Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Промышленная электроника»

Э. М. Виноградов

ПРОЕКТИРОВАНИЕ МИКРОКОНТРОЛЛЕРНОЙ СИСТЕМЫ УПРАВЛЕНИЯ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к курсовому проекту
по дисциплине «Микропроцессорная техника»
для студентов специальности 1-36 04 02
«Промышленная электроника»
дневной и заочной форм обучения

Гомель 2009

УДК 621.58(075.8)
ББК 30.973.26-04я73
В49

Рекомендовано научно-методическим советом
факультета автоматизированных и информационных систем
ГГТУ им. П. О. Сухого
(*протокол № 1 от 22.09.2008 г.*)

Рецензент: канд. техн. наук, доц. каф. «Автоматизированный электропривод»
ГГТУ им. П. О. Сухого *С. И. Захаренко*

Виноградов, Э. М.

В49 Проектирование микроконтроллерной системы управления : метод. указания к курсовому проекту по дисциплине «Микропроцессорная техника» для студентов специальности 1-36 04 02 «Промышленная электроника» днев. и заоч. форм обучения / Э. М. Виноградов. – Гомель : ГГТУ им. П. О. Сухого, 2009. – 89 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: http://lib.gstu.local. – Загл. с титул. экрана.

Даны методические рекомендации по выполнению курсового проекта, изложены требования по его оформлению, приведены схемы и программы, а также пример разработки микроконтроллерной системы.

Для студентов специальности 1-36 04 02 «Промышленная электроника» дневной и заочной форм обучения.

УДК 621.58(075.8)
ББК 30.973.26-04я73

Введение

Развитие и распространение однокристальных микроконтроллеров требует квалифицированных специалистов. Они должны, с одной стороны хорошо знать элементную базу цифровых устройств (микроконтроллеров, микросхем различной степени интеграции), с другой стороны, владеть методами алгоритмизации и программирования различных технических задач.

Для закрепления и углубления знаний, приобретенных студентами в курсе лекций по дисциплине “Микропроцессорная техника” программой специальности “Промышленная электроника” предусмотрен курсовой проект.

Целью курсового проекта является приобретение студентами навыков по проектированию и расчету микроконтроллерных систем, осуществляющих сбор данных, их обработку, отображение и преобразование в унифицированные информационные сигналы.

Задача курсового проекта – разработка микроконтроллерной системы управления технологическим объектом, выбор элементов системы, разработка ее программного обеспечения, составление структурных, функциональных и принципиальных схем.

1. Общие принципы проектирования микроконтроллерных систем

Проектирование микроконтроллерной системы (МКС) состоит обычно из трех этапов [1]:

1) выбор элементов и составление структурной, функциональной и принципиальной схем;
2) распределение памяти и присвоение элементам, к которым в процессе выполнения программы необходимо обращаться, соответствующих кодов и адресов;
3) составление и отладка программы.

Основными исходными данными для проектирования являются: описание или схема алгоритма решаемой задачи, характеристики входных и выходных сигналов, требования к точности, быстродействию, потребляемой мощности, надежности. Часто в задании на проектирование содержатся также рекомендации по применению определенной элементной базы. Во многих случаях указанные данные закладываются в схему устройства вначале лишь в общем виде (посред-

ством выбора подходящих элементов), а затем в процессе проектирования проверяется их выполнение и вносятся коррективы.

На первом этапе проектирования выбирается элементная база и, в первую очередь, микроконтроллер (МК). Выбор его определяет, прежде всего, специфика реализуемых в МКС алгоритмов.

Если МКС предназначена для реализации алгоритмов управления и контроля, то важнейшим фактором при выборе МК является чаще всего простота (схемная и программная) сопряжения МКС с объектом управления. Поскольку в системах автоматического управления таковыми обычно являются электромеханические и тепловые устройства, к быстродействию МК предъявляются сравнительно невысокие требования. И в таких системах во многих случаях можно использовать 8-разрядные микроконтроллеры семейства МК51 серий К1816 и К1830, которые позволяют строить достаточно простые и экономичные схемы.

В некоторых случаях решающим при выборе МК является возможность использования имеющегося программного обеспечения, поскольку трудоемкость разработки новых программ намного больше разработки аппаратурной части. С этой точки зрения микроконтроллеры семейства МК51 также весьма эффективны, так как для них имеется обширная литература по программному обеспечению [1,2].

На первом этапе проектирования выбираются также интерфейсные модули, обеспечивающие связь МК с внешними устройствами. Далее производится проверка электрического сопряжения элементов схемы, и при необходимости вводятся усилители, формирователи, преобразователи уровней и т.д. Затем составляются исходные варианты структурных, функциональных и принципиальных схем.

