколько килоом.

#### 4.6. Предотвращение гонок при опросе счетчика [78]

В схеме (рис. 4.8, а) при опросе счетчика через входной порт микроЭВМ возможно считывание недостоверного кода. Это связано с тем, что тактовый генератор G1 микроЭВМ и генератор G2 счетных импульсов не синхронизированы между собой, поэтому в момент опроса код в счетчике может меняться. Чтобы исключить возможность получения недостоверного результата, можно использовать двойной программный опрос счетчика со сравнением полученных кодов. Если они совпали или незначительно отличаются из-за задержки между опросами, то констатируется их достоверность; если отличия велики, то проводится третий опрос и т. д. Такое решение, однако, связано с потерями машинного времени.

В предлагаемой схеме (рис. 4.8, б) гарантируется получение достоверного результата опроса счетчика независимо от текущего соотношения сигналов с генераторов G1 и G2. Это достигается введением регистра-зашелки RG и фильтра, который предотвращает поступление положительного фронта сигнала CLK на вход счетчика CT (по которому к его старому содержимому прибавляется единица) в период загрузки регистра RG. Поэтому после загрузки этого регистра его содержимое (неизменное и «абсолютно достоверное») передается через входной порт микроЭВМ в процессор.

Счетчик CT и регистр RG приводятся в исходное состояние сигналом Сброс, который формируется, например, в одном из разрядов выходного порта микроЭВМ под управлением некоторой программы. После окончания этого сигнала счетчик CT начинает подсчет числа положительных полуволн сигнала CLK, а регистр RG не меняет состояния, так как  $W=0$  (запись в него запрещена). Фильтр (рис. 4.9) при  $W=0$  транслирует синхросигнал с входа на выход, выполняя функцию инвертора. Триггер на двух элементах ИЛИ—НЕ установлен в состояние  $A=0$ , которое подтверждается каждый раз, когда  $CL=0$  (рис. 4.10, а).

При обращении процессора к входному порту некоторая схема (связанная, например, с дешифратором адреса порта) формирует кратковременный сигнал

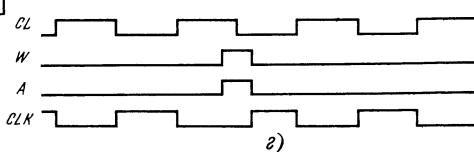
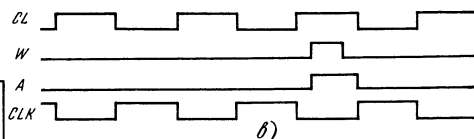
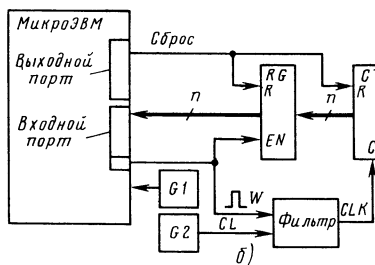
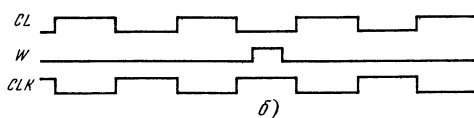
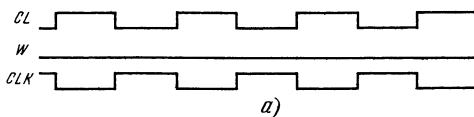
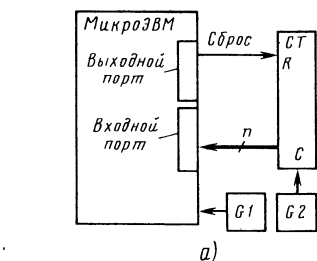


Рис. 4.8. Схема опроса счетчика СТ без защиты (а) и с защитой (б) от считывания недостоверного кода

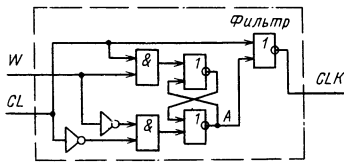


Рис. 4.9. Схема фильтра

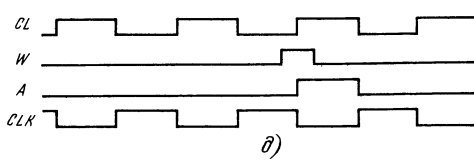


Рис. 4.10. Временные диаграммы считывания данных при разных соотношениях сигналов CL и W

$W=1$ , по окончании которого заведомо стабильные данные, поступающие во входной порт, передаются по внутренней магистрали микроЭВМ в процессор. Регистр RG при  $W=1$  транслирует данные с входов на выходы, а при переходе этого сигнала в 0 фиксирует их. Рассмотрим четыре возможные ситуации, показанные на рис. 4.10, б—д.

Если импульс  $W=1$  поступает в тот период, когда  $CL=0$  или  $CL=1$  (рис. 4.10, б, в), то последовательность сигналов CLK не нарушается, триггер фильтра остается в состоянии  $A=0$  или временно переходит в 1. К моменту окончания импульса  $W=1$  код в счетчике и регистре стабилен, поэтому данные ситуации (как и две последующие) не вызывают каких-либо осложнений при считывании.

Если сигнал  $W=1$  перекрывает отрицательный фронт импульса CL (рис. 4.10, з), то положительный фронт очередного импульса CLK смещается вправо до момента фиксации данных в регистре RG, так что прибавление единицы к старому коду в счетчике происходит уже после того, как этот код сохранен в регистре. Если сигнал  $W=1$  перекрывает положительный фронт импульса CL (рис. 4.10, д), то триггер временно устанавливается в состояние  $A=1$ , последовательность сигналов CLK не нарушается, в момент фиксации данных в регистре RG (при переходе сигнала W из 1 в 0) код в счетчике стабилен.

## 4.7. Предотвращение гонок в конвейерных схемах [79]

Конвейерные схемы состоят из ряда последовательно включенных комбинационных схем, разделенных регистрами  $RG$ , как показано на рис. 4.11. В установившемся режиме конвейер заполнен, регистры  $RG$  содержат входную, промежуточную и выходную информацию, комбинационные схемы вычисляют новые результаты, которые по фронту тактового сигнала  $CLK$  заносятся в регистры. Период тактового сигнала  $CLK$  должен быть достаточно большим для того, чтобы самая медленная комбинационная схема успела сформировать на своих выходах правильный результат с некоторым временным запасом для надежной записи результата в регистр.

Если регистров много или их разрядность велика, то исходный сигнал  $CLK$  размножают с помощью дерева из усилителей, каждая ветвь которого обслуживает один или несколько регистров. При этом из-за разброса параметров усилителей, различия характеристик линий связи и других факторов сигналы, поступающие на входы синхронизации разных регистров, несколько смещены друг относительно друга в ту или иную сторону.

В этой ситуации возникает проблема гонок, когда в конвейерной цепи из последовательно включенных регистров и комбинационных схем новая информация догоняет старую и уничтожает ее. Чем меньше задержка комбинационной схемы, тем больше опасность возникновения гонок при смещении синхроимпульсов соседних регистров. Поэтому далее рассматриваются комбинационные схемы со сквозными связями вход—выход (задержка передачи сигнала равна нулю).

Фрагмент конвейерной схемы, приведенный на рис. 4.12, а, содержит два D-триггера соседних регистров и часть комбинационной схемы с нулевой

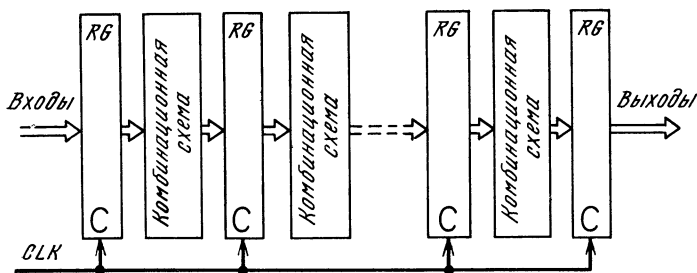


Рис. 4.11. Пример конвейерной схемы.

По фронту тактового сигнала  $CLK$  результаты обработки информации комбинационными схемами сдвигаются вправо, во входные регистры последующих каскадов

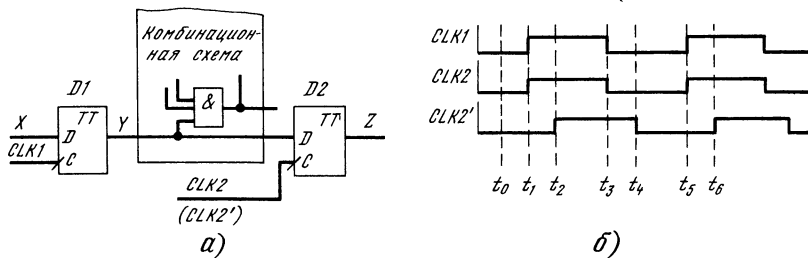


Рис. 4.12. Фрагмент конвейерной схемы, показанной на рис. 4.11 (а), и временные диаграммы сигналов синхронизации (б)

Сигналы синхронизации соседних регистров (D-триггеров) могут по различным причинам не совпадать друг с другом, что приводит к возникновению гонок

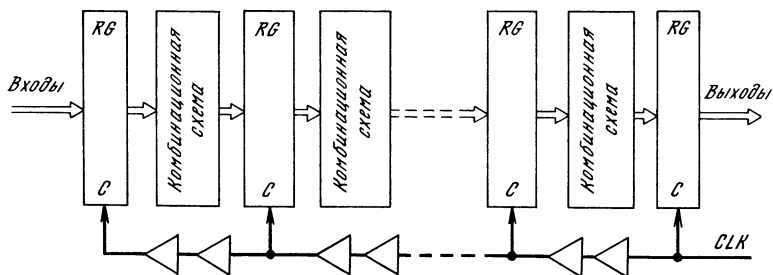


Рис. 4.13. Конвейерная схема с гарантированным отсутствием гонок.  
Последовательная цепь распространения синхросигнала ухудшает временные параметры схемы

задержкой вход—выход. D-триггер принимает новую информацию с входа D только в момент перехода сигнала синхронизации на входе C из 0 в 1. Предположим, что в момент  $t_0$  (рис. 4.12, б)  $X=1$ ,  $Y=0$ ,  $Z=1$ , сигнал X в дальнейшем не меняется. Если смещения импульсов синхронизации нет (используются сигналы CLK1 и CLK2), то в момент  $t_1$  триггеры D1 и D2 принимают новую информацию, в результате  $X=1$ ,  $Y=1$ ,  $Z=0$ , т. е. информация сдвигается вправо, как в обычном сдвиговом регистре. При очень малом смещении сигнала синхронизации триггера D2 (CLK2') схема остается работоспособной, так как триггер D1 обладает некоторой инерционностью и сигнал Y меняет состояние лишь после некоторой задержки, достаточной для восприятия старого значения сигнала триггером D2.

Однако если сигнал CLK2' смещен на время, приближающееся к задержке триггера D1 или превышающее эту задержку, то к моменту  $t_2$  сигнал Y принимает новое значение (старое потеряно), в результате в данном примере в триггер D2 заносится 1 (а не 0) и к моменту  $t_3$   $X=1$ ,  $Y=1$ ,  $Z=1$ , т. е. окончательная информация отличается от той, которая получена при правильной синхронизации.

Если сигнал CLK2' смещен в сторону, противоположную указанной на рис. 4.12, б, т. е. формируется раньше сигнала CLK1, то гонок не возникает, так как сначала считывается результат работы комбинационной схемы и лишь затем меняется информация на ее входах. Поэтому для ликвидации гонок можно использовать схему, показанную на рис. 4.13, в которой синхросигналы распространяются по цепочке усилителей-ретрансляторов навстречу сдвигаемой информации. Однако такое решение может оказаться неприемлемым из-за особенностей конструкции устройства или его логической структуры. Например, если некоторая внешняя схема собирает информацию одновременно со всех каскадов конвейера, то достоверность этой информации (как единого блока) будет гарантирована только после срабатывания всей последовательной цепи ретрансляторов синхросигнала, а это может свести на нет основное преимущество конвейерной обработки — параллелизм работы всех каскадов. Рассмотренные далее схемы не требуют последовательного стробирования регистров, как на рис. 4.13, и обеспечивают правильную работу при значительных сдвигах синхроимпульсов в обе стороны.

Для защиты от гонок в схему (рис. 4.11) вводятся дополнительные регистры, состоящие из триггеров типа «защелка». Триггер—защелка D2 (рис. 4.14, а) при  $EN=0$  передает сигнал с входа D на выход Q; при переходе EN из 0 в 1 триггер запоминает передаваемую в данный момент информацию.

Если сигналы CLK1 и CLK2' совпадают друг с другом (рис. 4.14, б), то смена информации в триггерах D1 и D3 происходит одновременно. Триггер D2 при CLK1=1 запоминает старую информацию и хранит ее до момента снятия синхроимпульса. Максимально допустимое время срабатывания комбинационной схемы определяется промежутком времени T между

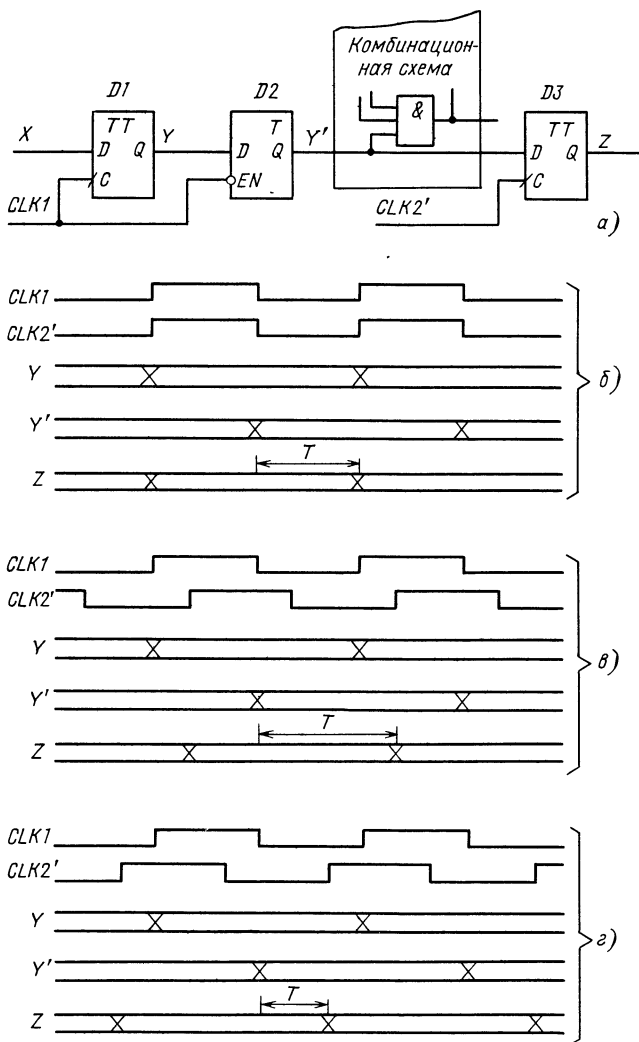


Рис. 4.14. Первый вариант конвейерной схемы (а), устойчивой к рассогласованию синхросигналов в соседних каскадах, и временные диаграммы ее работы (б—г)

изменением сигнала  $Y'$  и моментом записи результата в триггер  $D3$  (в данном примере информация передается через фрагмент комбинационной схемы без задержки).

Если сигнал  $CLK2'$  сдвинут вправо относительно сигнала  $CLK1$  (рис. 4.14, в), то временные диаграммы сигналов  $Y$  и  $Y'$  остаются такими же, как и в отсутствие сдвига, так как режим работы пары триггеров  $D1—D2$  не изменился. Моменты записи информации в триггер  $D3$  сместились вправо в соответствии со смещением сигнала  $CLK2'$ , поэтому промежуток времени  $T$ , отведенный для

работы комбинационной схемы, увеличился. При противоположном сдвиге (рис. 4.14, *з*) промежуток времени  $T$  уменьшается. Анализируя приведенные на рис. 4.14, *б—г* временные диаграммы, можно отметить, что гонки устраняются при замене «глобального» (но ненадежного) сдвига информации  $Y \rightarrow Z$  двумя последовательными «локальными» сдвигами:  $Y \rightarrow Y'$  и  $Y' \rightarrow Z$ , каждый из которых выполняется в соответствии с «надежной» временной диаграммой.

Использование схемы, приведенной на рис. 4.15, *а*, также позволяет исключить гонки, но при этом промежуток времени  $T$  (рис. 4.15, *а—г*) увеличивается примерно вдвое. В схеме применен триггер-защелка D2, который при

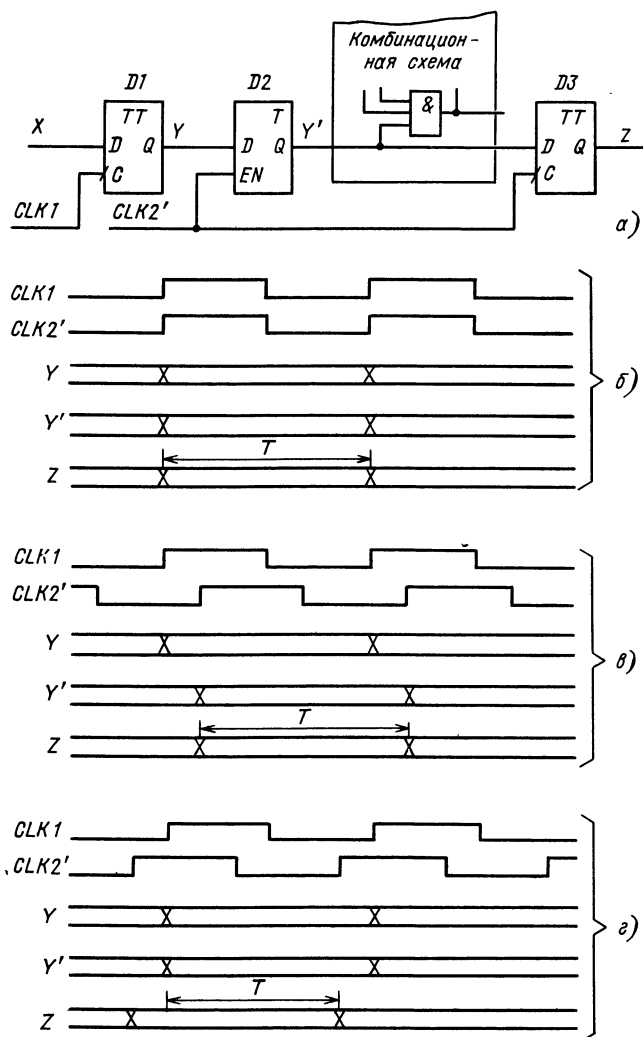


Рис. 4.15. Второй вариант конвейерной схемы (*а*), устойчивой к рассогласованию синхросигналов в соседних каскадах, и временные диаграммы ее работы (*б—г*)

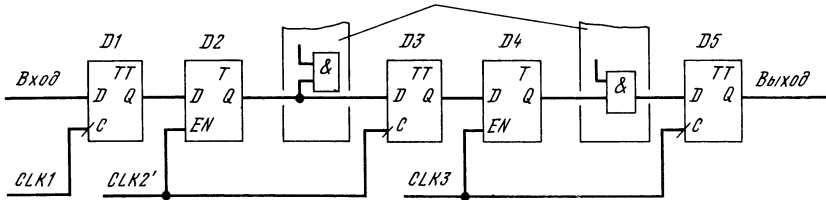
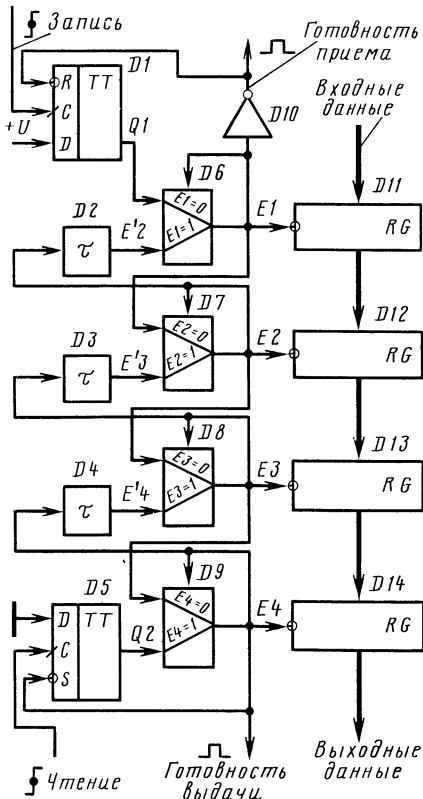


Рис. 4.16. Часть конвейерной схемы, устойчивой к рассогласованию сигналов синхронизации.

В отличие от схемы, показанной на рис. 4.13, сигналы синхронизации формируются параллельно, а не последовательно

EN=1 транслирует входную информацию на выход, а при EN=0 запоминает ее (в предыдущей схеме условия работы противоположные).