На втором этапе выделяются области памяти программ (ПЗУ) для хранения основной программы, подпрограмм, констант. В памяти данных (ОЗУ) выделяются области для запоминания поступающих из внешних устройств данных, промежуточных результатов, а также организации стека.

На третьем этапе составляется программа работы МКС. При программировании обычно используют язык Ассемблера [1,2]. Широкие возможности использования типовых решений на первом и втором этапах проектирования приводит к тому, что 70…80% трудоемкости проектирования МКС приходится на третий этап.

Эффективным средством разработки программ для МКС является метод декомпозиции (или “нисходящего” проектирования), при кото-

ром вся задача последовательно разделяется на меньшие функциональные модули, каждый из которых можно анализировать, разрабатывать и отлаживать отдельно от других [1]. При выполнении программы в МК управление передается от одного функционального модуля к другому. Схема связи этих функциональных модулей, каждый из которых реализует некоторую завершенную процедуру, образует общую структурную схему (блок-схему) алгоритма (БСА) программы. Это разделение задачи на модули и субмодули выполняется последовательно до такого уровня, когда разработка БСА модуля становится простым (даже тривиальным делом). Язык графических образов БСА можно использовать на любом уровне детализации описания модулей вплоть до того, что каждому блоку БСА будет соответствовать единственная команда МК.

При разработке программного обеспечения для микроконтроллеров могут быть использованы два способа организации прикладных программ: монолитный и модульный. При первом способе вся прикладная программа МК разрабатывается как единое целое, а при втором строится из отдельных программных блоков, каждый из которых реализует некоторую процедуру обработки данных или управления. Взаимосвязь блоков определяется разработчиком при монтаже из этих блоков блок-схемы алгоритма и законченной прикладной программы.

Отдельные фрагменты прикладной программы МК могут быть получены в виде линейной последовательности блоков, другие (многократно используемые) обычно оформляются в виде подпрограмм, к которым прикладная программа, называемая основной, имеет возможность обращаться по мере необходимости. Подпрограмма должна обладать следующими свойствами: выполнять законченную процедуру обработки данных, иметь только один вход и один выход.

Обращение к подпрограмме осуществляется по команде вызова CALL MARK, где MARK – символическое имя процедуры. Имя процедуры используется в качестве метки, отмечающей одну из команд (чаще всего первую) подпрограммы.

Для успешной работы любой подпрограммы необходимо однозначно определить способ передачи в нее исходных данных и способ вывода результатов ее работы. Получили распространение три способа передачи параметров: через память, через регистры общего назначения и через флаги МК. При передаче входных параметров через память основная программа обязательно содержит команды загрузки

некоторых ячеек памяти, а подпрограмма – команды считывания из этих ячеек. При передаче выходных параметров подпрограмма должна загрузить некоторые ячейки памяти, а основная программа – считать. Передача параметров через регистры осуществляется аналогичным образом. В третьем способе используются флаги C и F0, сохраняемые в регистре состояния PSW, либо флаги пользователя (128 флагов) из области прямоадресуемых битов памяти данных (адреса ячеек 20H…2FH).

Использование процедур, оформленных в виде подпрограмм, при разработке программного обеспечения имеет ряд достоинств. Прежде всего, относительно простые модули, выделенные из сложной программы, могут программироваться несколькими разработчиками с целью сокращения времени проектирования. Еще более важным является то, что любая подпрограмма допускает автономную отладку. Это, как правило, многократно сокращает время отладки всего прикладного программного обеспечения. И, наконец, механизм использования подпрограмм, реализующих требуемый набор процедур, уменьшает длину прикладной программы, что приводит к уменьшению требующейся емкости памяти программы. Существенным является и то обстоятельство, что отлаженные процедуры часто организуются разработчиками в библиотеки подпрограмм и могут быть многократно использованы в проектной работе.

2. Задание на проектирование

В курсовом проекте необходимо разработать микроконтроллерную систему (МКС) управления некоторым технологическим объектом. Структура МКС приведена на рис. 2.1. МКС состоит из объекта управления, микроконтроллера, пульта управления и аппаратуры их взаимной связи. Микроконтроллер (МК) путем периодического опроса принимает информацию об объекте от аналоговых и цифровых датчиков. Выходные сигналы датчиков вследствие их различной физической природы могут
потребовать промежуточного преобразования на аналого-цифровых преобразователях (АЦП) или на схемах формирователей сигналов (ФС), которые чаще всего выполняют функции гальванической развязки и формирования уровней двоичных сигналов стандарта ТТЛ.