Часть конвейерной схемы, показанная на рис. 4.16, построена на основе рассмотренного каскада, защищенного от гонок. Триггер-защелка D2 устраняет гонки из-за возможного взаимного сдвига в ту или иную сторону сигналов CLK1 и CLK2'. Триггер-защелка D4 устраняет гонки из-за возможного взаимного сдвига сигналов CLK2' и CLK3.



## 4.8. Буфер типа FIFO на регистрах-защелках [80]

Буфер (рис. 4.17) по запросам записи последовательно накапливает входные данные в регистрах RG и по запросам чтения выдает эти данные в порядке их поступления. Регистр-защелка D11 (D12—D14) транслирует входные данные на выходы при нулевом сигнале E1 (E2—E4); в момент перехода этого сигнала в единицу передаваемые данные «защелкиваются» (запоминаются).

Мультиплексор D6 (D7—D9) имеет два информационных и один управляющий входы, последний объединен с выходом. Если на управляющий вход подан нулевой сигнал E1 (E2—E4), то к выходу мультиплексора подключен его верхний информационный вход; при единичном сигнале на управляющем входе выход соединен с нижним входом. Эти условия для наглядности отражены на рисунке. Триггер D1 в отсутствие

Рис. 4.17. Буфер типа FIFO на регистрах-защелках D11—D14.

Мультиплексоры D6—D9 образуют цепь сдвига сигналов управления

готовности буфера к приему данных (когда буфер полон) установлен в 0 сигналом лог. 0 на входе R и не реагирует на запросы записи (их при правильной работе системы, однако, и быть не должно). Если буфер готов к приему данных (имеется хотя бы один свободный регистр), то на входе R триггера присутствует сигнал лог. 1 и по положительному фронту сигнала Запись триггер D1 устанавливается в 1.

Триггер D5 в отсутствие готовности буфера к выдаче данных (когда буфер пуст) установлен в 1 сигналом лог. 0 на входе S и не реагирует на запросы чтения (их также не должно быть при правильной работе системы в данной ситуации). Если буфер готов к выдаче данных (имеется хотя бы один регистр, заполненный поступившими данными), то на входе S триггера D5 присутствует сигнал лог. 1 и по положительному фронту сигнала Чтение этот триггер устанавливается в 0. В статическом режиме при отсутствии запросов записи и чтения, буфер может находиться в одном из пяти устойчивых состояний (табл. 4.1.)

Если буфер пуст (см. первую строку таблицы), то все регистры «прозрачны», входные данные проходят сквозь них и поступают на выходы. Так как  $E1=0$ , сигнал Готовность приема равен единице, источник входных данных имеет возможность записать первое слово данных в буфер, точнее, в нижний регистр D14. Так как  $E4=0$ , признак Готовность выдачи, равный нулю, информирует приемник выходных данных о том, что буфер пуст и считать его содержимое не имеет смысла. Все мультиплексоры транслируют информацию на выходы с верхних информационных входов. Поэтому сигнал  $Q1=0$  проходит через всю цепочку мультиплексоров, подтверждая нулевое состояние сигналов  $E1—E4$ .

Таблица 4.1

Номер состояния	Сигналы управления регистрами				Состояние буфера
	E4	E3	E2	E1	
1	0	0	0	0	Буфер пуст
2	1	0	0	0	Занят один регистр (D14)
3	1	1	0	0	Заняты два регистра (D13, D14)
4	1	1	1	0	Заняты три регистра (D12—D14)
5	1	1	1	1	Буфер полон

Если в буфере занят один регистр (им может быть только нижний D14), то, как показано во второй строке таблицы,  $E4=1$ ,  $E3=E2=E1=0$ . Нижний мультиплексор транслирует на выход сигнал  $E4=Q2=1$ . Остальные мультиплексоры образуют последовательную цепь передачи сигнала  $Q1=0$ . Регистры D11—D13 «прозрачны», регистр D14 нечувствителен к входным данным и хранит ранее записанное в него слово. Имеется готовность буфера к приему и выдаче данных.

Если в буфере заняты два регистра (ими могут быть только D13 и D14, как показано в третьей строке таблицы), то сигнал  $Q2=1$  распространяется через два нижних мультиплексора (D9 и D8), а сигнал  $Q1=0$  — через два верхних (D6 и D7). Регистры D11 и D12 «прозрачны», а регистры D13 и D14 хранят полученные ранее данные. Имеется готовность буфера к приему и выдаче данных. Оставшиеся состояния буфера (см. строки 4 и 5 таблицы) можно рассмотреть аналогично.

Чтобы убедиться в том, что буфер не может находиться в каком-либо ином, не отраженном в таблице статическом состоянии (их число составляет  $16-5=11$ ), достаточно показать неустойчивость ситуации, при которой  $E_i=1$ , а  $E_{i+1}=0$  ( $i=1, 2, 3$ ) — такие ситуации встречаются в каждом из оставшихся

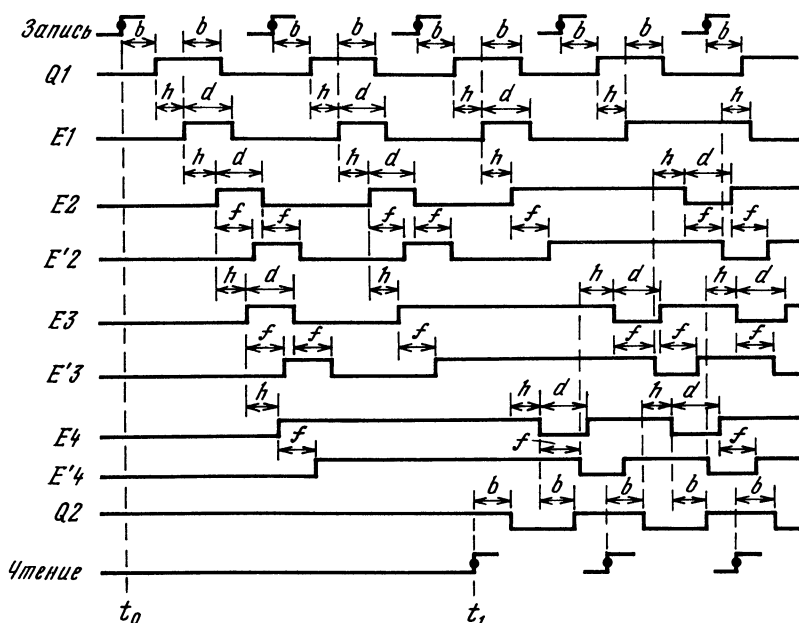


Рис. 4.18. Временные диаграммы работы буфера при записи и считывании данных.  
 $b$ —задержка D-триггера;  $d$  и  $h$ —задержки реакции мультиплексора на управляющий и информационный сигналы на входах;  $f$ —задержка элемента D2 (D3, D4)

одиннадцати состояний. Действительно, предположим, например, что существует некоторое устойчивое состояние буфера, при котором  $E2=1$ , а  $E3=0$ . Если это так, то выход мультиплексора D7 соединен с его нижним информационным входом, на который подан сигнал лог. 0 (элемент задержки D3 в статике не учитываем). Поэтому сигнал E2 также должен быть равен 0, что противоречит принятому предположению.

После включения напряжения питания и окончания переходных процессов буфер может оказаться в одном из пяти рассмотренных ранее устойчивых состояний. Чтобы провести начальную установку буфера, т. е. «очистить» его от случайных данных, необходимо сформировать как минимум четыре импульса Чтение, исходя из того, что, возможно, после включения напряжения питания он перешел в устойчивое состояние, при котором  $E1=E2=E3=E4=1$ . Каждое считывание освобождает очередной регистр, в конечном счете буфер оказывается пустым.

Для правильной работы буфера в динамике необходимо, чтобы данные передавались через регистры D11—D14 быстрее, чем управляющая информация через мультиплексоры D6—D9. Другие необходимые условия проявятся при рассмотрении временных диаграмм работы буфера, приведенных на рис. 4.18.

В момент  $t_0$  формируется первый запрос записи в исходно пустой буфер. Через время  $b$ , равное задержке триггера D1, сигнал  $Q1$  переходит из 0 в 1, затем через время  $h$  сигнал  $Q1=1$  проходит на выход мультиплексора D6. Этот мультиплексор переключается на прием сигнала  $E'2=0$ , поэтому через время  $d$  сигнал  $E1$  возвращается в нулевое состояние. К этому моменту триггер D1 также вернулся в нулевое состояние под действием сигнала лог. 0 на входе R, поэтому исключается повторная установка сигнала  $E1$  в единицу. Чтобы все происходило именно так, необходимо выполнение условия

Положительный импульс  $E1=1$  распространяется по последовательной цепи из мультиплексоров D7—D9. Мультиплексоры D7 и D8 (так же как и D6) временно переключаются на передачу битов с нижних входов, но так как эти биты нулевые, возвращаются в исходные состояния, а к этому времени сигналы на их верхних информационных входах уже успевают вернуться в состояние лог. 0. Поэтому повторных переключений мультиплексоров нет.

Импульсный сигнал  $E3=1$  переключает мультиплексор D9 на прием сигнала с нижнего информационного входа. В отличие от ранее рассмотренных аналогичных ситуаций, на этом входе вместо сигнала лог. 0 присутствует сигнал лог. 1 ( $Q2=1$ ), который поступает на выход мультиплексора, подтверждая ранее сформированный сигнал  $E4=1$ . В результате мультиплексор D9 не возвращается в исходное состояние и запоминает сигнал  $E4=1$ , выполняя роль триггера. Таким образом, буфер переходит из свободного состояния



5)

в частично заполненное, первое слово данных хранится в регистре D14 и может быть считано; остальные регистры свободны.

Последующие операции записи протекают аналогично, с поправкой на то, что происходит накопление статических единичных сигналов управления регистрами, как было показано в таблице. Буфер постепенно заполняется данными, и после четвертой записи сигнал E1 переходит в устойчивое единичное состояние, которое, однако, вскоре меняется в связи со считыванием данных из буфера.

В момент  $t_1$  приемник фиксирует в своем входном регистре выходные данные из буфера, затем начинается сдвиг данных в регистрах D11—D14. Сигналы Q2 и E4 последовательно устанавливаются в 0, мультиплексор D9 переключается на прием единичного бита с верхнего входа, сигнал E4 возвращается в исходное состояние и т. д. В результате формируются не перекрывающиеся во времени отрицательные импульсы E4—E2, данные последовательно сдвигаются на один каскад вниз. Регистр D11 освобождается, на его управляющем входе формируется статический сигнал E1=0. После второго чтения освобождаются два верхних регистра и т. д.

Время окончания переходных процессов в схеме управления при чтении уменьшается с уменьшением числа занятых регистров, так как сокращается путь распространения отрицательного импульса Q2 по цепи из мультиплексоров (при записи ситуация обратная—с уменьшением числа занятых регистров путь распространения положительного импульса Q1 увеличивается).

Чтобы иметь более полную информацию об уровне заполнения буфера, можно следить за сигналом  $E_i$ , соответствующим, например, половине его объема (рис. 4.19, а). Для устранения кратковременных положительных и отрицательных импульсов, сопровождающих операции записи и чтения (см. рис. 4.18), можно использовать фильтр, содержащий элемент задержки  $\Delta t$ , элемент Исключающее ИЛИ (M2) и D-триггер. Из временных диаграмм (рис. 4.19, б) следует, что входные импульсы  $E_i$ , длительность которых меньше  $\Delta t$ , не проходят на выход фильтра, так что сигнал  $G_i$  в полной мере отражает состояние регистра выбранного уровня (занят—свободен).

## 4.9. Быстродействующий буфер типа LIFO [81]

Буфер (рис. 4.20) выполнен по схеме, близкой к приведенной в [81], и содержит накопитель данных на регистрах RG0—RG7 и блок управления (регистр RG8, элементы И, И—НЕ, инверторы). После воздействия сигнала Уст начальной установки в регистре RG8 блока управления присутствует код 0...01, буфер готов к получению первого слова данных. Это слово поступает от внешнего источника данных по шине D<sub>1</sub>, после чего он формирует импульсный сигнал записи  $Z_p=1$ . В результате на входе синхронизации регистра RG0 формируется положительный фронт сигнала, по которому входное слово данных запоминается в регистре.

По окончании сигнала  $Z_p=1$  код в регистре RG8 сдвигается на один разряд влево ( $Q_1=1$ ,  $Q_0=Q_2=\dots=Q_8=0$ ). Поэтому следующее входное слово данных принимается в регистр RG1, код в регистре RG8 вновь сдвигается на один разряд влево и т. д. После приема данных в регистр RG7 буфер заполнен, о чем свидетельствует сигнал  $Q_8=1$ . Внешний источник данных, получив этот сигнал, прекращает пополнение буфера до момента появления в нем хотя бы одного незанятого регистра (когда  $Q_8=0$ ). Если при  $Q_8=1$  сигнал  $Z_p=1$  все же сформирован, то содержимое накопителя данных остается прежним, сдвиг кода в регистре RG8 запрещен сигналом  $H=0$ , к источнику данных поступает сигнал ошибки  $Ош1=1$ .

Если буфер заполнен, то сигнал  $Q_8=1$  разрешает выдачу данных из регистра RG7 в шину D<sub>0</sub> при получении сигнала чтения  $Ч_t=1$  от внешнего приемника данных. Выходы регистров выполнены по схеме с «третьим состоянием», сигнал  $OE=1$  вызывает подключение регистра к шине D<sub>0</sub>. По окончании сигнала  $Ч_t=1$  код в регистре RG8 сдвигается на один разряд

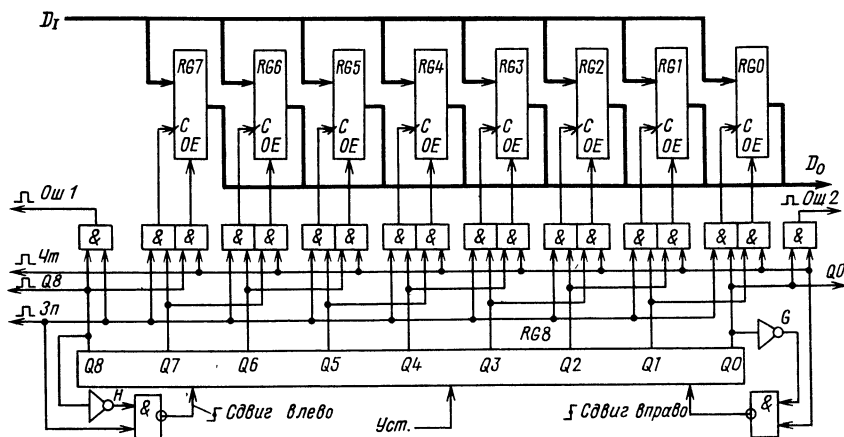


Рис. 4.20. Буфер типа LIFO

вправо ( $Q_7=1$ ,  $Q_8=Q_6=\dots=Q_0=0$ ), так что при следующем чтении данные выдаются в шину  $D_0$  из регистра  $RG_6$  и т. д.

После выдачи данных в шину  $D_0$  из регистра  $RG_0$  сигнал лог. 1 в регистре  $RG_8$  перемещается в его крайний правый разряд. Условие  $Q_0=1$  означает, что буфер пуст, поэтому внешний приемник данных должен прекратить выполнение циклов считывания; в противном случае в ответ на каждое считывание он получает сигнал ошибки  $Ош2=1$ . Так как  $G=0$ , сдвиг кода вправо в регистре  $RG_8$  запрещен. Циклы записи и считывания могут выполняться в произвольном порядке, но не одновременно.

#### 4.10. Упрощение схемы сравнения кодов заменой элементов Искключающее ИЛИ элементами ИЛИ—НЕ [82]

При построении таймеров, формирователей импульсов заданной длительности и других устройств используют схемы сравнения заданного кода с меняющимся, снимаемым с выходов счетчика. В традиционной схеме (рис. 4.21, а) триггер ТТ исходно установлен в 0 отрицательным импульсом  $R$ ; по сигналу загрузки  $LD=0$  заданный (эталонный) код  $D$  поступает в регистр-защелку  $RG$ , счетчик  $CT$  устанавливается в 0. В момент перехода сигнала  $LD$  из 0 в 1 эталонный код фиксируется в регистре  $RG$ , а триггер устанавливается в 1 (если эталонный код ненулевой).

На вход синхронизации счетчика подается последовательность импульсов  $F$ , на его выходах формируется монотонно возрастающий ряд чисел, которые с помощью элементов Искключающее ИЛИ ( $M_2$ ) поразрядно сравниваются с эталоном. При совпадении числа с эталоном триггер устанавливается в 0 сигналом лог. 0 с выхода элемента ИЛИ. Таким образом, эталон определяет длительность положительного импульса  $EQ$  на выходе схемы.

Предлагается заменить элементы Искключающее ИЛИ более простыми элементами ИЛИ—НЕ (рис. 4.21, б): первые содержат по 12 МДП-транзисторов, вторые — по 4. Эталон загружается в регистр  $RG$  в инверсном коде (или в прямом, когда используются инверсные выходы триггеров этого регистра), в остальном схема работает так же, как и традиционная. Чтобы убедиться в возможности такой замены, рассмотрим пример.

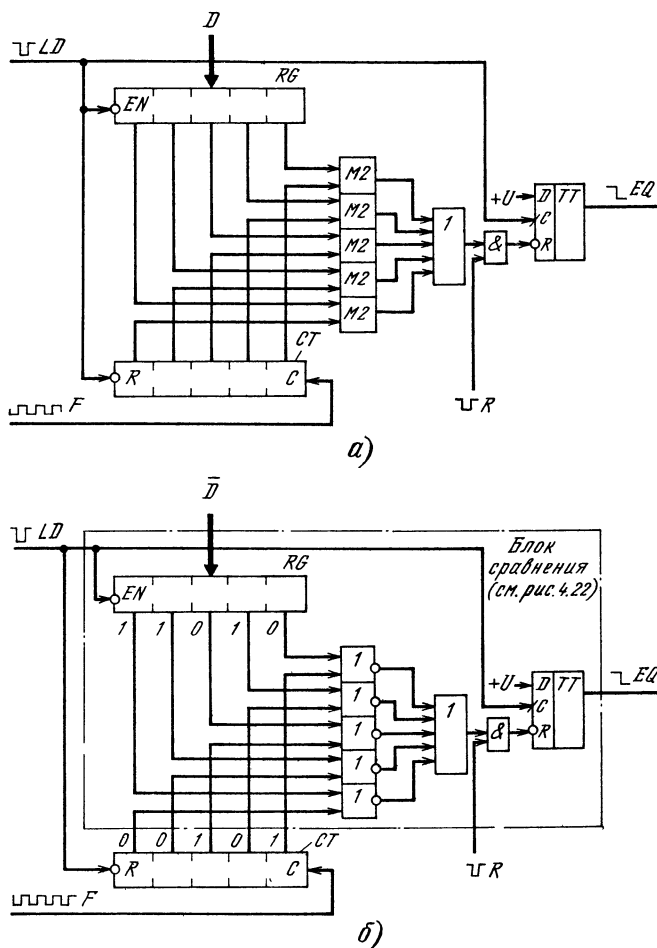


Рис. 4.21. Традиционная (а) и предлагаемая (б) схемы сравнения. Блок, выделенный штриховыми линиями, использован в схеме на рис. 4.22

Пусть  $D=00101$ , тогда по окончании импульса  $LD=0$  в регистре  $RG$  фиксируется код  $\bar{D}=11010$ ,  $EQ=1$ . Так как в начальном состоянии счетчик установлен в 0, то на входах элемента ИЛИ исходно присутствуют три нулевых и два единичных сигнала. Нулевые сигналы в дальнейшем сохраняются независимо от кода в счетчике, а единичные впервые одновременно гасятся в тот момент, когда код в счетчике достигает заданного значения 00101. При этом сигнал лог. 0 с выхода элемента ИЛИ проходит через элемент И и устанавливает триггер в 0 ( $EQ=0$ ), что и требовалось. При последующей работе счетчика будут наблюдаться повторные срабатывания элемента ИЛИ (подтверждающие нулевое состояние триггера), когда с эталоном будут сравниваться следующие коды: 00111, 01101, 01111, 10101, 10111, 11101 и 11111. Эти события, однако, несущественны — задача решена в момент регистрации первого срабатывания элемента ИЛИ.