МК с требуемой периодичностью вырабатывает управляющие сигналы на своих выходных портах. Некоторая часть управляющих сиг-

налов интерпретируется как совокупность простых двоичных сигналов управления, которые через схемы формирователей сигналов (усилители мощности, реле, оптроны и т.п.) поступают на исполнительные устройства. Другая часть управляющих сигналов представляет собой многоразрядные двоичные коды, которые через цифро-аналоговые преобразователи (ЦАП) воздействуют на исполнительные устройства аналогового типа.

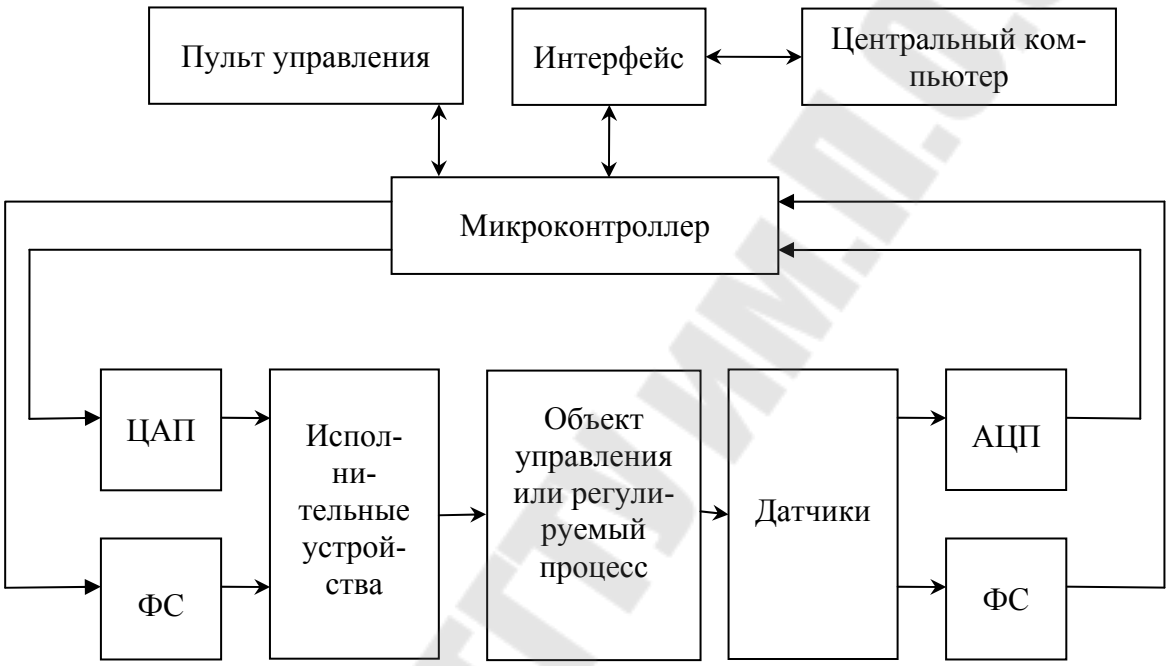


Рис. 2.1. Структура МКС

С помощью пульта управления (ПУ) оператор получает возможность управлять работой МКС и контролировать ее функционирование: запускать МКС, получать значения некоторых сигналов, снимать с индикаторов информацию о состоянии объекта и т.п. С помощью интерфейса МКС может передавать обработанную информацию на центральный компьютер по его запросу.

В курсовом проекте необходимо разработать структурную схему МКС, функциональные и принципиальные схемы отдельных модулей, разработать программы для МК, обеспечивающие выполнение алгоритмов управления и контроля в соответствии с заданием.

2.1. Алгоритм работы МКС

Блок-схема алгоритма работы МКС представлена на рис. 2.2. После включения электропитания или нажатия кнопки “Сброс” на пульте управления (ПУ) выполняется начальная установка (инициализация) системы (блок 1 БСА): вывод начальных значений управляющих воздействий, гашение элементов индикации на ПУ, очистка рабочих ячеек памяти данных МК, настройка последовательного порта и таймеров. Таймер опроса начинает отсчет времени $T_{опр}$.

Блок 2 выполняет задачу логического управления. В нем производится ввод и обработка цифровой информации, поступающей от двоичных датчиков. В блоке 3 производится ввод и обработка аналоговой информации. Далее в блоке 4 разрешаются прерывания от интерфейса. Запрос прерывания поступает по интерфейсу от центрального компьютера. В блоке 5 проверяется положение переключателя режима работы МКС, находящегося на ПУ. Если переключатель в положении “Пульт”, то управление системой передается пульту управления (блок 6 БСА). Если переключатель находится в положении “Работа”, то программа переходит к блоку 7, в котором циклически проверяется флаг окончания счета времени $T_{опр}$. По истечении времени $T_{опр}$ таймер устанавливает флаг окончания счета и запрещается прерывание от интерфейса. Далее программа переходит вновь на выполнение блока 2, т.е. зацикливается.