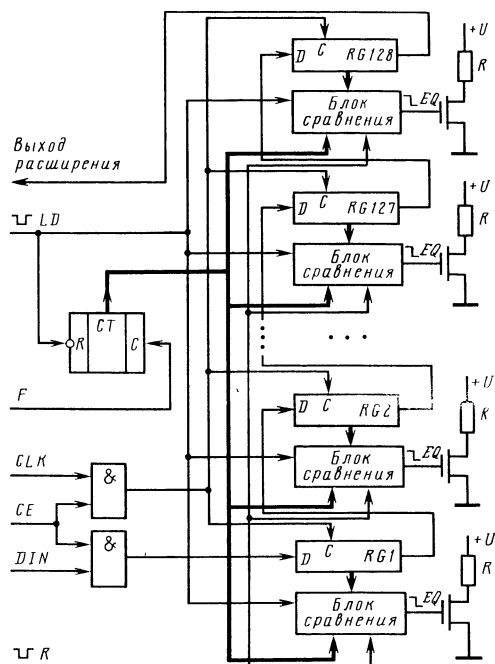


Рис. 4.22. Схема управления блоком нагревательных элементов  $R$  термопечатающего устройства.  
С помощью блоков сравнения (рис. 4.21) дозируется энергия нагрева каждого из 128 элементов  $R$

Возвращаясь к традиционной схеме (рис. 4.21, *а*), отмечаем, что она не дает повторных срабатываний элемента ИЛИ, так как поразрядное суммирование по модулю два — более сложная операция по сравнению с простым объединением сигналов по схеме ИЛИ — НЕ, которое, однако, в данной ситуации приемлемо.

Часть схемы, выделенная на рис. 4.21, *б* штриховой линией, использована в БИС управления нагревательными элементами  $R$  (рис. 4.22) термопечатающего устройства. Каждый элемент в зависимости от его температуры вызывает большее или меньшее потемнение контактирующего с ним элемента изображения на термочувствительной бумаге. Схема приводится в исходное состояние отрицательным импульсом  $R$ , все триггеры (рис. 4.21, *б*) устанавливаются в 0. При поступлении сигнала выбора кристалла  $CE=1$  входные данные  $DIN$ , сопровождаемые синхроимпульсами  $CLK$ , последовательно загружаются в единый сдвиговый регистр  $RG1—RG128$ . После этого сигналом  $LD=0$  устанавливается в 0 счетчик  $CT$  (общий для всех блоков сравнения), данные, ранее загруженные в регистры  $RG_i$ , поступают в блоки сравнения и определяют необходимую степень потемнения бумаги под соответствующим нагревательным элементом  $R$ .

При переходе сигнала  $LD$  из 0 в 1 блоки сравнения фиксируют в своих регистрах  $RG$  входные данные, триггеры  $TT$  устанавливаются в 1 (если код в регистре не равен 11111), начинается разогрев элементов  $R$ . На вход синхронизации счетчика поступают импульсы  $F$ , счетчик формирует монотонно возрастающую последовательность чисел. По мере работы счетчика в соответствии с заданными кодами триггеры всех блоков сравнения устанавливаются

в 0, в результате все элементы R выключаются в заранее заданные моменты времени. На бумаге проявляется линейка из 128 полутоновых точек, бумага протягивается на одну элементарную строку.

После перехода сигнала LD из 0 в 1 сдвиговый регистр RG1—RG128 освобождается, и, пока отрабатывается предыдущее задание (печатается строка из 128 точек), этот регистр последовательно заполняется новыми данными для печати следующей строки. Схема управления и блок нагревательных элементов могут наращиваться по мере необходимости. Для этого предусмотрен выход расширения регистра RG1—RG128.

## 4.11. Компаратор на основе ПЛМ [83]

Проверку условия  $X > C$ , где  $X$  и  $C$ — $n$ -разрядные двоичные числа  $x_{n-1}x_{n-2}\dots x_0$  и  $c_{n-1}c_{n-2}\dots c_0$  ( $x_0, c_0$ —младшие разряды), можно осуществлять с помощью стандартных компараторов. Однако, если  $X$ —переменная, а  $C$ —константа, то такое решение неэффективно, так как группа входов, соответствующих константе, соединена с шинами лог. 0 и лог. 1 в соответствии со значениями разрядов  $c_i$  и, таким образом, не несет должной информационной нагрузки; в результате снижается степень интеграции устройства.

Можно исключить входы задания константы  $C$ , если, например, использовать в качестве компаратора ПЗУ (ППЗУ) с соответствующей кодировкой. На его адресные входы подается переменная  $X$ , с выхода снимается сигнал  $Q=1$  при  $X > C$  или  $Q=0$  при  $X \leq C$ . Однако эффективность такого решения уменьшается с увеличением  $n$ , так как требуемый объем накопительной матрицы ПЗУ равен  $2^n$  бит.

Более приемлемым является использование в качестве компаратора программируемой логической матрицы (ПЛМ). На входы ПЛМ подаются сигналы  $x_{n-1}, x_{n-2}, \dots, x_0$  (параллельный код); с выхода снимается сигнал  $Q$  (равный 1 только при  $X > C$ ). Константа  $C$  определяет кодировку ПЛМ. Исходные данные для кодировки будем, как обычно, представлять в терминах булевой алгебры.

Условие  $X > C$  при  $n=3$  можно задать следующим образом:

$$Q = x_2 \overline{c_2} + x_1 \overline{c_1} (\overline{x_2 \oplus c_2}) + x_0 \overline{c_0} (\overline{x_2 \oplus c_2}) (\overline{x_1 \oplus c_1}). \quad (4.1)$$

Запись (4.1) можно трактовать так. Число  $X$  больше числа  $C$ , если выполняется одно из трех условий: 1)  $x_2 \overline{c_2} = 1$ , т. е. старший разряд числа  $X$  равен 1, а старший разряд числа  $C$ —нулю; 2)  $x_1 \overline{c_1} (\overline{x_2 \oplus c_2}) = 1$ , т. е. старшие разряды равны (их сумма по модулю два с инверсией равна 1), при этом  $x_1 = 1$ , а  $c_1 = 0$ ; 3)  $x_0 \overline{c_0} (\overline{x_2 \oplus c_2}) (\overline{x_1 \oplus c_1}) = 1$ , т. е. два старших разряда кодов  $X$  и  $C$  совпадают,  $x_0 = 1$ ,  $c_0 = 0$ .

Пусть далее  $C = 4_{10} = 100_2$ . Тогда входящие в формулу (4.1) произведения примут следующий вид:

$$\begin{aligned} x_2 \overline{c_2} &= x_2 \overline{1} = x_2 0 = 0; \\ x_1 \overline{c_1} (\overline{x_2 \oplus c_2}) &= x_1 \overline{1} (\overline{x_2 \oplus 1}) = x_1 \dot{x}_2; \\ x_0 \overline{c_0} (\overline{x_2 \oplus c_2}) (\overline{x_1 \oplus c_1}) &= x_0 \overline{1} (\overline{x_2 \oplus 1}) (\overline{x_1 \oplus 0}) = x_0 x_2 \overline{x_1}. \end{aligned}$$

В результате

$$Q = x_1 x_2 + x_0 x_2 \overline{x_1}. \quad (4.2)$$

Из формулы (4.2) следует, что  $Q=1$  при  $x_1 x_2 = 1$ , т. е. когда  $X=6$ ,  $X=7$  или при  $x_0 x_2 \overline{x_1} = 1$ , что соответствует  $X=5$ . Таким образом,  $Q=1$  при  $X > C=4$ , что и требовалось.

Условие (4.1) можно распространить на любое  $n$ :

$$Q = x_{n-1} \overline{c_{n-1}} + x_{n-2} \overline{c_{n-2}} (\overline{x_{n-1} \oplus c_{n-1}}) + x_{n-3} \overline{c_{n-3}} (\overline{x_{n-1} \oplus c_{n-1}}) (\overline{x_{n-2} \oplus c_{n-2}}) + \dots \\ \dots + x_0 c_0 (\overline{x_{n-1} \oplus c_{n-1}}) (\overline{x_{n-2} \oplus c_{n-2}}) \dots (\overline{x_1 \oplus c_1}). \quad (4.3)$$

Из формулы (4.3) следует, что при  $c_i = 1$  ( $i = 0, 1, 2, \dots, n-1$ ) соответствующее булево произведение, в которое в качестве множителя входит  $\overline{c_i}$ , обращается в нуль. Таким образом, при заданном числе  $C$  из правой части формулы (4.3) удаляются все произведения, соответствующие единичным разрядам его двоичного представления  $c_{n-1} c_{n-2} \dots c_0$ . В частности, при  $c_{n-1} c_{n-2} \dots c_0 = 11 \dots 1$  из правой части формулы (4.3) удаляются все произведения и переменная  $Q$  становится тождественно равной 0 (условие  $X > C$  в данном случае невыполнимо, так как  $C$  — максимально представимое  $n$  разрядами число). В противоположность этому при  $C = 0$  формула (4.3) после ряда упрощений приводится к виду

$$Q = x_{n-1} + x_{n-2} + x_{n-3} + \dots + x_0. \quad (4.4)$$

Из формулы (4.4) следует, что  $Q = 0$  только при  $x_{n-1} = x_{n-2} = \dots = x_0 = 0$ , в остальных ситуациях  $Q = 1$ , т. е.  $X > C = 0$ .

Таким образом, число логических произведений в кодировке ПЛМ равно числу нулей в двоичном числе  $c_{n-1} c_{n-2} \dots c_0$ . В приведенной ранее формуле (4.2) число логических произведений равно двум, по числу нулей в двоичном коде 100.

Если число нулей в двоичном коде  $c_{n-1} c_{n-2} \dots c_0$  превышает число единиц, то для экономии числа логических произведений (что прямо связано с экономией аппаратуры) следует дважды проинвертировать условия задачи следующим образом. Выполнение условия  $X > C$  равносильно невыполнению (отрицанию) противоположного условия  $X \leq C$ :

$(X > C) = \overline{X \leq C}$  (если «больше», то не может быть «меньше или равно»).

То же в более подробном виде:

$$(X > C) = \overline{(X = C) + (X < C)} = \overline{E + P}. \quad (4.5)$$

Условие  $E$  равенства чисел  $X$  и  $C$  можно записать так:

$$E = (\overline{x_{n-1} \oplus c_{n-1}}) (\overline{x_{n-2} \oplus c_{n-2}}) \dots (\overline{x_0 \oplus c_0}). \quad (4.6)$$

При  $c_i = 0$   $\overline{x_i \oplus c_i} = \overline{x_i}$ , при  $c_i = 1$   $\overline{x_i \oplus c_i} = x_i$ .

Условие  $P$ , соответствующее соотношению  $X < C$ , по аналогии с условием  $Q$  (формула (4.3)) можно записать так:

$$P = \overline{x_{n-1} c_{n-1}} + \overline{x_{n-2} c_{n-2}} (\overline{x_{n-1} \oplus c_{n-1}}) + \overline{x_{n-3} c_{n-3}} (\overline{x_{n-1} \oplus c_{n-1}}) (\overline{x_{n-2} \oplus c_{n-2}}) + \dots \\ \dots + \overline{x_0 c_0} (\overline{x_{n-1} \oplus c_{n-1}}) (\overline{x_{n-2} \oplus c_{n-2}}) \dots (\overline{x_1 \oplus c_1}). \quad (4.7)$$

В правой части формулы (4.7) уничтожаются произведения, соответствующие нулевым значениям разрядов  $c_i$  в слове  $c_{n-1} c_{n-2} \dots c_0$ .

Таким образом, при построении кодировки ПЛМ, выполняющей функцию проверки условия  $X > C$ , где  $C$  — некоторая константа, необходимо представить константу в двоичной записи и подсчитать число содержащихся в этой записи нулей. Если число нулей оказалось меньшим или равным числу единиц, то для получения кодировки ПЛМ используется формула (4.3). При этом число логических произведений равно числу нулей в двоичном коде константы  $C$ . Если число нулей оказалось большим числа единиц, то для получения кодировки применяется формула (4.5) и ее составные части (4.6) и (4.7). Число логических произведений на единицу превышает число единиц, имеющих в двоичном коде константы  $C$ .

Пусть  $n=20$ ,  $C=50\,000_{10}$ . Представив число 50 000 в двоичной записи, получим

Номер разряда (i)	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Значение $c_i$	0	0	0	0	1	1	0	0	0	0	1	1	0	1	0	1	0	0	0	0

Из полученной записи видно, что число нулей равно 14, а число единиц 6. Поэтому для построения кодировки ПЛМ пользуемся формулами (4.5)—(4.7):

$$E = \overline{x_{19}x_{18}x_{17}x_{16}x_{15}x_{14}x_{13}x_{12}x_{11}x_{10}x_9x_8x_7x_6x_5x_4x_3x_2x_1x_0}.$$

$$P = \overline{x_{15}x_{19}x_{18}x_{17}x_{16} + x_{14}x_{19}x_{18}x_{17}x_{16}x_{15} + x_9x_{19}x_{18}x_{17}x_{16}x_{15}x_{14}x_{13}x_{12}x_{11}} \times \\ \times \overline{x_{10} + x_8x_{19}x_{18}x_{17}x_{16}x_{15}x_{14}x_{13}x_{12}x_{11}x_{10}x_9 + x_6x_{19}x_{18}x_{17}x_{16}x_{15}x_{14}x_{13}x_{12}} \times \\ \times \overline{x_{11}x_{10}x_9x_8x_7 + x_4x_{19}x_{18}x_{17}x_{16}x_{15}x_{14}x_{13}x_{12}x_{11}x_{10}x_9x_8x_7x_6x_5}.$$

## 4.12. Схемы параллельного сравнения кода с несколькими эталонами

Задачу одновременного быстрого сравнения параллельного кода  $X$  с эталонными кодами  $H_1, H_2, \dots, H_n$  можно решить с помощью  $n$  компараторов, каждый из которых сравнивает код  $X$  с одним из эталонных. Выходы компараторов объединяются элементом ИЛИ; при обнаружении совпадения кода  $X$  с одним из эталонных на выходе элемента ИЛИ формируется сигнал лог. 1. Подобные схемы обычно используются в системах отладки и диагностики для регистрации заранее заданных кодовых комбинаций в исследуемом устройстве.

Более экономичное решение, не лишенное, однако, недостатков, приведено в [84] (рис. 4.23). Схема сравнения кодов выполнена на основе трех статических ОЗУ и предназначена для формирования сигнала прерывания процессора при обнаружении заданных условий. Эти условия определяются содержимым регистров  $RG$ , в которых временно запоминается 16-разрядный код  $A0—A15$  адреса и сигналы  $C0—C7$  управления: чтение—запись, команда—операнд, режим операционной системы—режим пользовательской программы и т. п.

Схема позволяет посылать сигнал прерывания в процессор, например, при условии, что выполнялась пользовательская программа и был считан операнд из ячейки ПЗУ (на рисунке не показано) с адресом  $07FA_{16}$ . Если данные в магистрали передаются по тем же линиям, что и адрес, то схему можно настроить на поиск ситуации, когда передается один из заранее ожидаемых кодов данных, и т. п.

Сигналы, характеризующие текущую кодовую ситуацию, с выходов регистров  $RG$  по-

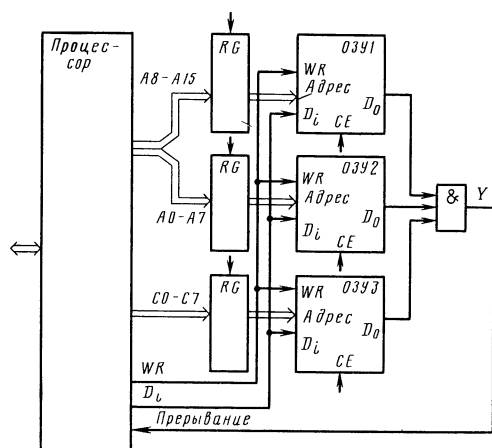


Рис. 4.23. Схема слежения за состоянием сигналов  $A0—A15$ ,  $C0—C7$ . При обнаружении заданных сочетаний сигналов вырабатывается признак  $Y=1$

ступают на адресные входы ОЗУ1—ОЗУ3. В режиме сравнения из каждого ОЗУ ( $256 \times 1$  бит) в соответствии с адресным кодом считывается содержимое ячейки—0 или 1. Во все ОЗУ предварительно записана некоторая информация с таким расчетом, что при возникновении заданной ситуации (т. е. при совпадении входного кода с одним из эталонных) из всех ОЗУ одновременно считываются лог. 1. Например, если необходимо выявить две ситуации, когда код A15 A14...A8, A7 A6...A0, C7 C6...C0 совпадает с кодом 07, 2E, FF<sub>16</sub> или 07, 58, FF<sub>16</sub>, то в ОЗУ1 предварительно записывается лог. 1 по адресу 07 (в остальных ячейках нули); в ОЗУ2 лог. 1 записывается в ячейки с адресами 2E и 58 (в остальных ячейках нули); в ОЗУ3—в ячейку с адресом FF (в остальных ячейках нули).

При появлении на входах ОЗУ1—ОЗУ3 любого из двух указанных 24-разрядных кодов срабатывает элемент И и в процессор поступает лог. 1—сигнал прерывания. Чтобы узнать, какая из двух ожидавшихся ситуаций реализовалась, процессор, перейдя к прерывающей программе, считывает содержимое регистров RG, которые выполнены программно-доступными. (Цепи записи и считывания регистров и схемы управления ОЗУ на рисунке не показаны. Отметим, что информация в регистрах RG не должна меняться после формирования сигнала прерывания, чтобы процессор мог ее проанализировать.)

В рассмотренном примере оба 24-разрядных эталонных кода совпадают в двух байтах, отличия имеются только в одном байте. Это позволило избежать ложных сравнений, возникающих при использовании эталонных кодов, отличающихся друг от друга в двух или трех байтах, как показано далее на примере.

Предположим, что необходимо выявить в проходящем потоке два кода. Первый, как и в предыдущем примере, равен 07, 2E, FF, второй—88, 01, 23. В ОЗУ1—ОЗУ3 использованы по две единичные ячейки—07; 88 (ОЗУ1), 2E; 01 (ОЗУ2), FF; 23 (ОЗУ3), в остальных ячейках записаны нули. Это обеспечивает выдачу лог. 1 из всех ОЗУ и срабатывание элемента И при поступлении любого из двух указанных 24-разрядных кодов. Однако из-за «разобщенности» сравниваемых частей полного 24-разрядного кода и их независимой обработки микросхемами ОЗУ1—ОЗУ3 возникают ложные совпадения при поступлении на входы следующих шести кодов: 07, 2E, 23; 07, 01, FF; 07, 01, 23; 88, 01, FF; 88, 2E, 23; 88, 2E, FF.

Чтобы отличить истинные совпадения от ложных, процессор, выполняя прерывающую программу, считывает содержимое регистров RG и анализирует полученный 24-разрядный код, последовательно сопоставляя его с эталонными кодами, хранимыми в «обычном» ОЗУ (на рисунке не показано). Ложные совпадения не учитываются процессором, и поиск продолжается.

При увеличении числа эталонных кодов, отличающихся друг от друга в разных байтах, быстро растет доля ложных совпадений и, следовательно, быстро растут потери времени на их выявление программным путем. Поэтому рассмотренное в [84] решение (рис. 4.23) следовало бы улучшить (что мы и попытаемся сейчас сделать), чтобы уменьшить число ложных совпадений, в идеальном случае вовсе исключив возможность их появления.

На рис. 4.24 приведен улучшенный вариант схемы параллельного сравнения заданного кода с множеством эталонных. Для уменьшения доли ложных совпадений используются дополнительные ОЗУ (ОЗУ3 и ОЗУ4). Число входов схемы в данном примере равно 16, число эталонных кодов (A15 A14...A8, A7 A6...A0) равно восьми: H1=07, AF; H2=0A, 12; H3=12,88; H4=14,30; H5=1F, B4; H6=5A, 9C; H7=60,07; H8=FF, AB. Старшие байты этих кодов соответствуют адресам ячеек ОЗУ1, в которых записаны лог. 1 (в остальных ячейках ОЗУ1—нули). Аналогично младшие байты эталонных кодов соответствуют единичным ячейкам ОЗУ2. Существенно, что адресные входы ОЗУ3 и ОЗУ4 соединены с линиями как старшего, так и младшего байтов.

Благодаря такому использованию адресных входов ОЗУ3 и ОЗУ4 коды H1—H8 преобразуются в коды H1'—H8' (A15 A14 A13 A12 A7 A6 A5

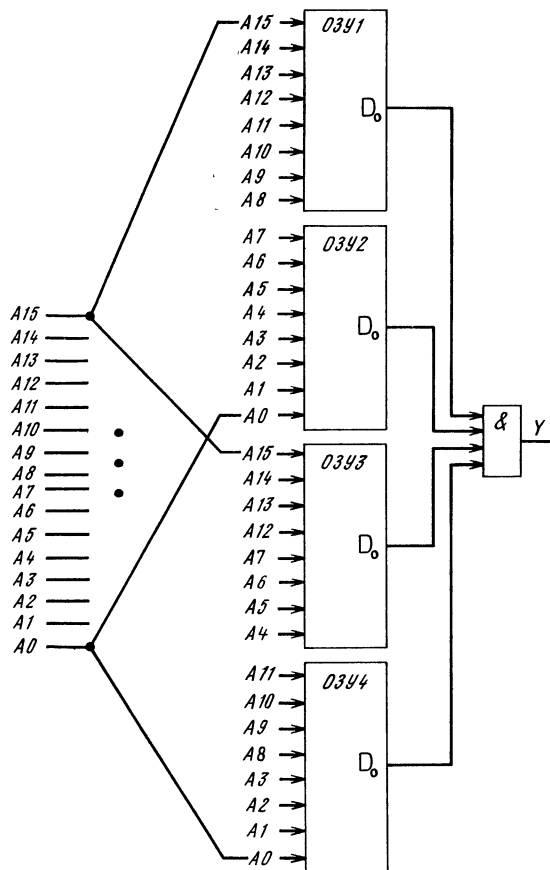


Рис. 4.24. Избыточная схема сравнения кода  $A_0 \text{—} A_{15}$  с эталонными кодами.  
Уменьшено число ложных совпадений

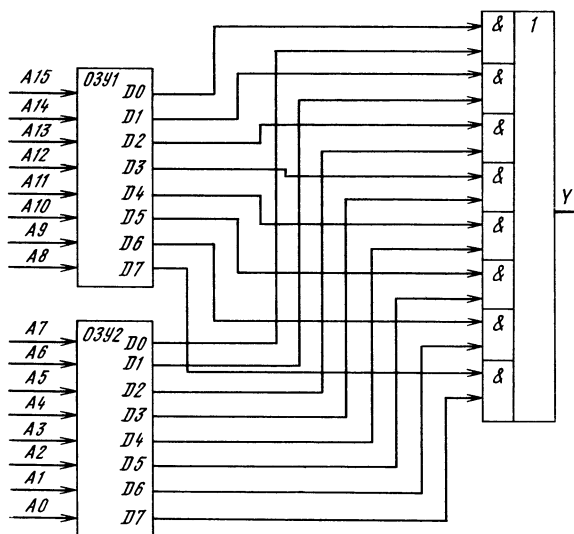


Рис. 4.25. Схема сравнения кода A0—A15 с эталонными кодами.

Ложные сравнения исключены, однако число эталонных кодов не должно превышать восьми

A4, A11 A10 A9 A8 A3 A2 A1 A0) с переставленными цифрами: H1'=0A, 7F; H2'=01, A2; H3'=18,28; H4'=13,40; H5'=1B, F4; H6'=59, AC; H7'=60,07; H8'=FA, FB. В соответствующие ячейки ОЗУ3 и ОЗУ4 записываются лог. 1, в остальных ячейках хранятся лог. 0.

Из указанных восьми эталонных кодов H1—H8 образуются 56 «ложных», которые в отсутствие ОЗУ3 и ОЗУ4 вызвали бы ошибочное срабатывание выходного элемента И. К этим кодам относятся следующие: 07, 12; 07, 88; 07, 30; ...; FF, 07. Рассматривая каждый из этих кодов, можно убедиться, что после преобразования адресов он не вызывает одновременного выбора лог. 1 из ОЗУ3 и ОЗУ4. Иными словами, ложные коды нейтрализуются избыточными микросхемами памяти, а истинные приводят к одновременной выдаче лог. 1 из всех ОЗУ.

Однако и для этой схемы можно подобрать группы эталонных кодов, порождающие ее ложные срабатывания. Пусть, например, H1=1A, BC и H2=1B, AC. После перестановки цифр получим H1'=1B, AC и H2'=1A, BC. Ложные срабатывания будут наблюдаться при поступлении на вход схемы кодов 1A, AC и 1B, BC (после преобразования адресов эти коды превращаются «сами в себя»); в ответ из всех четырех ОЗУ будут выданы лог. 1. Схему можно и далее улучшать введением в нее дополнительных микросхем памяти с «перепутанными» адресными входами, однако этот путь, по-видимому, не самый удачный.

В схеме, показанной на рис. 4.25, использованы микросхемы ОЗУ со структурой 256×8 бит. Схема позволяет проводить параллельное сравнение входного кода A15 A14...A8, A7 A6...A0 с несколькими (от одного до

восьми) эталонными, причем ложные сравнения исключены. Предположим, что необходимо сформировать сигнал  $Y=1$  при совпадении входного кода с одним из эталонных: 1A, BC; 1A, BD; 1A, BE; F9, 89; FB, 89; FC, DE; FC, BC и FF, 00. Пронумеруем эти коды в произвольном порядке от нуля до семи, как показано в табл. 4.2, и пометим единицами соответствующие разряды выходных слов ОЗУ1 и ОЗУ2 так, чтобы в нужных ситуациях срабатывала схема И—ИЛИ.

Таблица 4.2

Номер кода	Код	Выходы ОЗУ1								Выходы ОЗУ2							
		D7	D6	D5	D4	D3	D2	D1	D0	D7	D6	D5	D4	D3	D2	D1	D0
0	1A, BC	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1
1	1A, BD	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	0
2	1A, BE	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0
3	F9, 89	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0
4	FB, 89	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0
5	FC, DE	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0
6	FC, BC	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	1
7	FF, 00	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Примечание. Во всех ячейках ОЗУ1 и ОЗУ2, не указанных в таблице, записаны нулевые коды.

Рассмотрим, например, код 5:FC, DE. По адресу FC из ОЗУ1 извлекается код, содержащий лог. 1 в разрядах D5 и D6. В то же время из ОЗУ2 по адресу DE считывается код, содержащий лог. 1 в разряде D5, в результате срабатывает соответствующий элемент И схемы И—ИЛИ ( $Y=1$ ), что и требовалось. Любые сочетания чужеродных байтов эталонных слов (например, 1A, 89) не вызовут ложного срабатывания схемы, так как совпадения сигналов лог. 1 ни на одном элементе И не произойдет.

Отметим, что номер элемента И, на который поступили две лог. 1, совпадает с номером эталонного кода в таблице. Поэтому если использовать восемь раздельных схем И, то с их выходов можно считывать номер кода для передачи в процессор.

Схему, показанную на рис. 4.25, нельзя считать наиболее совершенной по сравнению с предыдущими, так как число эталонных кодов определяется числом выходов ОЗУ и, следовательно, не может быть большим. Таким образом, все описанные здесь решения требуют кардинального усовершенствования.

#### 4.13. Усовершенствованные схемы генераторов кода маски [85, 86]

Коды маски используются для выделения в слове данных групп разрядов, чтобы в дальнейшем обрабатывать не все слово, а лишь отдельную его часть (части). Выделение осуществляется поразрядным логическим сложением или умножением кода маски на код данных.

Предположим, например, что 16-разрядный код маски 0001.1111.0000.0000 поразрядно логически умножается на код данных 1011.0011.0101.1111 (точки введены для удобства визуального сравнения кодов). Результат умножения 0001.0011.0000.0000 содержит пять разрядов исходного кода данных, остальные

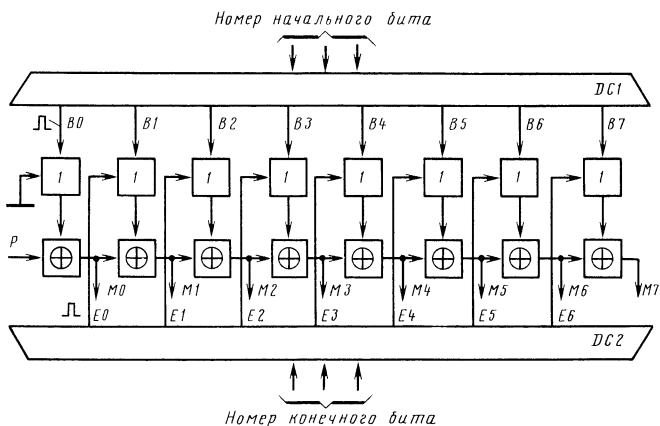


Рис. 4.26. Исходная схема генератора кодов маски—объект усовершенствования (см. рис. 4.27, 4.28)

заменены нулями. Если код маски равен 1110.0000.1111.1111 (инвертированный код из предыдущего примера), то после его поразрядного логического сложения с кодом данных (тем же, что и ранее) результирующий код 1111.0011.1111.1111 также содержит пять разрядов исходного кода данных, однако остальные разряды заменены единицами.

Если разрядность кода маски равна  $n$ , то он может принять  $2^n$  различных значений. Однако на практике обычно используют только те кодовые комбинации, которые позволяют «обрезать» слово данных с одной или двух сторон с заменой отброшенных разрядов нулями или единицами, как было показано на примерах, где слово данных «обрезалось» с обеих сторон. Для быстрого формирования таких кодов применяют различные схемные решения.

Генератор кода маски (рис. 4.26) имеет входы для указания номеров

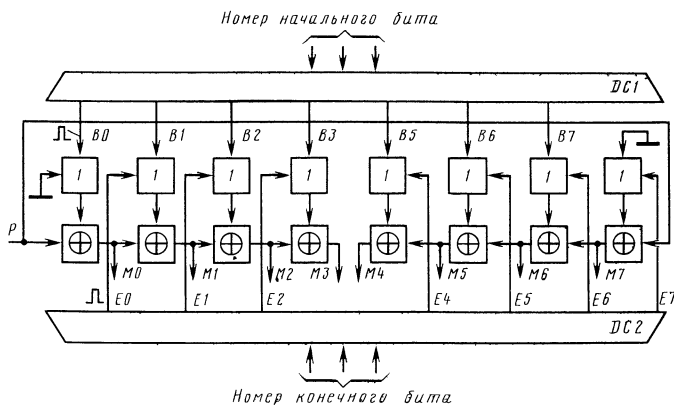
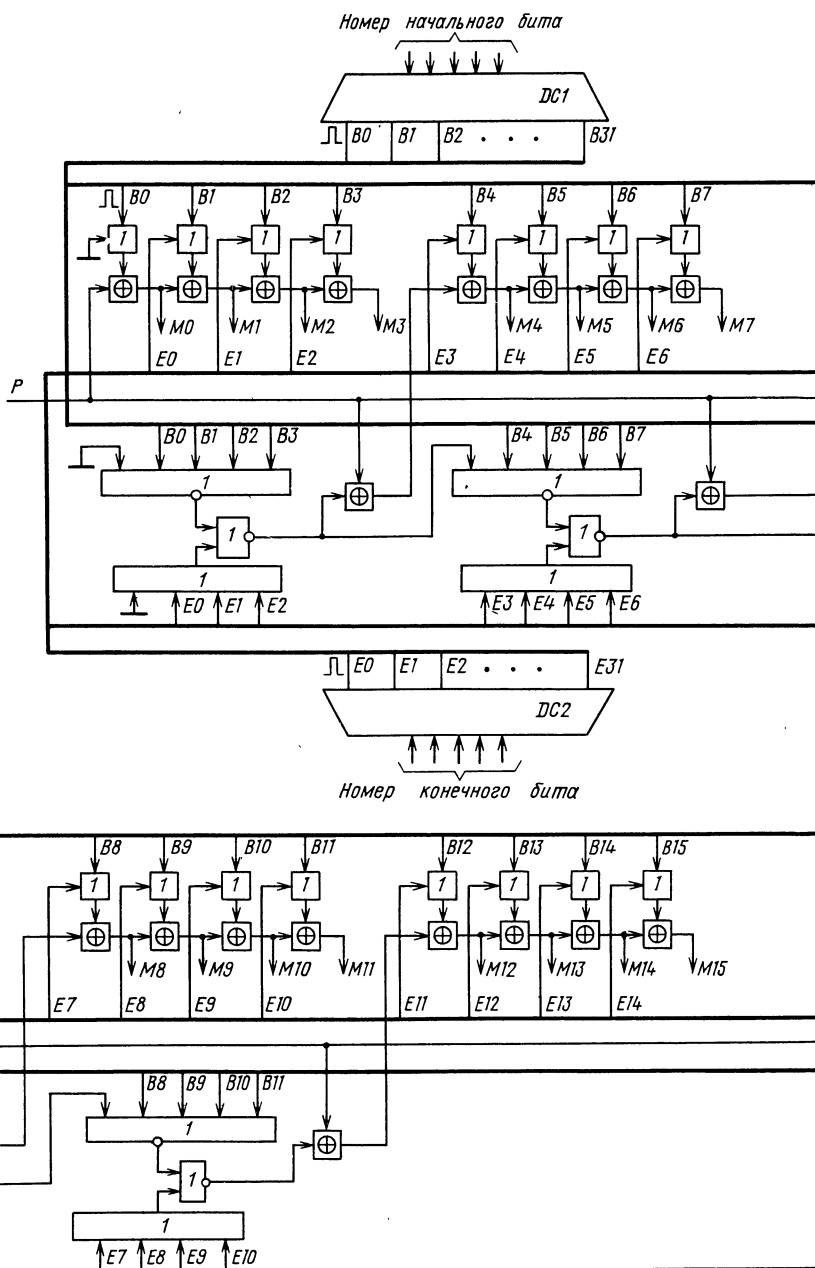


Рис. 4.27. Усовершенствованная схема генератора кодов маски (см. рис. 4.26). Последовательная цепь распространения сигнала Р заменена двумя более короткими цепями, включенными параллельно



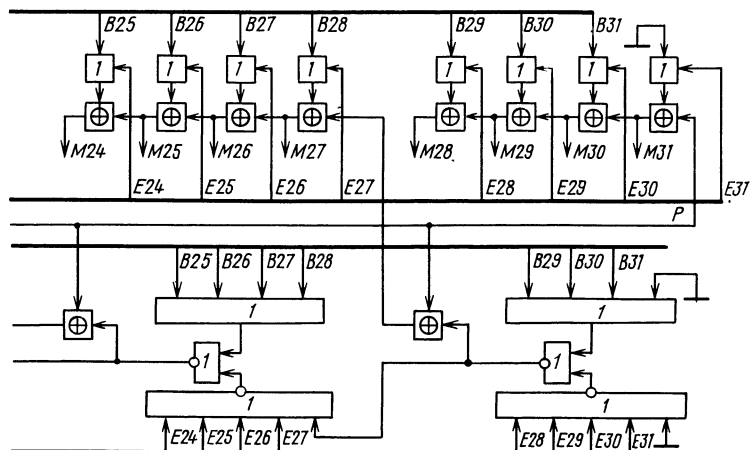
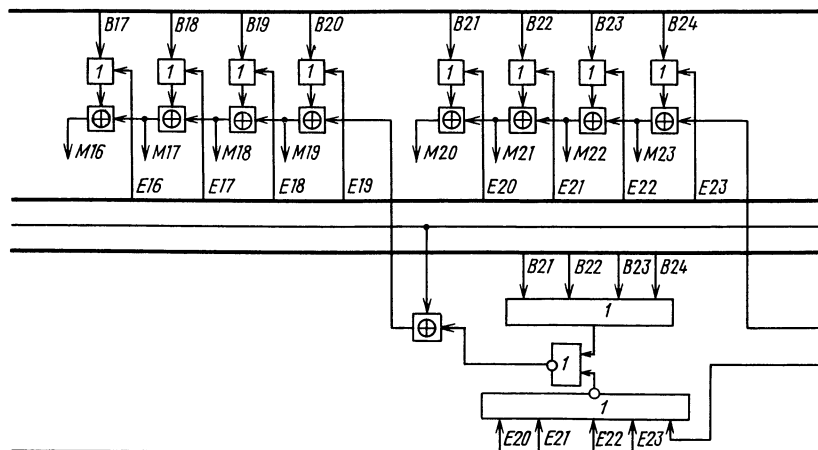


Рис. 4.28. 32-разрядный генератор кодов маски с двусторонней обработкой сигналов и ускоренным переносом (состыкуйте эти фрагменты по горизонтали в порядке возрастания номеров выходов  $M_i$ )

начального и конечного битов, соответствующих границам сохраняемой части слова данных, вход  $P$  «полярности» маски (см. приведенные ранее примеры взаимно обратных кодов маски) и выходы для передачи сигналов  $M_i$  к схемам поразрядного логического умножения или сложения с разрядами слова данных (эти схемы не показаны).

Дешифраторы DC1 и DC2 анализируют входные трехразрядные коды и формируют сигналы лог. 1 (напряжения высокого уровня) на соответствующих выходах. Эти сигналы проходят через элементы ИЛИ и воздействуют на верхние входы элементов Исключающее ИЛИ. Сигнал  $P$  полярности маски распространяется по последовательной цепи из элементов Исключающее ИЛИ и инвертируется теми из них, на верхние входы которых поданы сигналы лог. 1 (таких элементов может быть не более двух).

Пусть, например, номера начального и конечного битов выделяемого поля слова данных равны соответственно 1 и 5 (счет ведется от нуля, слева направо),  $P=0$ . Тогда  $B1=E5=1$ , а на остальных выходах дешифраторов DC1 и DC2 сформированы сигналы лог. 0. Сигнал  $P=0$  проходит через первый элемент Исключающее ИЛИ без инвертирования ( $M0=0$ ), так как на его верхнем входе присутствует сигнал лог. 0, а через второй элемент — с инвертированием, поэтому  $M1=1$ . Так как четыре последующих элемента не инвертируют транслируемый сигнал, то  $M2=M3=M4=M5=1$ . Сигнал  $E5=1$  проходит через элемент ИЛИ и переводит соответствующий элемент Исключающее ИЛИ в режим инвертирования, поэтому  $M6=0$ . Поскольку  $B7=E6=0$ , выполняется условие  $M7=M6=0$ . Таким образом, получаем следующий код маски  $M0...M7$ : 01111100. При  $P=1$  этот код инвертируется: 10000011. Отметим, что в дешифраторе DC2 использованы семь (а не восемь) выходов.

Недостаток схемы (рис. 4.26) — ее невысокое быстродействие из-за последовательной цепи распространения сигнала  $P$ . Предлагаемое усовершенствование [85] состоит в двустороннем формировании кода маски с применением схем ускоренного переноса. Прежде чем рассмотреть окончательный вариант схемы (рис. 4.28), поясним идею двустороннего формирования кода маски на простом примере (рис. 4.27).

В этой схеме сигнал  $P$  распространяется параллельно по двум встречным направлениям, т. е. время прохождения всего пути уменьшено вдвое. Левая часть схемы не отличается от рассмотренной ранее; правая является своеобразным «зеркальным отражением» левой и работает аналогично. В данном варианте оба дешифратора имеют по семь выходов; фактически исключен еще один элемент ИЛИ (первый его вход соединен с шиной лог. 0, поэтому элемент просто транслирует сигнал со второго входа на выход как отрезок провода; эти элементы показаны лишь для придания схеме некой однородности). Поэтому помимо ускорения работы генератора достигнута незначительная экономия аппаратуры.

В окончательном варианте схемы рис. 4.28 использована рассмотренная идея двустороннего формирования кода маски: сигнал  $P$  вводится одновременно со стороны разрядов 0 и 31; схема в некотором смысле симметрична и делится на две 16-разрядные части, работающие параллельно. В каждой части имеются три последовательно включенные схемы ускоренного переноса, каждая из которых анализирует определенную группу выходных сигналов дешифраторов DC1 и DC2. Сигналы  $M0—M3$ ,  $M4—M7$ ,  $M8—M11$  и т. д. формируются 4-разрядными секциями, аналогичными показанным на рис. 4.27.

Рассмотрим иное решение [86]. Сначала — общая идея (рис. 4.29). Как и в предыдущих схемах, в качестве исходной информации для построения кода маски использованы номера начального и конечного битов, соответствующих границам сохраняемой части слова данных, а также сигнал  $P$ , определяющий «полярность» маски.