Блок-схема алгоритма обработки цифровых сигналов приведена на рис. 2.3. Сигналы от двоичных датчиков X1, X2, X3, X4 вводятся в МК и запоминаются в его памяти данных (ПД). Затем вычисляется значение логической (булевой) функции f(X1,X2,X3,X4) в соответствии с выражением, определенном в задании на курсовой проект. Это значение выдается в качестве управляющего сигнала Y1 по соответствующему выходному каналу на исполнительное устройство. При единичном значении логической функции f(X1,X2,X3,X4) МК вырабатывает выходной сигнал ТТЛ - уровня Y1=1 длительностью t1. Форма сигнала Y1 приведена на рис. 2.4

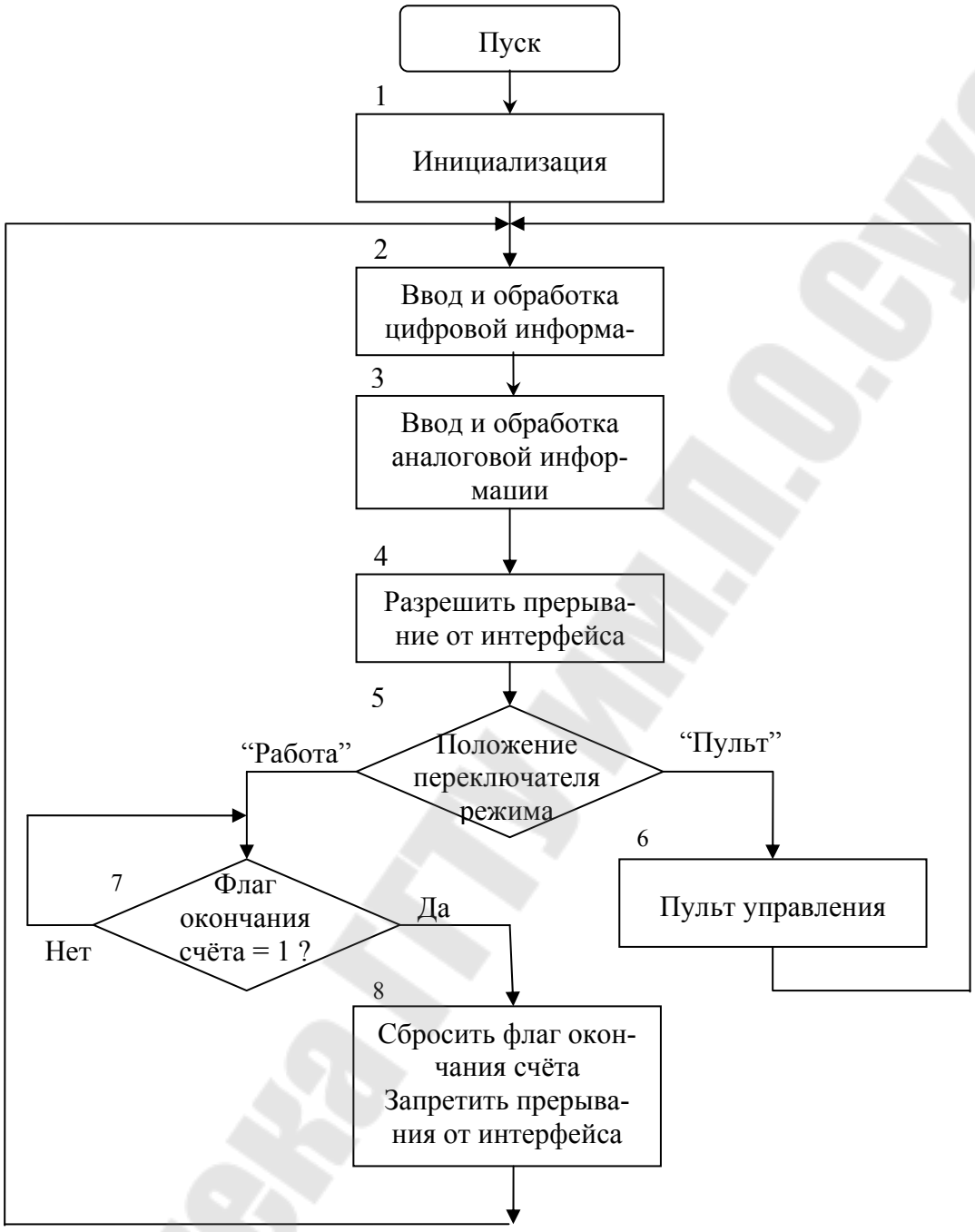


Рис. 2.2. Блок-схема алгоритма работы МКС

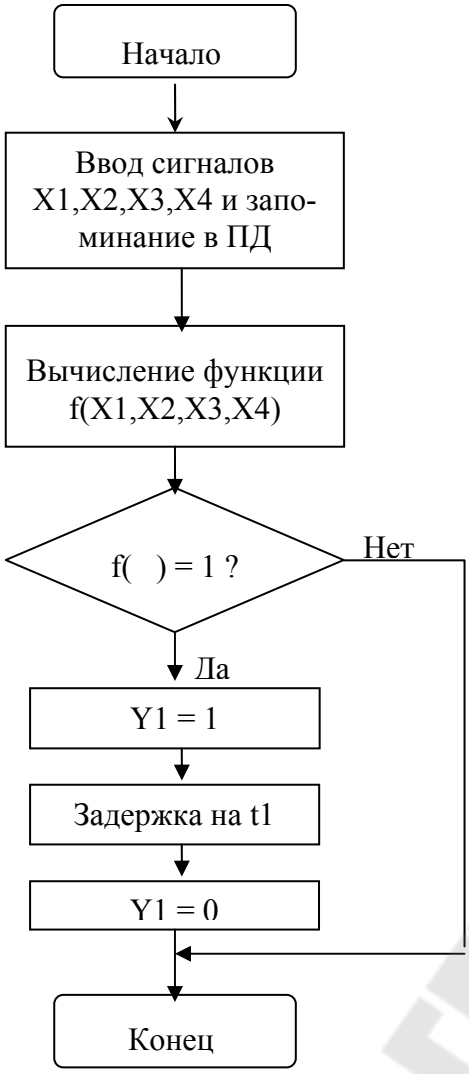


Рис. 2.3. БСА обработки цифровых сигналов

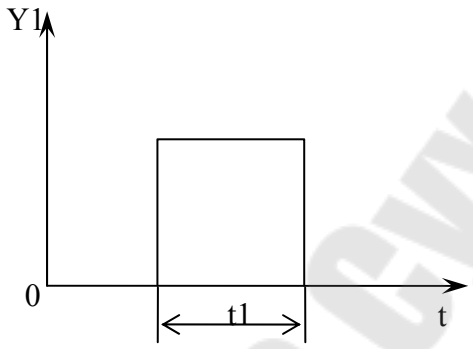


Рис.2.4. Форма сигнала Y1

На рис. 2.5. приведена блок-схема алгоритма обработки аналоговой информации. Сигналы U1, U2, U3, U4 от аналоговых датчиков (однополярное напряжение от 0 до +10 В) преобразуются в цифровую форму с помощью АЦП. В МКС предусматривается только один АЦП, и сигналы с датчиков U1, U2, U3, U4 подключаются к нему последовательно во времени с помощью аналогового коммутатора (мультиплексора). С выхода АЦП n-разрядные коды W1, W2, W3 и W4, представляющие собой целые беззнаковые двоичные числа, поступают на обработку.

Рис. 2.5. БСА обработки аналоговых сигналов