Пусть, например, номера  $B^*$  и  $E^*$  начального и конечного битов равны 5 и 13 (здесь и далее счет разрядов ведется от нуля, справа налево). Тогда

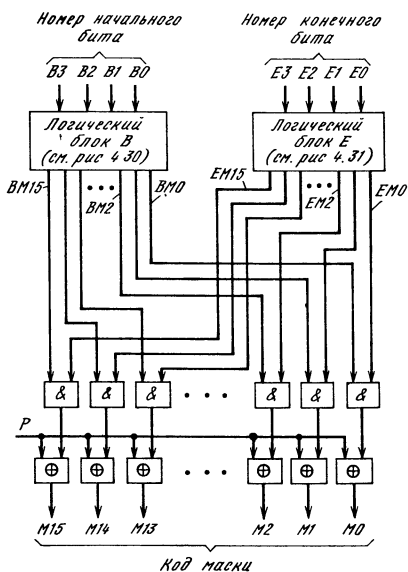


Рис. 4.29. Генератор кодов маски на основе логических блоков В и Е

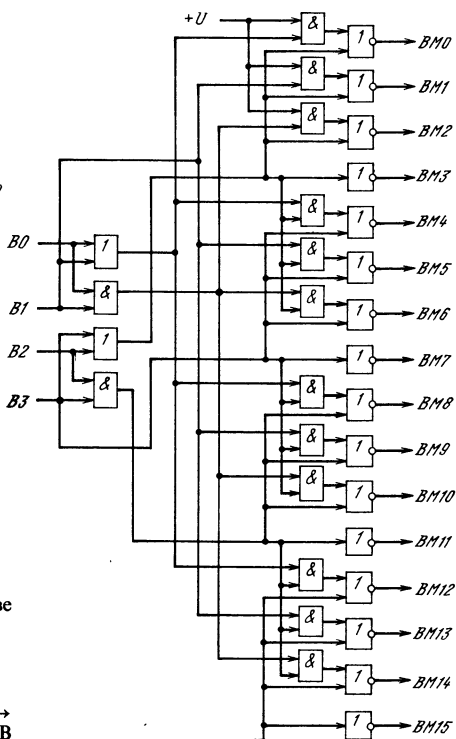


Рис. 4.30. Структура логического блока В

на выходах логических блоков В и Е сформируются следующие коды BM15...BM0 и EM15...EM0:

	Номера граничных разрядов кода маски:															
	$E^* = 13, B^* = 5$															
Номер разряда	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Код BM15...BM0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
Код EM15...EM0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$BM_i \& EM_i$	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0

Граничные разряды кода маски ( $P=0$ )

Эти коды после поразрядного логического умножения ( $BM_i \& EM_i$ ) образуют результирующий код маски, который можно проинвертировать сигналом  $P=1$ .

Логический блок В (рис. 4.30) формирует коды в соответствии с табл. 4.3.

Таблица 4.3

Входной код В3 В2 В1 В0	Выходной код ВМ15...ВМ0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 0 0 0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 0 0 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0 0 1 0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0 0 1 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0 1 0 0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0 1 0 1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0 1 1 0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0 1 1 1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
1 0 0 0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1 0 0 1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
1 0 1 0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
1 0 1 1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
1 1 0 0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1 1 0 1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1 1 1 0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1 1 1 1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Логический блок Е можно построить на основе логического блока В двумя способами. Первый (рис. 4.31, а) заключается в подаче кода Е3 Е2 Е1 Е0 на входы блока В, инвертировании полученного результата и его сдвиге на один разряд влево, как показано в приведенном на рисунке примере. Освободившийся при сдвиге разряд заполняется единицей. Инвертирование результата эквивалентно замене выходных элементов ИЛИ—НЕ в блоке В (см. рис. 4.30) элементами ИЛИ и замене выходных элементов

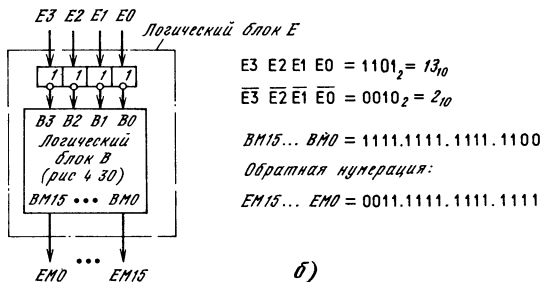
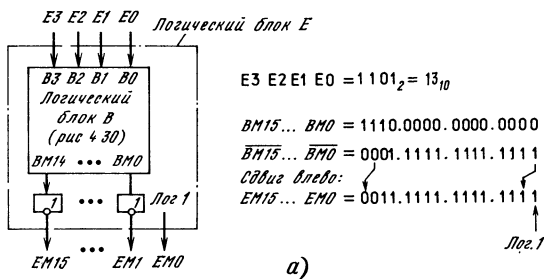


Рис. 4.31. Первый (а) и второй (б) варианты построения логического блока Е на основе логического блока В

НЕ переключателями, соединяющими их входы с выходами. Второй способ (рис. 4.31, б) связан с инвертированием входного кода (или с пересмотром логики работы входного каскада блока В) и выдачей результата с обратной нумерацией разрядов.

#### 4.14. Усовершенствованные генераторы псевдослучайных чисел [87]

Генератор (рис. 4.32), построенный на сдвиговом регистре RG с элементом суммирования по модулю два (Исключающее ИЛИ) в цепи обратной связи, сигналом S устанавливается в некоторое ненулевое состояние, а затем с поступлением положительных фронтов синхросигнала С формирует периодическую последовательность 6-разрядных двоичных чисел  $Q_0...Q_5$ . Каждый период состоит из чисел от 1 до  $63_{10}$ , расположенных в некотором (псевдослучайном) порядке, зависящем от схемного решения цепи обратной связи.

Недостаток схемы в том, что из последовательности исключено число 0. Действительно, нулевой код преобразуется «сам в себя», поэтому он представляет собой своеобразную ловушку, из которой было бы невозможно выбраться. В усовершенствованном генераторе (рис. 4.33, а) число 0 включено в последовательность чисел, так что период повторения увеличен до 64.

Мультиплексор MS управляется сигналом с выхода элемента ИЛИ—НЕ. Если этот сигнал равен нулю, то MS транслирует сигнал с выхода элемента Исключающее ИЛИ, если единице—то с выхода элемента НЕ. В «нормальном» режиме, когда хотя бы один из сигналов  $Q_0...Q_4$  ненулевой, на выходе элемента ИЛИ—НЕ присутствует сигнал лог. 0, поэтому схемы рис. 4.33, а и 4.32, по существу, можно считать одинаковыми. Однако при  $Q_0=Q_1=...=Q_4=0$  на выходе элемента ИЛИ—НЕ формируется сигнал лог. 1, MS формирует на выходе сигнал лог. 0, так как  $Q_5=1$ .

В следующем такте в регистре RG устанавливается нулевой код, на выходе элемента ИЛИ—НЕ по-прежнему присутствует сигнал лог. 1, на выходе MS формируется сигнал лог. 1, так как  $Q_5=0$ . Поэтому при поступлении очередного положительного фронта синхросигнала С в верхнем разряде регистра устанавливается сигнал  $Q_0=1$ , сигналы  $Q_1...Q_5$  нулевые. На выходе элемента ИЛИ—НЕ формируется сигнал лог. 0, генератор возвращается в «нормальный» режим. Далее процесс повторяется.

Остальные генераторы (рис. 4.33, б—г) работают аналогично. Генератор на рис. 4.33, б отличается от рассмотренного последовательностью

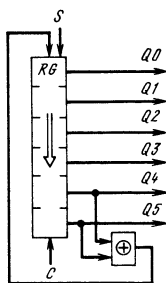


Рис. 4.32. «Классический» генератор псевдослучайных чисел

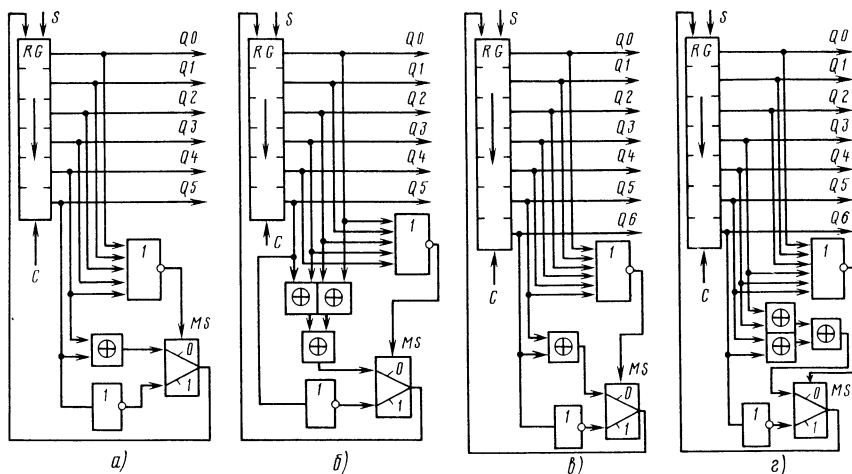


Рис. 4.33. Усовершенствованные генераторы псевдослучайных чисел

формируемых чисел. Генератор на рис. 4.33,г отличается от генератора на рис. 4.33,в видом последовательности, длина периода составляет 128 чисел.

#### 4.15. Контроль степени загрузки процессора полезной работой [88]

Проектировщик микропроцессорной системы реального времени стремится возложить на процессор как можно больше работы по обслуживанию внешних источников и приемников информации. Например, оправдано его желание увеличить число обслуживаемых каналов связи или повысить точность вычислений, но возможности процессора ограничены, и на этапе «бумажного» проектирования не всегда можно с уверенностью предсказать максимально допустимое число каналов или верхний предел длительности цикла вычислений. Даже после тщательного математического моделирования системы решающее значение имеет эксперимент в реальных условиях.

Процессор микроЭВМ (рис. 4.34,а) при появлении данных от источников принимает и обрабатывает их, а результаты вычислений посылает

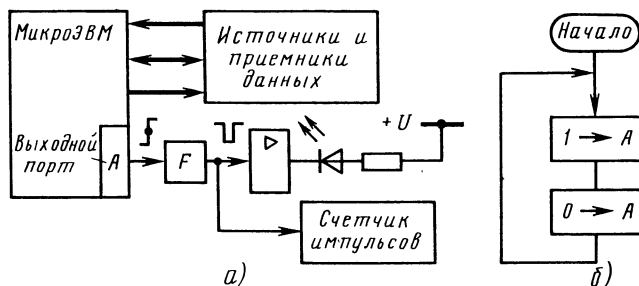


Рис. 4.34. Подключение к выходному порту микроЭВМ элементов контроля степени загрузки процессора полезной работой (а) и алгоритм ожидания полезной работы (б)

к приемникам. При правильной работе системы процессор успевает рассасывать входной поток данных. Точнее, между периодами обработки «порций» данных остаются промежутки времени, когда процессор не занят полезной работой. Если таких промежутков нет, то, вероятнее всего, это означает, что процессор не справляется с потоком данных; очередь на обслуживание растет до определенного уровня, а затем входные данные просто теряются.

В отсутствие входных данных процессор выполняет фоновую программу (рис. 4.34, б), которая предусматривает попеременную выдачу сигналов  $A=1$  и  $A=0$  в один из разрядов выходного порта. Формирователь импульса  $F$  по положительному фронту сигнала  $A$  вырабатывает отрицательный импульс, который проходит через усилитель с открытым коллектором и вызывает кратковременное включение светодиода. Таким образом, если процессор «свободен», то светодиод излучает световые импульсы, которые воспринимаются как непрерывное свечение.

При получении сигнала прерывания в связи с готовностью входных данных процессор микроЭВМ временно прекращает выполнение фоновой программы, запоминает в ОЗУ параметры возврата к ней и переходит к прерывающей программе, в которой предусмотрены считывание, обработка данных и выдача результатов к приемникам. В процессе работы по обслуживанию источников и приемников информации сигнал  $A$  не изменяется, поэтому светодиод выключен. После завершения прерывающей программы процессор возвращается к фоновому заданию, светодиод вновь начинает излучать свет.

Таким образом, проектировщик системы, наблюдая за состоянием светодиода, может качественно оценить степень загрузки процессора полезной работой. Для точной оценки можно использовать приборы — счетчик импульсов, частотомер, логический анализатор и т. п.

#### **4.16. Уменьшение энергопотребления однокристалльной микроЭВМ, выполняющей в отсутствие штатного напряжения питания функции часов с питанием от резервной батареи [89]**

Интеллектуальные устройства управления различными приборами — видеоманитофоном, телевизором и т. п. — помимо выполнения прочих действий ведут учет текущего времени. Чтобы не вводить в устройство отдельную БИС, используемую в наручных часах (или какую-либо подобную схему), функции отсчета времени дополнительно возлагаются на однокристалльную микроЭВМ, которая управляет прибором. Для уменьшения стоимости устройства кварцевый резонатор, задающий тактовую частоту работы микроЭВМ, заменяют времязадающей RC- или LC-цепью. Однако эта цепь из-за разброса параметров ее элементов (при серийном производстве устройств) не обеспечивает приемлемой точности работы часов. Поэтому в качестве эталона при подсчете времени используют период переменного напряжения первичного питания частотой 60 Гц. Эта частота в энергосети США поддерживается с высокой точностью.

Питание микроЭВМ в отсутствие штатного напряжения осуществляется от аккумулятора, батареи или конденсатора большой емкости. Последнее предпочтительно, так как для увеличения срока службы аккумулятора нужно соблюдать правила его подзарядки, поэтому потребовалась бы дополнительная схема, а батарею придется иногда менять, что также нежелательно. При отключении прибора от сети переменного тока «служба времени» не должна прекращать работу, несмотря на то, что исчез эталон — сигнал частотой 60 Гц. Можно было бы вести счет времени, опираясь на тактовую частоту работы микроЭВМ, если предварительно, до выключения штатного питающего

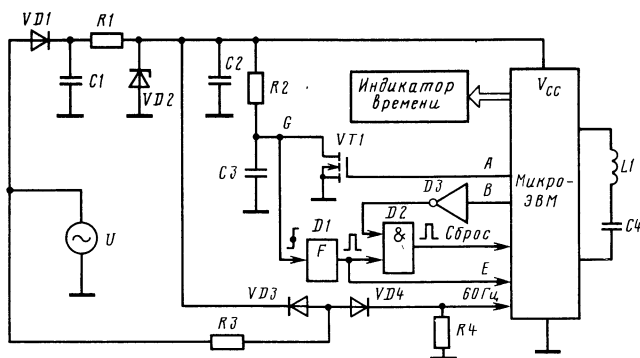


Рис. 4.35. Схема подключения однокристалльной микроЭВМ к источнику переменного напряжения  $U$ .

Конденсатор  $C1$  выполняет функцию аккумулятора

напряжения, вычислены соответствующие коэффициенты пересчета, зависящие от параметров константной времязадающей RC- или LC-цепи. Такое решение, однако, требует постоянного поддержания микроЭВМ в активном состоянии, при котором ее энергопотребление относительно велико. Было бы желательно в отсутствие штатного напряжения питания перевести микроЭВМ в пассивное состояние с низким потреблением тока (например, с уменьшением тока от 100 до 10 мкА), одновременно сохранив ее способность работать по программе и вести счет текущего времени. Как видим, требования несовместимые — микроЭВМ должна не работать и в то же время работать! Предлагаемое решение (рис. 4.35) как раз и направлено на разрешение указанного противоречия.

Схема содержит микроЭВМ с цифровым индикатором времени, источник переменного напряжения  $U$  питания (таким источником может быть низковольтная вторичная обмотка силового трансформатора), однополупериодный выпрямитель (диод  $VD1$  — конденсатор большой емкости  $C1$ ), стабилизатор (токоограничительный резистор  $R1$  — стабилитрон  $VD2$ ), фильтрующий конденсатор  $C2$ , схему формирования сигнала сброса (RC-цепь  $R2C3$  — n-канальный МДП-транзистор  $VT1$  — одновибратор  $D1$  — элемент И  $D2$  — элемент НЕ  $D3$ ), времязадающую цепь  $L1C4$  генератора тактовой частоты микроЭВМ и формирователь импульсов частотой 60 Гц (резисторы  $R3, R4$ , диоды  $VD3, VD4$ ).

При включении источника переменного напряжения  $U$  конденсатор  $C1$  заряжается положительными полуволнами этого напряжения через диод  $VD1$ ; в установившемся режиме на этом конденсаторе поддерживается постоянное напряжение, близкое к пиковому напряжению положительных полуволн, уменьшенному на величину падения напряжения на открытом диоде  $VD1$ . Положительное напряжение на конденсаторе  $C1$  вызывает протекание тока через резистор  $R1$ . Часть этого тока ответвляется в стабилитрон  $VD2$ , благодаря которому на входе питания  $V_{CC}$  микроЭВМ поддерживается достаточно стабильное напряжение. Конденсатор  $C2$  сглаживает пульсации этого напряжения.

Положительные полуволны переменного напряжения  $U$  проходят через резистор  $R3$ , диод  $VD4$  и поступают на вход микроЭВМ в качестве меток времени. Диод  $VD3$  ограничивает амплитуду сигнала на его аноде. Как будет показано, после выключения штатного напряжения  $U$  питания микроЭВМ переходит в пассивное состояние, однако она периодически выходит из него, чтобы прибавить единицу к счетчику периодов (размещенному в ОЗУ микроЭВМ) и вновь вернуться в пассивное состояние. Таким образом, благодаря преимущественному пребыванию микроЭВМ в пассивном состоянии

экономится энергия, отбираемая от конденсатора  $C1$ , а счет времени не прекращается. Цепь  $R2C3$  определяет длительность интервала времени пребывания микроЭВМ в пассивном состоянии в каждом периоде.

Транзистор  $VT1$  управляется от микроЭВМ и служит для быстрого разряда конденсатора  $C3$ . Одновибратор  $D1$  реагирует на нарастание входного напряжения до порогового уровня выдачей положительного импульса  $E=1$ , который при  $B=0$  проходит через элемент И  $D2$  на вход начальной установки микроЭВМ.

Как отмечалось, микроЭВМ может находиться в одном из двух состояний — активном или пассивном. В активном состоянии микроЭВМ работает по программе, размещенной во внутреннем ПЗУ; для хранения текущей информации используются регистры процессора и ячейки внутреннего статического ОЗУ. В пассивном состоянии содержимое статического ОЗУ остается неизменным, генератор тактовой частоты выключается, потребление тока снижается. Переход из активного в пассивное состояние осуществляется под управлением одной из команд, специально введенных для этой цели, а обратный переход возможен только после воздействия на микроЭВМ импульсного сигнала  $Cброс$ .

Рассмотрим сначала процесс первого включения источника переменного напряжения  $U$ , когда исходно конденсатор  $C1$  был полностью разряжен (рис. 4.36). В момент  $t_0$  включается источник переменного напряжения  $U$  питания, в момент  $t_1$  на входе  $V_{cc}$  устанавливается стабилизированное напряжение, в момент  $t_2$  напряжение на конденсаторе  $C3$  достигает порога срабатывания одновибратора  $D1$  (транзистор  $VT1$  выключен сигналом  $A=0$ ), в микроЭВМ поступает сигнал  $Cброс$  (сигнал  $B=0$  не препятствует этому), по окончании которого процессор приступает к работе в соответствии с блок-схемой алгоритма, приведенной на рис. 4.37.

Прежде всего процессор выясняет, включено ли штатное напряжение питания (блок 1). Для этого он программно выполняет ряд проверок входного сигнала 60 Гц. Если в течение времени, чуть превышающего половину периода, проверяемый сигнал не принял единичного значения, то это означает, что штатное напряжение питания не включено, поэтому процессор переходит к выполнению действий, предписанных блоком 8. Отметим, что на этапе

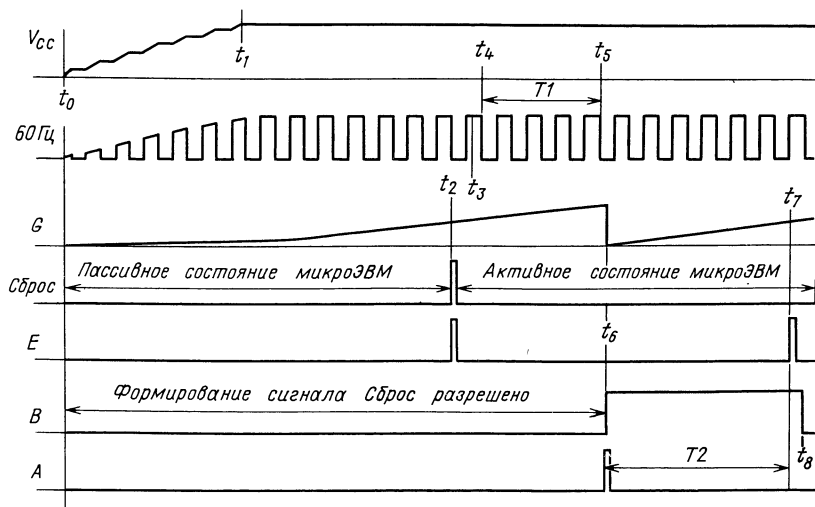


Рис. 4.36. Временные диаграммы работы схемы (рис. 4.35) при первоначальном включении штатного напряжения питания

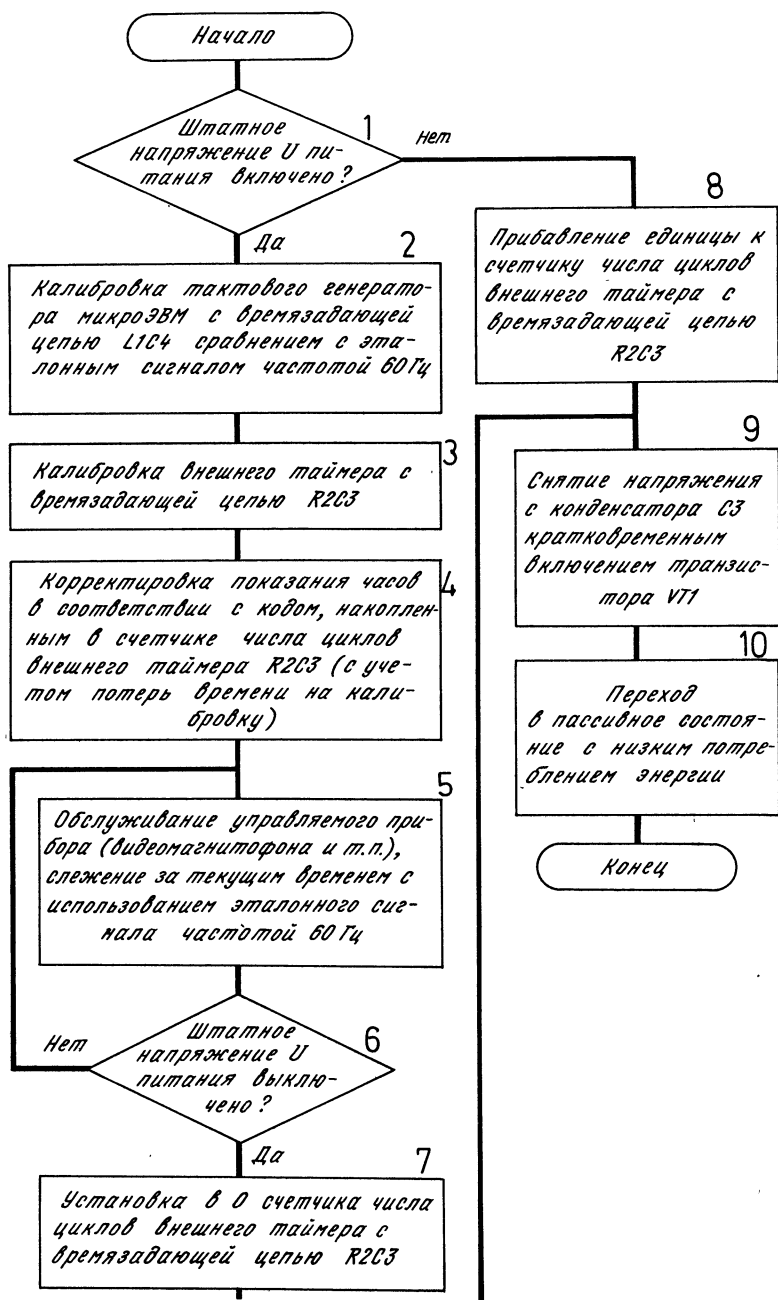


Рис. 4.37. Блок-схема алгоритма работы микроЭВМ (рис. 4.35)

проверки наличия штатного напряжения питания (блок 1) еще нет необходимости использовать точный эталон времени—достаточно опираться на временные параметры цепи L1C4 (пока точно не известные), которая задает тактовую частоту работы микроЭВМ.

В данном примере в момент  $t_3$  процессор, выполняя очередную проверку, обнаруживает, что сигнал 60 Гц принял единичное значение. Поэтому осуществляется переход к блоку 2. Этот блок предписывает процессору с достаточно высокой точностью определить частоту своего тактового генератора, которая зависит от параметров цепи L1C4. Эталон (сигнал с высокостабильной частотой 60 Гц) имеется, поэтому задача решается на основе подсчета числа однотипных команд, которые процессор успевает выполнить за некоторый известный промежуток времени T1.

Этот промежуток ( $t_4 - t_5$ ) в данном примере выбран равным четырем периодам сигнала 60 Гц. В результате несложных вычислений процессор определяет частоту своего тактового генератора практически с той же точностью, с которой поддерживается частота 60 Гц в сети переменного тока. Таким образом, процессор с помощью таймера, входящего в состав микроЭВМ, способен теперь с высокой точностью измерить параметры времязадающей цепи R2C3 (см. блок 3).

Для этого в момент  $t_6$  формируется сигнал  $B=1$  запрета работы элемента И D2 и вырабатывается импульс  $A=1$ , конденсатор C3 быстро разряжается, процессор начинает отсчет времени и ожидает появления сигнала  $E=1$ . При получении этого сигнала процессор регистрирует длительность промежутка времени T2 и затем в момент  $t_8$  переводит сигнал B в исходное состояние (лог. 0). Для повышения точности измерения промежутка времени T2 проводятся несколько описанных циклов с усреднением полученных результатов (на рисунке показан только один цикл измерения).

Далее (блок 4) процессор корректирует содержимое ячеек ОЗУ, выделенных для хранения показаний часов. Старое содержимое этих ячеек отражает состояние часов в момент последнего выключения прибора, которое, возможно, произошло несколько дней или недель тому назад. Сколько времени прошло с того момента? Информация об этом в виде числа циклов срабатывания внешнего таймера на основе цепи R2C3 также хранится в ОЗУ. Поэтому процессор, зная длительность цикла и число прошедших циклов, вычисляет длительность интервала времени, в течение которого прибор был выключен. Добавив этот интервал времени к показаниям часов, получим текущее время (здесь необходимо также учесть затраты времени на калибровку, см. блоки 2 и 3). При первом включении прибора в ячейках ОЗУ микроЭВМ находится случайная информация, поэтому, как и при работе с обычными часами, необходима их начальная установка.

После выполнения описанных подготовительных операций (блоки 1—4) микроЭВМ приступает к решению основной задачи—обслуживанию прибора, в который она встроена (блок 5), в соответствии с командами, поступающими с пульта управления. При этом процессор также следит за текущим временем и, как в обычных часах, выдает на индикатор соответствующие показания. Здесь возможна работа в режиме прерывания по сигналу 60 Гц либо в режиме периодического программного опроса этого сигнала.

После выключения штатного напряжения U питания (блоки 6, 7), т. е. при обнаружении отсутствия нескольких периодов сигнала 60 Гц, процессор принимает к сведению, что предстоит работа с энергоснабжением от ранее заряженного конденсатора C1. Счетчик числа циклов внешнего таймера (R2C3), размещенный в ОЗУ, устанавливается в 0 (блок 7), затем осуществляется разряд конденсатора C3 сигналом  $A=1$  (блок 9) и, наконец, процессор, выполняя соответствующую команду программы, переводит микроЭВМ в пассивное состояние с низким потреблением энергии. В данном случае сигнал  $B=0$ , поэтому в дальнейшем, после заряда конденсатора C3, сигнал  $E=1$  с выхода одновибратора поступает на вход начальной установки микроЭВМ, которая вновь переходит в активный режим и проверяет наличие штатного

напряжения питания (блок 1). На этот раз выясняется, что прибор выключен, поэтому к счетчику числа циклов внешнего таймера (R2C3) прибавляется единица, и далее процессы повторяются.

Таким образом, в отсутствие штатного напряжения питания большую часть времени микроЭВМ проводит в пассивном состоянии (в те периоды, когда происходит накопление заряда на конденсаторе C3), а счет времени не прерывается. После формирования очередного сигнала Сброс микроЭВМ на короткое время переходит в активное состояние, выполняет действия, предписанные блоками 1, 8, 9, 10, и вновь переходит в пассивное состояние, пока напряжение на конденсаторе C3 не достигнет порога срабатывания одновибратора.

Чтобы уменьшить время проверки наличия сигнала 60 Гц (блок 1), можно ввести в схему дополнительный выпрямитель, предпочтительно — двухполупериодный. Тогда проверка наличия штатного напряжения питания сведется к однократному программному опросу выхода этого выпрямителя. Элементы D2 и D3 можно исключить из схемы, при этом выход одновибратора соединяется непосредственно с входом начальной установки микроЭВМ, а сигнал Е не используется. В этом случае, однако, несколько усложняется алгоритм анализа состояния микроЭВМ по окончании сигнала Сброс. Действительно, этот сигнал будет дополнительно сопровождать процесс калибровки внешнего таймера (R2C3), и всякий раз при окончании сигнала Сброс придется анализировать состояние некоторого флага, хранимого в ОЗУ, чтобы отличить «обычный» сигнал от «искусственного», который возникает при калибровке.

#### 4.17. Многоканальный тактовый генератор [90]

Синхросигналы, передаваемые с выхода тактового генератора в различные точки вычислительного устройства, в той или иной мере искажаются из-за разброса задержек усилителей, неодинаковых параметров линий связи и т. п. Поэтому при проектировании устройства приходится рассчитывать на наименее худшее сочетание задержек «размноженных» синхросигналов в разных каналах: чем больше разброс задержек, тем ниже производительность устройства. Для более или менее точного выравнивания задержек в разных каналах необходима индивидуальная регулировка сигналов «по месту», выполняемая на конкретном образце устройства (вручную или с помощью автоматизированной контрольно-наладочной аппаратуры). В схеме тактового генератора (рис. 4.38) предусмотрена индивидуальная подстройка сигналов в каждом канале, а также общая регулировка их частоты.

Генератор содержит сдвиговой регистр D2, замкнутый в кольцо через инвертирующий мультиплексор D1, регистр управления D11, инвертор D3 и блоки 1—N формирования синхросигналов Q1—Q<sub>N</sub>. Последние могут вырабатываться практически одновременно или с некоторым взаимным сдвигом во времени, компенсирующим искажения при их передаче в различные точки вычислительного устройства. Блок 1 (2, 3, ..., N) содержит неинвертирующие D4, D8, D9 и инвертирующий D5 мультиплексоры, D-триггеры D6, D7 и элемент И D10. Настройка блока на формирование импульса с заданными временными параметрами выполняется подачей соответствующего 8-разрядного кода с выходов регистра D11 на управляющие входы мультиплексоров D4, D5, D8, D9. Частота сигналов Q1—Q<sub>N</sub> регулируется трехразрядным кодом на управляющих входах мультиплексора D1.

В исходном состоянии на вход генератора поступает непрерывная последовательность синхронимпульсов CL со скважностью, равной двум, регистр D2 установлен в нуль сигналом R=1, триггер D6 (D7) при каждом переходе сигнала CL из 1 в 0 подтверждает состояние 0 (1), элемент И D10 закрыт по нижнему входу, Q1=Q2=...=Q<sub>N</sub>=0. После снятия сигнала R=1 с поступлением положительных фронтов сигнала CL сдвиговой регистр D2

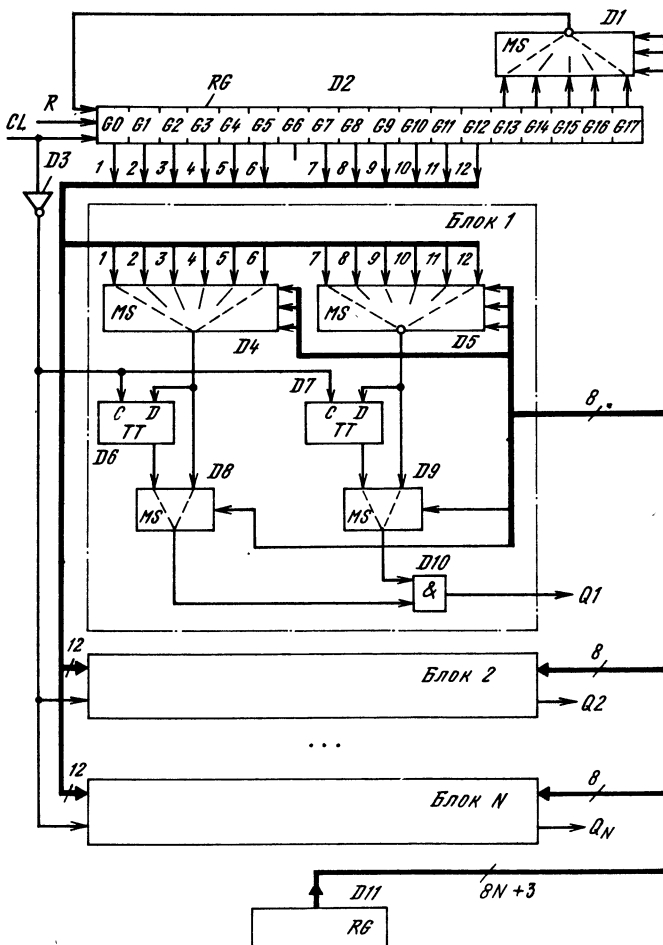


Рис. 4.38. Многоканальный тактовый генератор

начинает заполняться единицами, поступающими с выхода инвертирующего мультиплексора D1 (рис. 4.39). В зависимости от настройки этого мультиплексора поток единиц достигает выбранного разряда (G13—G17), после чего начинается заполнение регистра нулями и т. д. Процесс повторяется с периодом от  $T_{\min}$  до  $T_{\max}$ .

Момент формирования сигнала  $Q1=1$  определяется настройкой мультиплексоров D4 и D8, момент перехода Q1 в нуль — настройкой мультиплексоров D5 и D9. Триггеры D6 и D7 задерживают сигналы с выходов мультиплексоров D4 и D5 на половину периода сигнала CL, т. е. позволяют регулировать моменты формирования положительного и отрицательного фронтов сигнала Q1 с повышенной точностью.

В примерах, приведенных на рис. 4.39, сигналы Q1 и Q1\* формируются при различных вариантах настройки мультиплексоров D4 и D5 (мульти-

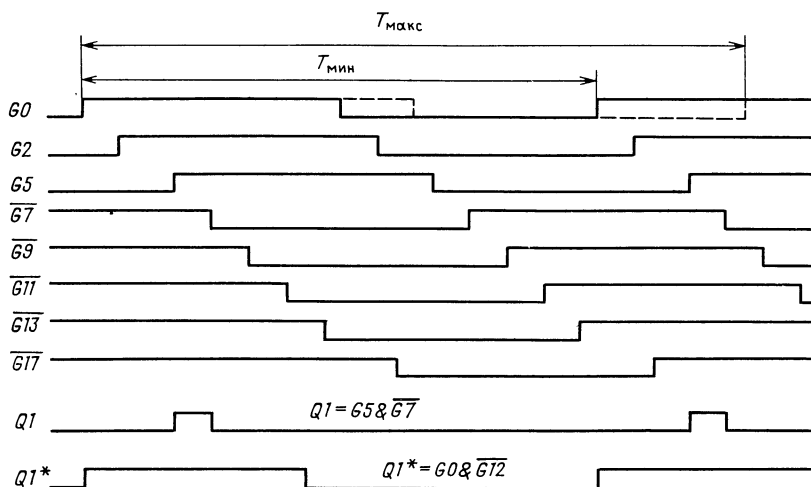


Рис. 4.39. Временные диаграммы работы тактового генератора (рис. 4.38)

плексоры D8 и D9 настроены на передачу сигналов непосредственно с выходов D4 и D5). Для формирования сигнала Q1 (Q1\*) мультиплексоры D4 и D5 настраиваются на передачу сигналов с выходов разрядов G5 и G7 (G0 и G12) регистра D2.

## 4.18. Многопроцессорная система для параллельной обработки операндов [91]

При работе с векторными данными, например при вычислении скалярного произведения двух векторов, ЭВМ должна выполнить ряд однотипных операций над разными компонентами данных. Эти операции могут выполняться одновременно, если использовать ряд независимых процессоров в соответствии с размерностью векторов. Далее рассмотрена гибкая структура многопроцессорной системы с переменной размерностью, упрощенной аппаратурой ввода—вывода и возможностью простого удаления неисправного процессора без снижения ее производительности.

В системе, показанной на рис. 4.40, каждый процессор в порядке очередности (этот порядок не всегда линейен) получает задание с входной шины в виде одного 72-разрядного слова, в котором, например, содержатся два 32-разрядных операнда-сомножителя и 8-разрядный код операции умножения. Получив задание, процессор немедленно приступает к его выполнению. Время выполнения каждого задания заранее известно и определяется числом периодов сигнала синхронизации CL, общего для всех процессоров.

Источник информации, например ЦП (на рисунке не показан), подтверждает готовность каждого кода  $D_i$  в группе заданий сигналом  $EN=1$ . С точки зрения ЦП безразлично, какой именно процессор из группы 1—5 примет очередное задание к исполнению—важно, что оно будет безусловно выполнено и результат  $D_0$  поступит на выходную шину через строго определенное число периодов сигнала CL. Таким образом, источник информации выдает в систему группу заданий и получает от нее в ответ группу результатов (в порядке выдачи заданий), не заботясь об адресации процессоров, занятых в обработке данных. Код  $D_0$  в данном примере содержит 64-разрядное произведение двух 32-разрядных чисел (в соответствии с кодом  $D_i$ ) и ряд

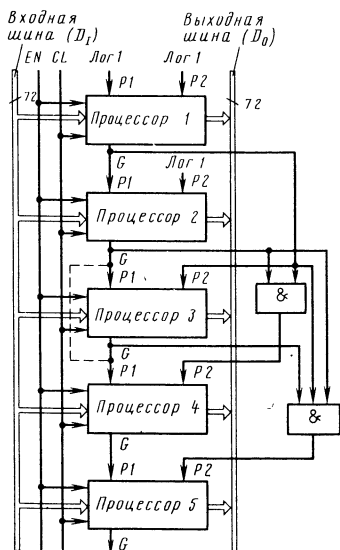


Рис. 4.40. Многопроцессорная система.

По мере усложнения выполняемых операций в работу включается все большее число процессоров. Штриховой линией показана «обходная» связь, которая вводится при удалении процессора 3

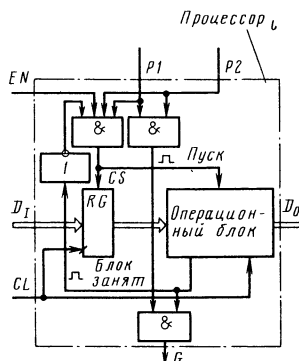


Рис. 4.41. Структура процессора.

Если операционный блок не занят или имеются свободные операционные блоки в процессорах более высоких уровней ( $P1 \& P2 = 0$ ) или и то, и другое, то сигнал  $G=0$  препятствует размещению очередного задания в процессорах более низких уровней

признаков, характеризующих результат (равенство результата нулю, признак переноса из старшего разряда и другие, как в обычном арифметическо-логическом устройстве).

Каждый процессор имеет два входа P1, P2 и один выход G для построения последовательной приоритетной цепи. На входы P1 и P2 процессора 1 и на вход P2 процессора 2 поданы сигналы лог. 1, вход P1 процессора 2 соединен с выходом G процессора 1, входы P1 и P2 последующих процессоров соединены непосредственно или через элементы И с выходами G предыдущих процессоров. Такое включение соответствует уменьшению приоритетов процессоров сверху вниз, по мере увеличения их номеров. (Приоритет в данном случае отражает степень участия процессора в обработке группы заданий.)

В исходном состоянии все процессоры не заняты. Процессор с номером  $i$  принимает задание (код  $D_i$ ) только в том случае, если заняты все предыдущие процессоры с номерами от 1 до  $i-1$ . Поэтому загрузка в систему группы заданий начинается с первого процессора и продолжается в естественном порядке, сверху вниз. Если первый процессор, получив задание, успел завершить его выполнение до того, как закончилась загрузка всей группы заданий, то он повторно включается в работу и т. д. Иными словами, первый процессор — самый активный; он не может бездействовать, когда есть нерешенные задачи. Второй, третий и следующие процессоры привлекаются к работе лишь в связи с тем, что все предыдущие процессоры заняты, а работа есть. В результате, группы сложных заданий, требующих для выполнения большого числа периодов сигнала CL, распределяются между процессорами равномерно (каждому заданию — свой процессор), а группы простых заданий выполняются двумя-тремя процессорами, попе-

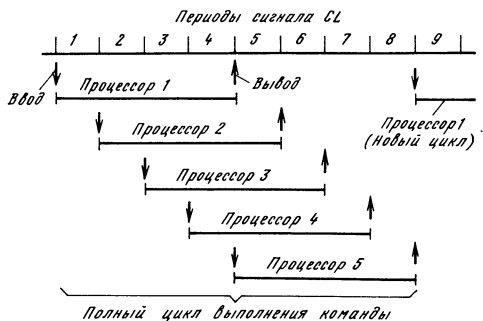


Рис. 4.42. Временная диаграмма работы системы (рис. 4.40) при выполнении сложной команды. Команда содержит пять заданий, каждое из которых выполняется за четыре периода сигнала CL. В работе участвуют все пять процессоров

ременно и многократно вступающими в действие, без привлечения к работе остальных.