Обработка кодов начинается с вычисления функции g(W1,W2,W3, W4,К1,К2,К3,К4), где коэффициенты К1,К2,К3,К4 – 16-разрядные целые беззнаковые двоичные числа, хранящиеся в ПЗУ. Вид функции g( ) определяется из задания на курсовой проект. Полученное значение функции g( ) сравнивается с константой Q, хранящейся в ПЗУ. В зависимости от результата сравнения МК вырабатывает двоичные управляющие сигналы ТТЛ-уровня Y2 или Y3 длительностью t2 или t3. Форма сигналов Y2 и Y3 аналогична сигналу Y1, изображенному на рис. 2.4.

2.2. Пульт управления

Пульт управления должен содержать следующие элементы:
Светодиоды для индикации значений X1, X2, X3, X4, Y1, Y2, Y3.
1. Четырехпозиционный линейный дисплей на семисегментных индикаторах для отображения аналоговых напряжений U1, U2, U3, U4.
2. Клавишный переключатель для выбора U1, U2, U3, U4.
3.Переключатель “Пульт-работа”, определяющий режим работы МКС. В положении “Работа” система циклически выполняет ввод и обработку информации с датчиков с периодом $T_{опр}$. В этом режиме все элементы индикации на пульте управления погашены. В положении переключателя “Пульт” управление системой переходит к пульту управления, т.е. оператору. В этом режиме выводится информация на индикаторы. Таймер опроса при этом не работает. Для выхода из этого режима переключатель переводится в положение “Работа”, гаснут индикаторы, запускается таймер опроса и программа переходит на ввод и обработку цифровой информации (блок 2 на рис. 2.2.)

3. Методические указания по выполнению курсового проекта

Проектирование микропроцессорной системы рекомендуется начинать с аппаратурной части. Необходимо выбрать схему включения микроконтроллера, разработать схемы ввода сигналов от датчиков, подключения элементов индикации и управления, сопряжения с интерфейсом. Затем следует разработать блок-схемы алгоритмов программ и написать их на языке Ассемблера.

3.1. Схемы подключения микроконтроллера

Для работы микроконтроллеров семейства MCS-51 необходимо:

1) присоединить времязадающую цепь для работы внутреннего тактового генератора;
2) обеспечить сброс при включении электропитания;
3) подключить память программ (внутреннюю или внешнюю).

На рис. 3.1 приведена типовая схема подключения микроконтроллера KM1816BE751, имеющего внутреннюю память программ объемом 4 Кбайт.

Для обеспечения генерации тактовой частоты f_{CLK} к выводам XTAL1 и XTAL2 подключен кварцевый резонатор ZQ1. Частота резонатора f_{ZQ} выбирается из условия $f_{ZQ} = f_{CLK}$ и должна быть в диапазоне 4…12 МГц [1].

Конденсаторы C2,C3 служат для облегчения запуска внутреннего генератора. Цепочка C1,R1 обеспечивает сброс МК при подаче электропитания.

С помощью кнопки SB1, расположенной на пульте управления, сброс МК может выполнить оператор в любой момент времени. На вывод EA подан высокий уровень, что разрешает работу внутренней памяти программ. Конденсатор C4 служит для фильтрации высокочастотных помех, возникающих на выводах источника питания при работе микросхемы.

К выводам порта P0 присоединены “подтягивающие” резисторы R2-R9, которые обеспечивают ток для входов микросхем ТТЛ, подключенных к этому порту, при высоком логическом уровне, когда все линии P0 находятся в z-состоянии. Порты P1,P2,P3 имеют внутренние “подтягивающие” резисторы.

На рис. 3.2 приведена типовая схема подключения МК КР1830ВЕ31, не имеющего внутренней памяти программ. Микросхема DD11 выполняет функции внешней памяти программ объемом 2 Кбайт. Регистр DD10 служит для запоминания (фиксации) младшего байта адреса ячейки памяти. На вывод EA подан низкий уровень для разрешения работы внешней памяти программ. Порт P0 в этой схеме включения работает в динамическом режиме, поэтому “подтягивающие” резисторы на его выводах не нужны [2].