Процессор на рис. 4.41 содержит операционный блок («собственно процессор»), входной регистр RG и фрагмент распределенной приоритетной схемы на элементах И, НЕ. Прием задания в процессор происходит при срабатывании 4-входного элемента И, т. е. когда операционный блок не занят, код  $D_1$  принял истинное значение ( $EN=1$ ), предыдущие процессоры заняты ( $P1=P2=1$ ). Эти условия окончательно проверяются в момент поступления фронта синхросигнала CL, чтобы не было «гонок» в последовательной приоритетной цепи. При удачной проверке задание (код  $D_1$ ) запоминается в регистре RG, операционный блок приступает к работе и формирует сигнал Блок занят, поддерживая этот сигнал в состоянии лог. 1 на протяжении всего интервала времени, выделенного для вычислений и выдачи результата (кода  $D_0$ ) в выходную шину. Сигнал  $G=1$  формируется только в том случае, когда данный операционный блок и все предшествующие заняты.

При выполнении сложной команды (рис. 4.42) ЦП последовательно выдает в систему пять заданий, каждое из которых обрабатывается за 4 периода сигнала CL. Процессор 1 освобождается только в начале пятого периода, поэтому на границе между четвертым и пятым периодами последнее задание принимается процессором 5 (все предыдущие процессоры заняты). К началу шестого периода процессор 1 свободен, и если бы в группе было шестое задание, то этот процессор включился бы в работу.

Менее сложная команда представлена пятью заданиями (рис. 4.43), каждое из которых завершается за два периода сигнала CL. В выполнении пяти заданий участвуют только три процессора, так как процессоры 1 и 2 успевают включиться в работу дважды.

Если в системе имеются резервные процессоры, включенные в цепочку на общих основаниях, то допустимо исключение из системы соответствующего числа штатных процессоров в связи с их ремонтом. Производительность



Рис. 4.43. Временная диаграмма работы системы (рис. 4.40) при выполнении простой команды. Команда содержит пять заданий, каждое из которых выполняется за два периода сигнала CL. В работе участвуют три процессора из пяти

системы не снижается, так как общее число работоспособных процессоров остается достаточным для максимального распараллеливания.

При исключении, например, процессора 3 (рис. 4.40) устанавливается перемычка между выходом G процессора 2, выходом P1 процессора 4 и входом трехходового элемента И, как показано штриховой линией. С точки зрения ЦП изменений в системе нет, так как распределение заданий между процессорами осуществляется без его участия, а в работу включается процессор 6 (на рисунке не показан).

Система без резерва не теряет работоспособности, если в ней остался хотя бы один исправный процессор. При этом снижается ее производительность, так как уменьшается степень параллелизма выполнения заданий (при одном исправном процессоре все задания выполняются последовательно). Центральный процессор после тестирования системы определяет число оставшихся исправных процессоров и при необходимости соответствующим образом корректирует временные диаграммы выдачи заданий в систему и приема результатов вычислений.

#### 4.19. Генератор импульсов с малым потреблением энергии [92]

Микросхема K561КТЗ содержит четыре электронных ключа. Генератор, показанный на рис. 4.44, выполнен на двух ключах SW1 и SW2, представленных для наглядности в виде механических выключателей. Ключ SW1 (SW2) разомкнут, если напряжение  $U_1$  ( $U_0$ ) между его управляющим входом 12 (5) и шиной нулевого потенциала не превышает порогового значения, примерно равного половине питающего напряжения, приложенного между контактами 14 и 7 микросхемы. Если напряжение на управляющем входе ключа превышает пороговое, то ключ замыкается (внутренним сопротивлением замкнутого ключа для простоты пренебрегаем, хотя оно и достаточно велико).

После включения напряжения питания оба ключа разомкнуты. Конденсатор C разряжен, поэтому напряжения на нем нет,  $U_1 = U_0$ . Зарядный ток начинает протекать от источника питания через последовательно включенные резисторы R1 и R2. Так как  $R_1 \gg R_2$ , напряжение  $U_0$  не достигает порога срабатывания ключа SW2, а в дальнейшем, по мере уменьшения зарядного тока, это напряжение стремится к 0. В то же время по мере накопления заряда на конденсаторе напряжение  $U_1$  экспоненциально возрастает.

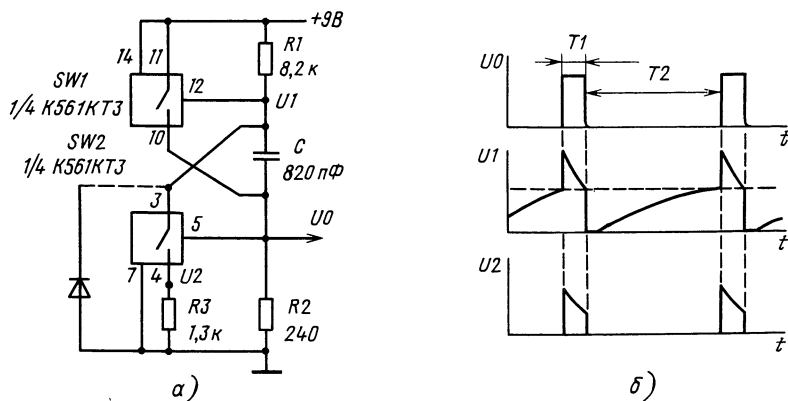


Рис. 4.44. Генератор импульсов (а) и временные диаграммы его работы (б)

Когда напряжение  $U_1$  достигает порога срабатывания ключа  $SW_1$ , этот ключ начинает замыкаться, сопротивление между его контактами 11 и 10 падает. В результате начинает повышаться напряжение  $U_0$ , а это, в свою очередь, приводит к повышению напряжения  $U_1$ , что способствует более быстрому замыканию ключа  $SW_1$ , повышению напряжения  $U_0$  и т. д. Таким образом, развивается лавинообразный процесс, напряжение  $U_0$  скачком возрастает до амплитудного значения, ключ  $SW_2$  также замыкается.

Сразу после замыкания обоих ключей нижняя обкладка конденсатора  $C$  подключается к шине питания  $+9$  В. Заряд, накопленный ранее в конденсаторе, не может измениться мгновенно, поэтому напряжение  $U_1$  скачком возрастает до уровня, превышающего  $U_0$  на величину, равную порогу срабатывания ключа  $SW_1$ . После этого напряжение  $U_1$  начинает экспоненциально уменьшаться с постоянной времени, равной  $CR_1R_3/(R_1+R_3)$ , и стремится достичь уровня, задаваемого делителем напряжения на резисторах  $R_1$  и  $R_3$ .

В процессе перезаряда конденсатора  $C$  напряжение  $U_1$  уменьшается до порога размыкания ключа  $SW_1$ , сопротивление между его контактами 11 и 10 начинает возрастать, напряжение  $U_0$  снижается, что, в свою очередь, приводит к ускорению размыкания ключа  $SW_1$  благодаря обратной связи через конденсатор. В результате развивается лавинообразный процесс размыкания обоих ключей. Напряжение  $U_0$  скачкообразно снижается, напряжение  $U_1$  также резко уменьшается, и, если напряжение на конденсаторе поменяло полярность (все определяется параметрами схемы), на контакте 3 ключа  $SW_2$  формируется отрицательный выброс напряжения. Для защиты ключа от этого выброса (если выброс есть) в схему вводится диод, как показано на рисунке штриховыми линиями.

После размыкания ключей конденсатор начинает заряжаться током через последовательно включенные резисторы  $R_1$  и  $R_2$ , и описанные ранее процессы повторяются.

При заданной емкости конденсатора  $C$  длительность паузы  $T_2$  между импульсами (рис. 4.44, б) задается резистором  $R_1$  (строго говоря, последовательной цепью из резисторов  $R_1$  и  $R_2$ ). Однако изменение длительности паузы  $T_2$  подбором резистора  $R_1$  приводит и к изменению длительности импульса  $T_1$ . Поэтому, чтобы задать нужную длительность импульса  $T_1$ , не меняя  $T_2$ , необходимо воспользоваться резистором  $R_3$ . Регулирование параметров импульсов осуществляется в широких пределах, при этом отношение  $T_1/T_2$  может быть как меньшим, так и большим 1. Резистор  $R_2$  определяет нагрузочную способность генератора. При указанных на рисунке параметрах схемы  $T_1=1,2$  мкс,  $T_2=6,8$  мкс, подключения диода не требуется.

## 4.20. Схема расширения импульсов [93]

Схема (рис. 4.45) содержит двоичный счетчик, инвертор, D-триггер и предназначена для расширения импульса  $A$ , поступающего, например, с магнитной карты (импульс  $A$  преобразуется в импульс  $B$ ).

В исходном состоянии (рис. 4.46, а)  $A=1$ ,  $B=1$ , на вход  $SEP$  разрешения счета счетчика поступает сигнал лог. 0, запрещающий его работу. На информационных выходах  $Q_0—Q_3$  счетчика присутствуют сигналы лог. 0, на выходе  $TC$  окончания счета сформирован сигнал лог. 0. На вход  $C$  синхронизации счетчика поступает сигнал  $CLK$ , не меняющий его состояния.

Входной отрицательный импульс  $A=0$  устанавливает триггер в 0, разрешая работу счетчика, и одновременно воздействует на вход установки 0 счетчика. (При безошибочной работе системы ввода информации с магнитной карты счетчик не требовал бы установки в 0 импульсом  $A$ , однако из-за возможных сбоев начальное состояние счетчика может отличаться от описанного ранее) После снятия сигнала  $A=0$  счетчик начинает подсчет входных импульсов

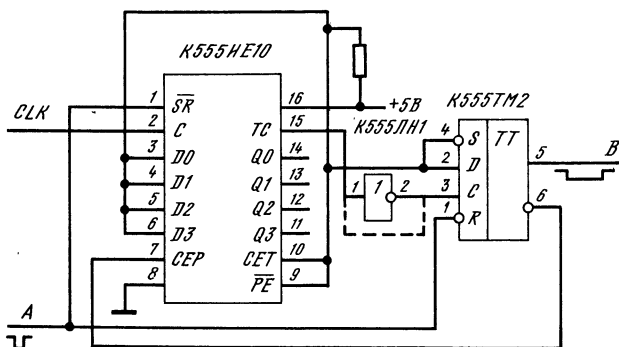


Рис. 4.45. Схема расширения импульсов

CLK. На его выходах формируется последовательность чисел 0, 1, 2, ..., 15, 0. При наличии кода  $15_{10}$  вырабатывается сигнал окончания счета TC, по спаду которого устанавливается в 1 триггер и счетчик приостанавливает работу, так как на его вход CEP поступает сигнал лог. 0. Таким образом, схема переходит в описанное ранее исходное состояние.

Если исключить из схемы инвертор, а вместо него ввести перемычку, показанную на рисунке штриховой линией, то расширенный импульс В (рис. 4.46, б) окажется несколько менее широким, так как счет прекращается при достижении кода 15, а не 0.

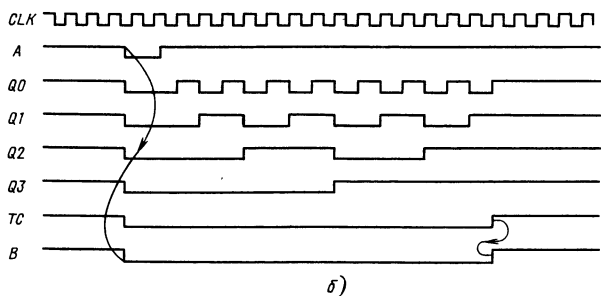
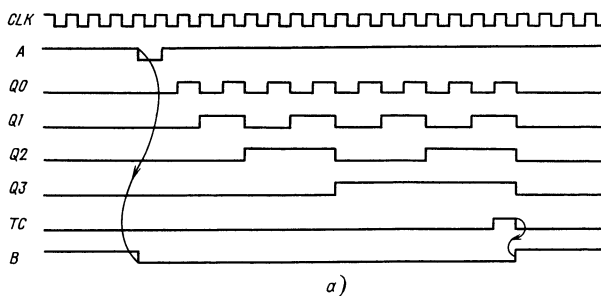


Рис. 4.46. Временные диаграммы работы схемы расширения импульсов (рис. 4.45): а—при наличии инвертора между выходом TC счетчика и входом С триггера; б—в отсутствие инвертора, при непосредственном соединении выхода TC с входом С (см. штриховую линию на рис. 4.45)

## 4.21. Двоичный счетчик со схемой контроля [94]

Чтобы проверить правильность работы  $n$ -разрядного двоичного счетчика, достаточно подать на его счетный вход последовательность из  $2^n$  импульсов и проследить за его выходной реакцией. Однако с увеличением  $n$  время проверки растет и может оказаться недопустимо большим.

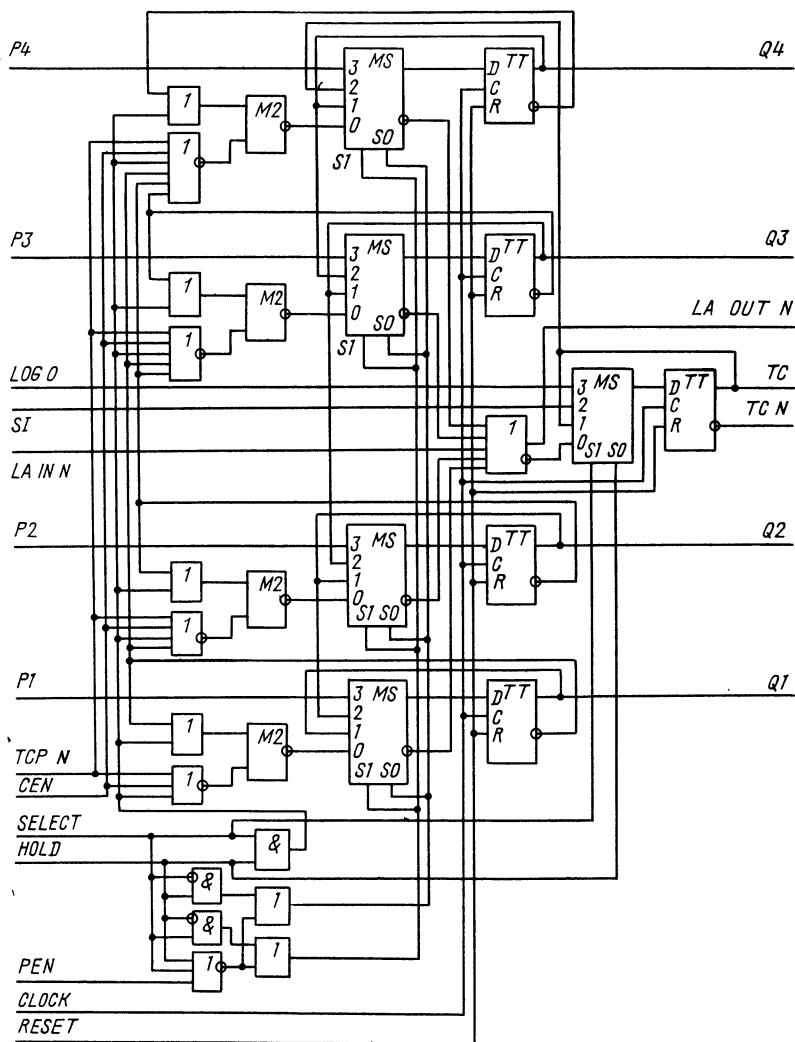


Рис. 4.47. Схема 4-разрядной секции двоичного счетчика

Для улучшения контролепригодности можно разделить счетчик на несколько малоразрядных секций и проверять их параллельно, с последующим контролем цепей переноса информации между секциями. Однако при этом возрастает число внешних выводов счетчика или вносятся нежелательные задержки в цепи распространения сигналов переноса.

Предлагается иное решение [94]. Для проверки счетчика или, по крайней мере, значительной части его составных частей, он временно преобразуется в сдвиговый регистр. Тестовая информация загружается в счетчик (сдвиговый регистр) извне и последовательно продвигается по всем его триггерам, выходная информация сравнивается с эталонной.

Такой счетчик, как показано в примере на рис. 4.47, содержит пять D-триггеров: четыре для хранения текущего кода Q1—Q4 (Q4—старший разряд) и один для хранения признака ТС заполнения счетчика единицами ( $Q1=Q2=Q3=Q4=1$ ). Элементы суммирования по модулю два с инвертированными выходами формируют сигнал лог. 1 только при равенстве соответствующих входных сигналов (00 и 11). Мультиплексоры MS в зависимости от входного управляющего кода SIS0 (00, 01, 10, 11) транслируют на выходы сигналы с соответствующих входов (0, 1, 2, 3).

Входы P1—P4 используются для начальной загрузки счетчика, вход RESET—для его асинхронной (безусловной) установки в 0, выходы LA OUT N и TC N подключаются соответственно к входам LA IN N и TCP N последующей (аналогичной) схемы, соответствующей более старшим разрядам. Вход LOG0 подключен к шине нулевого потенциала (шина на рисунке не показана), однако этот вход можно использовать при создании искусственной кодовой ситуации ( $Q1=Q2=Q3=Q4=0$ ,  $TC=1$ ) при тестировании, если на этапе синхронной установки счетчика в 0 (нижняя строка табл. 4.4) на вход LOG0 подать сигнал лог. 1.

Вход SI используется при тестировании для ввода очередного бита в счетчик—сдвиговый регистр. Выход Q1 подключается к входу SI аналогичной схемы, соответствующей группе более младших разрядов счетчика. На вход CLOCK поступают сигналы синхронизации, положительные фронты которых воздействуют на D-триггеры. Режимы работы счетчика представлены в табл. 4.4.

Таблица 4.4

Значения сигналов		Режим работы счетчика
SELECT	HOLD	
0	0	При $PEN=1$ и $CEN=0$ производится счет импульсов CLOCK. При $CEN=1$ счет приостанавливается. При $PEN=0$ (независимо от значения сигнала CEN) по положительному фронту импульса CLOCK в счетчик загружается код P1—P4
0	1	Счетчик сохраняет текущее состояние выходов и не реагирует на импульсы CLOCK
1	0	Счетчик преобразуется в сдвиговый регистр. Тестовая информация по положительным фронтам импульсов CLOCK проходит по цепи SI—TC—Q4—Q3—Q2—Q1
1	1	Счетчик устанавливается в 0 по положительному фронту сигнала CLOCK (производится синхронная установка счетчика в 0)

#### 4.22. Конструктивные особенности печатных плат, улучшающие эксплуатационные характеристики одноплатных устройств [99, 100]

В плате (рис. 4.48) контактные площадки печатной вилки имеют разную длину. При установке платы в соединитель (не показан) она перемещается в направлении, показанном стрелкой. Первыми входят в соприкосновение с соответствующими розетками соединителя контакты шины нулевого потенциала, затем контакты шины  $+15\text{ В}$  (нестабилизированное напряжение) и, наконец, функциональные контакты, по которым передаются адреса, данные, сигналы управления и т. п.

Такое решение позволяет устанавливать плату в действующее устройство «на ходу», не нарушая его работы по программе. Для исключения резкого провала нестабилизированного напряжения питания в момент соприкосновения контактов  $+15\text{ В}$  с соответствующими розетками соединителя применен стабилизатор  $S$ , обеспечивающий плавное нарастание выходного напряжения  $+9\text{ В}$ , питающего плату.

К моменту соприкосновения функциональных контактов с соответствующими розетками соединителя напряжение  $+9\text{ В}$  уже достигло номинального уровня, логические элементы-передатчики, соединенные с функциональными контактами, выключены благодаря действию сигнала начальной установки,

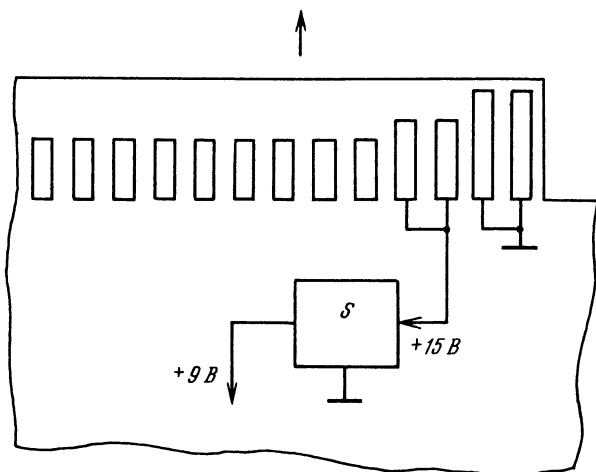


Рис. 4.48. Конструкция печатного соединителя с контактными площадками разной длины

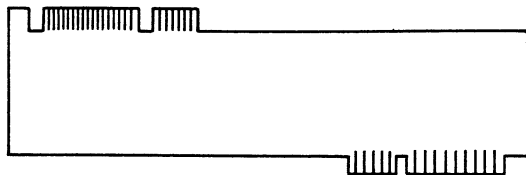


Рис. 4.49. Конструкция «платы-перевертыша»

сформированного соответствующей схемой (на рисунке не показана), реагирующей на включение питающего напряжения. Таким образом, устройство, в которое устанавливается данная плата, «не ощущает» каких-либо нежелательных воздействий по питающим и функциональным цепям.