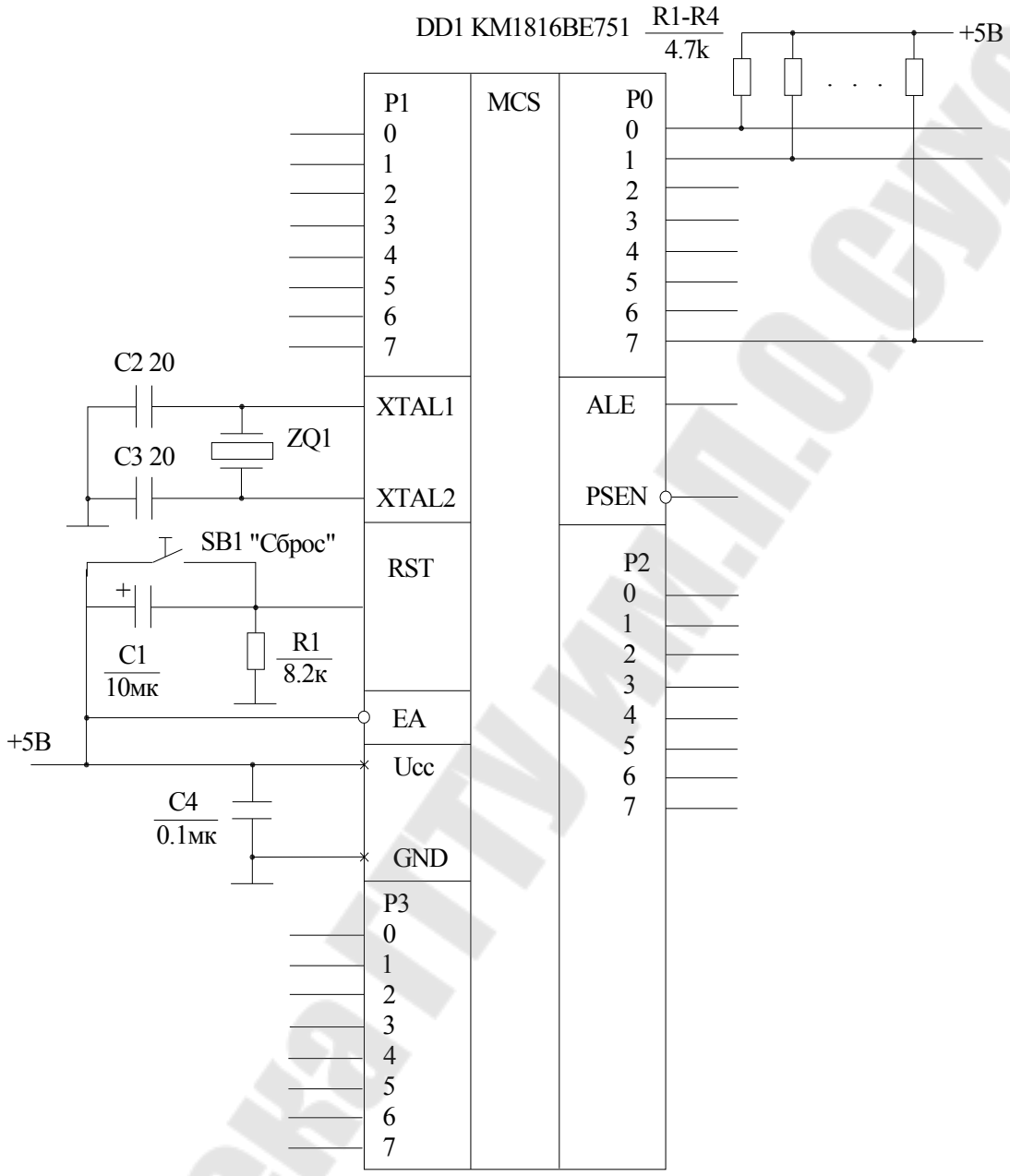


Рис. 3.1. Схема подключения микроконтроллера с внутренней памятью программ

3.2. Увеличение линий ввода-вывода микроконтроллера

При разработки МКС часто возникает необходимость увеличения количества линий ввода-вывода микроконтроллера. Наиболее часто для этих целей используются параллельные регистры микросхем

ТТЛ-серий КР1533 и К555, например, КР1533ИР33, К555ИР22, К555ИР23, КР1533ИР37.