Обратный процесс, при котором плату «на ходу» извлекают из соединителя, также протекает «незамеченным», так как сначала размыкаются функциональные контакты, затем снимается напряжение питания и только после этого разъединяется шина нулевого потенциала. Конечно, перед извлечением платы из соединителя ее следует перевести в логически пассивное состояние (иначе может нарушиться обмен сигналами через функциональные контакты и система «зависнет»), но это уже забота программиста.

Плата, показанная на рис. 4.49, представляет собой некий «перевертыш». В зависимости от интерфейса конкретной микроЭВМ задействуется тот или иной печатный соединитель. Например, соединители могут соответствовать разным стандартам ф. IBM, принятым в ПЭВМ IBM PS/2 и IBM PC/AT.

## Список литературы

1. Европейский патент EP 0.252.852 A1, МКИ GO6F 3/05.
2. Европейский патент EP 0.301.100 A1, МКИ GO6F 3/02.
3. Международная заявка PCT/WO/88/06765, МКИ GO6F 15/20.
4. Пат. 4.860.202 США, НКл 364/200.
5. Европейский патент EP 0.103.903 A1, МКИ GO6F 5/00.
6. Европейский патент EP 0.359.064 A2, МКИ GO6F 13/00.
7. Пат. 2.198.564 А, Великобритания, МКИ GO6F 7/28.
8. Пат. 2.180.967 А, Великобритания, МКИ GO6F 7/04.
9. Пат. 2.212.954 А, Великобритания, МКИ GO6F 1/00.
10. Пат. 4.788.642 США, НКл 364/200.
11. Пат. 4.992.970 США, НКл 364/900.
12. Пат. 4.951.250 США, НКл 364/900.
13. Пат. 2.189.912 А, Великобритания, МКИ GO6F 9/22.
14. Европейский патент EP 0.357.431 A2, МКИ GO6F 13/14.
15. Пат. 4.799.148 США, НКл 364/200.
16. Пат. 2.219.420 А, Великобритания, МКИ GO6F 9/46.
17. Пат. 4.881.169 США, НКл 364/200.
18. Европейский патент EP 0.358.300 A2, МКИ GO6F 12/06.
19. Международная заявка PCT/WO/89/02124, МКИ GO6F 11/16.
20. Европейский патент EP 0.344.426 A2, МКИ GO6F 11/18.
21. Международная заявка PCT/WO/86/07477, МКИ GO6F 11/20.
22. Европейский патент EP 0.276.445 A2, МКИ GO6F 11/00.
23. Пат. 4.878.168 США, НКл 364/200.
24. Европейский патент EP 0.308.086 A2, МКИ GO6F 11/00.
25. Европейский патент EP 0.328.899 A2, МКИ GO6F 11/10.
26. Европейский патент EP 0.353.435 A2, МКИ GO6F 11/16.
27. Европейский патент EP 0.096.783 A2, МКИ GO6F 11/00.
28. Пат. 2.179.183 А, Великобритания, МКИ GO6F 12/16.
29. Европейский патент EP 0.070.184 B1, МКИ GO6F 11/10.
30. Европейский патент EP 0.319.183 A2, МКИ GO6F 11/10.
31. Пат. 2.228.111 А, Великобритания, МКИ GO6F 11/30.
32. Пат. 2.149.945 А, Великобритания, МКИ GO6F 11/30.
33. Пат. 2.228.600 А, Великобритания, МКИ GO6F 11/00.
34. Пат. 2.227.861 А, Великобритания, МКИ GO6F 11/00.
35. Пат. 4.815.025 США, НКл 364/900.
36. Международная заявка PCT/WO/85/04736, МКИ GO6F 11/22.
37. Европейский патент EP 0.359.849 A1, МКИ GO6F 11/00.
38. Европейский патент EP 0.266.836 A2, МКИ GO6F 11/00.
39. Европейский патент EP 0.266.837 A2, МКИ GO6F 11/00.
40. Пат. 2.173.326 А, Великобритания, МКИ GO6F 13/38.
41. Европейский патент EP 0.169.676 A2, МКИ GO6F 11/00.
42. Европейский патент EP 0.366.115 A2, МКИ GO6F 7/02.
43. Европейский патент EP 0.072.219 B1, МКИ GO6F 12/06.
44. Пат. 4.831.552 США, НКл 364/200.
45. Пат. 2.204.163 А, Великобритания, МКИ GO6F 12/00.
46. Европейский патент EP 0.114.485 A2, МКИ GO6F 3/04.

47. Европейский патент EP 0.366.458 A2, МКИ G06F 12/06.
48. Европейский патент EP 0.367.426 A2, МКИ G06F 12/06.
49. Европейский патент EP 0.080.626 A2, МКИ G06F 12/06.
50. Пат. 2.190.771 А, Великобритания, МКИ G06F 12/06.
51. Европейский патент EP 0.332.910 A2, МКИ G06F 12/06.
52. Пат. 4.825.404 США, НКл 364/900.
53. Пат. 4.896.263 США, НКл 364/200.
54. Международная заявка PCT/WO/88/08573 A1, МКИ G06F 12/06.
55. Пат. 2.197.970 А, Великобритания, МКИ G06F 3/023.
56. Европейский патент EP 0.340.325 A1, МКИ G06F 12/06.
57. Пат. 4.974.143 США, НКл 364/200.
58. Пат. 4.845.667 США, НКл 364/900.
59. Пат. 4.803.617 США, НКл 364/200.
60. Международная заявка PCT/WO/87/04826, МКИ G06F 13/18.
61. Европейский патент EP 0.270.882 A2, МКИ G06F 15/16.
62. Пат. 2.181.577 А, Великобритания, МКИ G06F 13/14.
63. Европейский патент EP 0.339.782 A2, МКИ G06F 13/36.
64. Пат. 1.508.854, Великобритания, НКл G4A.
65. Шевкопяс Б. В. Микропроцессорные структуры. Инженерные решения: Справочник.— 2-е изд., перераб. и доп.— М.: Радио и связь, 1990.— 512 с.
66. Пат. 2.206.468 А, Великобритания, МКИ G06F 13/36.
67. Пат. 2.190.220 А, Великобритания, МКИ G06F 12/08.
68. Пат. 4.858.111 США, НКл 364/200.
69. Европейский патент EP 0.075.633 B1, МКИ G06F 9/46.
70. Пат. 2.176.919 А, Великобритания, МКИ G06F 12/02.
71. Европейский патент EP 0.026.267 B1, МКИ G06F 5/00.
72. Пат. 2.235.995 А, Великобритания, МКИ G06F 13/42.
73. Европейский патент EP 0.362.691 A2, МКИ G06F 1/10.
74. Европейский патент EP 0.307.793 A2, МКИ G06F 13/40.
75. Европейский патент EP 0.341.841 A2, МКИ G06F 13/40.
76. Пат. 4.922.449 США, НКл 364/900.
77. Пат. 4.823.309 США, НКл 364/900.
78. Пат. 4.789.959 США, НКл 364/900.
79. Европейский патент EP 0.285.892 A2, МКИ G06F 9/38.
80. Пат. 4.819.201 США, НКл 364/900.
81. Европейский патент EP 0.286.356 A2, МКИ G06F 7/00.
82. Пат. 2.213.299 А, Великобритания, МКИ G06F 7/04.
83. Пат. 2.194.365 А, Великобритания, МКИ G06F 7/02.
84. Европейский патент EP 0.136.168 A2, МКИ G06F 11/00.
85. Европейский патент EP 0.332.845 A2, МКИ G06F 7/00.
86. Европейский патент EP 0.308.061 A2, МКИ G06F 7/00.
87. Европейский патент EP 0.340.694 A2, МКИ G06F 11/26.
88. Европейский патент EP 0.320.329 A2, МКИ G06F 11/34.
89. Пат. 4.905.187 США, НКл 364/900.
90. Пат. 4.931.986 США, НКл 364/900.
91. Пат. 4.809.171 США, НКл 364/200.
92. Electronic Design.— Vol. 36, № 24, October 27, 1988, p. 136.
93. Международная заявка PCT/WO/85/05203, МКИ G06F 7/10.
94. Международная заявка PCT/WO/87/07968, МКИ G06F 11/26.
95. Европейский патент EP 0.425.070 A2, МКИ G06F 11/00.
96. Европейский патент EP 0.379.770 A2, МКИ G06F 11/10.
97. Европейский патент EP 0.436.891 A2, МКИ G06F 13/40.
98. Европейский патент EP 0.429.787 A2, МКИ G06F 13/42.
99. Европейский патент EP 0.402.055 A2, МКИ G06F 13/40.
100. Европейский патент EP 0.350.573 A1, МКИ G06F 13/40.

## Оглавление

Предисловие .....	3
Глава 1. Взаимодействие микроЭВМ с объектами .....	4
1.1. Опрос резисторного датчика температуры и определение положения движка переменного резистора с использованием порта ввода — вывода микроЭВМ .....	4
1.2. Упрощение схемы связи микроЭВМ с клавиатурой и светодиодными индикаторами .....	5
1.3. Идентификация нажатых клавиш с использованием преобразователя сопротивление — частота .....	6
1.4. Схема дешифрации кода, поступающего от источника импульсов инфракрасного излучения, размещенного на дистанционном пульте управления бытовым прибором .....	7
1.5. Метод и аппаратура кодирования — декодирования цифровой информации при ее записи — считывании с бытового кассетного магнитофона .....	8
1.6. Коммутатор линий интерфейса RS-232 .....	11
1.7. Увеличение числа информационных входов и выходов микроЭВМ подключением внешних смывовых и параллельных регистров ...	12
1.8. Повышение надежности передачи команды Сброс от главной микроЭВМ к подчиненной по последовательному каналу связи ...	16
1.9. Обмен данными между двумя удаленными друг от друга микроЭВМ по двухпроводной линии связи .....	18
1.10. МикроЭВМ с групповым обслуживанием контроллеров .....	19
1.11. Подключение группы контроллеров к порту ввода — вывода микроЭВМ .....	21
1.12. Порт ввода — вывода с защитой от электрических перегрузок ..	27
1.13. Поиск источника прерывания преобразованием суммарного потенциального запросного сигнала в последовательность импульсных сигналов .....	28
1.14. МикроЭВМ с распределенным арбитром запросов прерываний ..	31
1.15. Уменьшение аппаратуры обслуживания запросов прерываний .....	35
1.16. Обработка прерываний в многопроцессорной системе с индивидуальными блоками кэш-памяти .....	37
1.17. Упрощение аппаратуры слежения за временем выполнения заданий одноклассными устройствами .....	40
1.18. Подключение пользовательских контроллеров к ПЭВМ типа IBM PC (PS) с выделением им «не освоенных» фирмой IBM областей адресного пространства ввода — вывода .....	43
Глава 2. Контроль, диагностика, восстановление, поиск информации .....	47
2.1. Схемы выходных каскадов мажоритарно-резервированных устройств .....	47
2.2. Самоконтролируемый выходной каскад мажоритарно-резервированного устройства .....	51

2.3. Схема обнаружения и исправления ошибок в дублированной системе связи с датчиками .....	53
2.4. Обнаружение ошибок при последовательной передаче информации между двумя устройствами .....	55
2.5. Передача тестовой информации для проверки составных частей микроЭВМ в обход внутренней магистрали через последовательный канал связи .....	58
2.6. Схема контроля частоты сигналов в различных точках цифрового устройства .....	63
2.7. Ускоренное формирование контрольных разрядов четности результата суммирования двух чисел .....	65
2.8. Дублированное ОЗУ с коррекцией одиночных ошибок .....	66
2.9. Повышение корректирующих возможностей системы исправления ошибок, использующей код Хэмминга .....	68
2.10. Контроль динамического ОЗУ, совмещенный с выполнением циклов его регенерации .....	72
2.11. Уменьшение энергии, потребляемой статическим ОЗУ с контролем по четности от резервной батареи в отсутствие штатного напряжения питания .....	74
2.12. Вычислительное устройство со «сквозным» контролем по четности .....	76
2.13. Контролируемая схема сопряжения порта микроЭВМ с нагрузочными резисторами .....	80
2.14. Пример рационального использования разрядов порта ввода—вывода микроЭВМ в управляющей системе с самоконтролем ..	82
2.15. Контроль правильности формирования и обработки сигналов прерывания .....	84
2.16. Использование диагностического процессора для оперативного отключения отказавших блоков от магистрали микроЭВМ .....	85
2.17. Быстрое выявление обращений процессора по «несуществующим» адресам .....	87
2.18. Устранение ошибок адресации, вызывающих беспорядочное «блуждание» процессора по памяти .....	89
2.19. Схема контроля длительности выполнения программного цикла, целостности линий магистрали и порядка следования выполняемых фрагментов программы .....	89
2.20. Межпроцессорный обмен данными с обнаружением ошибок формирования ответных сигналов .....	93
2.21. Точный или приближенный поиск эталонного кода в потоке последовательных входных данных .....	99
2.22. Двумерная решетчатая структура для точного или приближенного распознавания символического сообщения .....	102
2.23. Схема контроля дешифратора .....	107
2.24. Самоконтролируемая схема проверки нечетности числа единиц в передаваемых кодах .....	108
2.25. Защита от неправильного подключения плат контроллеров к системной плате микроЭВМ .....	111
<b>Глава 3. Адресация, обмен данными .....</b>	<b>112</b>
3.1. Способ расширения памяти микроЭВМ .....	112
3.2. Расширение ПЗУ без увеличения выделенного ему адресного пространства .....	117
3.3. Адресация блоков памяти, имеющих разное быстродействие .....	119
3.4. Адресное взаимодействие устройств и передача запросов прерываний по параллельной информационной шине и двум управляющим линиям .....	120
3.5. МикроЭВМ с расширенными возможностями адресации .....	125
	251

3.6. Подключение к микроЭВМ плат памяти одинакового или разного объема с автоматическим перераспределением адресного пространства .....	130
3.7. Адресный дешифратор, обслуживающий блоки памяти разного объема .....	132
3.8. Повышение быстродействия динамического ОЗУ путем «предсказания» номера блока памяти, к которому ожидается обращение .....	136
3.9. Определение типа платы, установленной в разъем, без использования дополнительных контактов этого разъема .....	140
3.10. Изменение функций, выполняемых системой из нескольких микроЭВМ, заменой ППЗУ в одной из этих микроЭВМ .....	141
3.11. МикроЭВМ с многоплатной «этажерочной» конструкцией и автоматическим назначением собственных адресов плат .....	142
3.12. Повышение эффективности схем опроса постоянной информации, заданной переключками на плате .....	146
3.13. Способ адресации контроллеров .....	149
3.14. Быстрое сохранение—восстановление содержимого регистров процессора при прерываниях программы .....	151
3.15. Схема сопряжения между двумя микроЭВМ с использованием параллельных портов ввода—вывода .....	154
3.16. Повышение производительности многопроцессорной системы при замене общей двунаправленной шины двумя однонаправленными ...	157
3.17. Бесконфликтный обмен данными в многопроцессорной системе с распределенным управлением .....	159
3.18. Арбитраж магистрали с предоставлением права исключительного пользования ею устройствам, проявляющим повышенную активность .....	161
3.19. Распределенный арбитраж магистрали многомашинной системы с переменными приоритетами микроЭВМ .....	164
3.20. Устранение конфликтных ситуаций в многомашинной системе ..	166
3.21. Структура блоков кэш-памяти в многопроцессорной системе с общим ОЗУ .....	169
3.22. Повышение производительности микропроцессорной системы с кэш-памятью .....	172
3.23. Схема быстрого выделения ресурсов по запросам текущих заданий .....	177
3.24. Схема учета и поиска свободных устройств .....	180
3.25. Сжатие графической информации при ее передаче по последовательному каналу связи .....	185
3.26. Ускорение обмена данными между двумя микроЭВМ с раздельной синхронизацией .....	188
3.27. Двусторонний обмен данными между главной и подчиненной микроЭВМ по общей информационной шине .....	191
<b>Глава 4. Аппаратная поддержка вычислений, узлы микроЭВМ</b> .....	193
4.1. Выравнивание задержек каналов усиления синхросигналов .....	193
4.2. Уменьшение потерь времени при формировании сигнала подтверждения истинности кода, выдаваемого в общую шину .....	194
4.3. Повышение пропускной способности магистрали с третьим состоянием .....	198
4.4. Быстродействующая КМОП-магистраль .....	199
4.5. Выходной каскад КМОП-элемента, обеспечивающий низкое потребление энергии при сопряжении с входами TTL- и КМОП-элементов .....	200
4.6. Предотвращение гонок при опросе счетчика .....	202
4.7. Предотвращение гонок в конвейерных схемах .....	204
4.8. Буфер типа FIFO на регистрах-защелках .....	208
4.9. Быстродействующий буфер типа LIFO .....	212

4.10. Упрощение схемы сравнения кодов заменой элементов Исключающее ИЛИ элементами ИЛИ—НЕ .....	213
4.11. Компаратор на основе ПЛИМ .....	216
4.12. Схемы параллельного сравнения кода с несколькими эталонами .....	218
4.13. Усовершенствованные схемы генераторов кода маски .....	222
4.14. Усовершенствованные генераторы псевдослучайных чисел .....	229
4.15. Контроль степени загрузки процессора полезной работой .....	230
4.16. Уменьшение энергопотребления однокристальной микроЭВМ, выполняющей в отсутствие штатного напряжения питания функции часов с питанием от резервной батареи .....	231
4.17. Многоканальный тактовый генератор .....	236
4.18. Многопроцессорная система для параллельной обработки операндов .....	238
4.19. Генератор импульсов с малым потреблением энергии .....	241
4.20. Схема расширения импульсов .....	242
4.21. Двоичный счетчик со схемой контроля .....	244
4.22. Конструктивные особенности печатных плат, улучшающие эксплуатационные характеристики одноплатных устройств .....	246
Список литературы .....	248

**Шевкопляс Б. В.**

**Ш37** Микропроцессорные структуры. Инженерные решения. Дополнение первое: Справочник.— М.: Радио и связь, 1993.— 256 с.: ил.

**ISBN 5-256-00973-7.**

Рассмотрены практические вопросы проектирования микропроцессорных устройств и систем на их основе. Приведены схемы сопряжения микроЭВМ с объектами, даны примеры построения устройств для контроля, диагностики, восстановления и поиска информации, описаны методы и схемы расширения оперативной памяти, адресного взаимодействия устройств и т. п. Всего приводится свыше ста оригинальных структурных и схемных решений, признанных изобретениями.

Для инженерно-технических работников, занимающихся вопросами проектирования, производства и эксплуатации средств вычислительной техники.

**Ш** 2302037000-036 13-93  
046(01)-93

**ББК 32.97**

Справочное издание  
ШЕВКОПЛЯС БОРИС ВЛАДИМИРОВИЧ  
**МИКРОПРОЦЕССОРНЫЕ СТРУКТУРЫ  
ИНЖЕНЕРНЫЕ РЕШЕНИЯ**  
ДОПОЛНЕНИЕ ПЕРВОЕ

Справочник

Заведующий редакцией Ю. Н. Рысев  
Редактор М. М. Лисина  
Обложка художника В. Н. Ожигова  
Художественный редактор В. И. Мусяненко  
Технический редактор Л. А. Горшкова  
Корректор Т. В. Дземидович

**ИБ № 2466**

Сдано в набор 12.11.92. Подписано в печать **20.04.93**  
Формат 60x88/16. Бумага типографская № 2. Гарнитура таймс. Печать офсетная.  
Усл. печ. л. 15,68. Усл. кр.-отт. 15,93. Уч.-изд. л. 21,26.  
Тираж 20 000 экз. Изд. № 23414. Зак. № **214** С-036.

Издательство "Радио и связь", 101000, Москва, Почтамт, а/я 693.

Набрано в Государственном Ордена Октябрьской Революции, Ордена Трудового  
Красного Знамени, Московском предприятии "Первая Образцовая типография"  
Министерства печати и информации РФ. 113054, Москва, Валовая, 28.  
Отпечатано с готовых диапозитивов в Московской типографии № 4 Министерства  
печати и информации РФ. 129041, Москва, Б. Переяславская, 46.

# ПРЕДЛАГАЕМ

организациям, предприятиям,  
кооперативам  
и совместным предприятиям!

Публиковать текстовую рекламную информацию о разработках Вашей отрасли, изделиях Ваших предприятий в книгах нашего издательства.

*Текст для публикации должен быть отпечатан в двух экземплярах. Желательно, чтобы объем материала не превышал одной машинописной страницы.*

Цена договорная.

Срок публикации до трех месяцев.

**В сопроводительном письме надо указать: гарантии оплаты за публикацию, номер Вашего расчетного счета и отделение Гюсбанка.**

НАШ АДРЕС: 101000, Москва, ул. Мясницкая, 40,

ИЗДАТЕЛЬСТВО «РАДИО И СВЯЗЬ»

т е л е ф о н 923-49-04