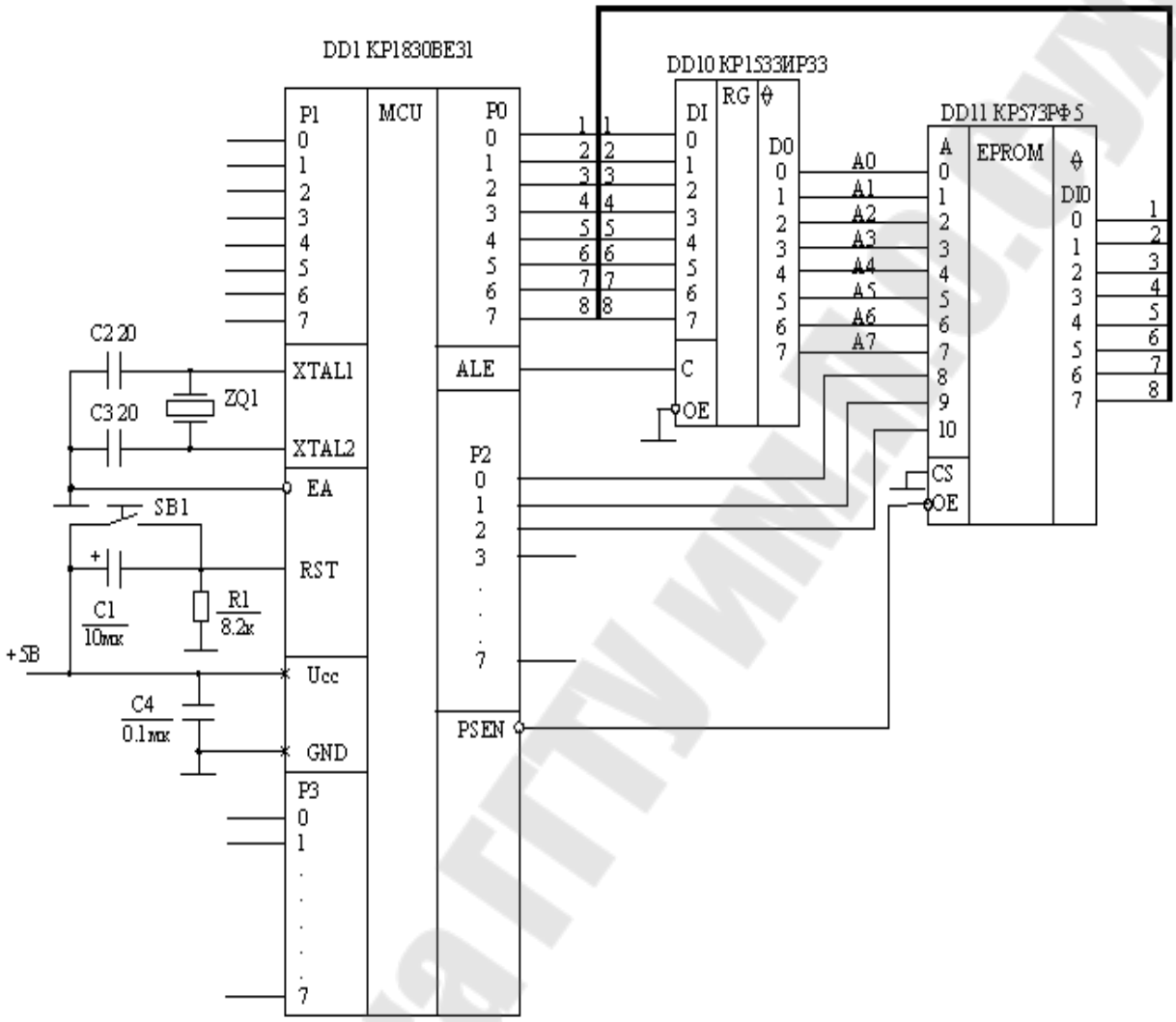


Рис. 3.2. Схема подключения микроконтроллера с внешней памятью программ

Обычно регистры подключаются к порту P0 микроконтроллера, который имеет в 2 раза большую нагрузочную способность, чем порты P1,P2 и P3, а также больше функциональных возможностей [1, 2]. Способ подключения регистров к порту P0 определяется режимом его работы. Порт P0 может работать в одном из двух режимов: статическом и динамическом. При работе в статическом режиме его выходы могут быть в одном из двух состояний: низкого или высокого уровня ТТЛ. В динамическом режиме его выходы могут иметь еще третье

или z-состояние. В статическом режиме порт P0 может работать только в МК, имеющих внутреннюю память программ. В динамическом режиме порт P0 работает в МК с внешней памятью программ или внешней памятью данных.

На рис. 3.3 приведена схема подключения регистров КР1533ИР33 к МК с целью увеличения количества линий ввода-вывода. МК в этой схеме использует внутреннюю память программ. При этом порт P0 работает в статическом режиме, что требует “подтягивающих” резисторов на его выводах. Управление регистрами производится от порта P2. Регистр DD2 осуществляет вывод данных из МК, т.е. он выполняет функцию порта вывода в МКС. При сигнале C=1 триггеры регистра открыты (“прозрачны”) и информация с входов DI поступает на выходы DO, т.е. выполняется вывод данных из МК через порт P0 на внешние устройства системы. При C=0 входы триггеров регистра DD2 закрываются, а информация на выходах DO запоминается (триггеры “защелкиваются”). Для исключения *z*-состояния выходов DO регистра на вывод OE подается низкий уровень. Регистр DD3 осуществляет ввод данных из внешних устройств в МК через порт P0, т.е. он выполняет функцию порта ввода в МКС. На его управляющем входе C постоянно присутствует высокий уровень, поэтому триггеры регистра открыты и принимают информацию с линий ввода. Однако на выходы DO регистра информация поступает через выходные буферы, которые управляются сигналом на выводе OE. При OE=1 выходные буферы DD3 находятся в z-состоянии (закрыты), и выходы DO оказываются отключенными от выводов порта P0 микроконтроллера. Для передачи информации из триггеров DD3 необходимо на OE подать низкий уровень, который откроет выходные буферы регистра.

После сброса МК, например, при включении электропитания, все порты МК настроены на ввод, при этом на их выводах будут высокие уровни (логические 1). Следовательно, в схеме на рис. 3.3 на выходах DD2 будут также логические 1, а выходы DD3 будут в z-состоянии. Обычно в начальном состоянии (после пуска) на линии вывода портов МКС выводятся логические нули. Для этого в программе инициализации МКС следует включить команды:

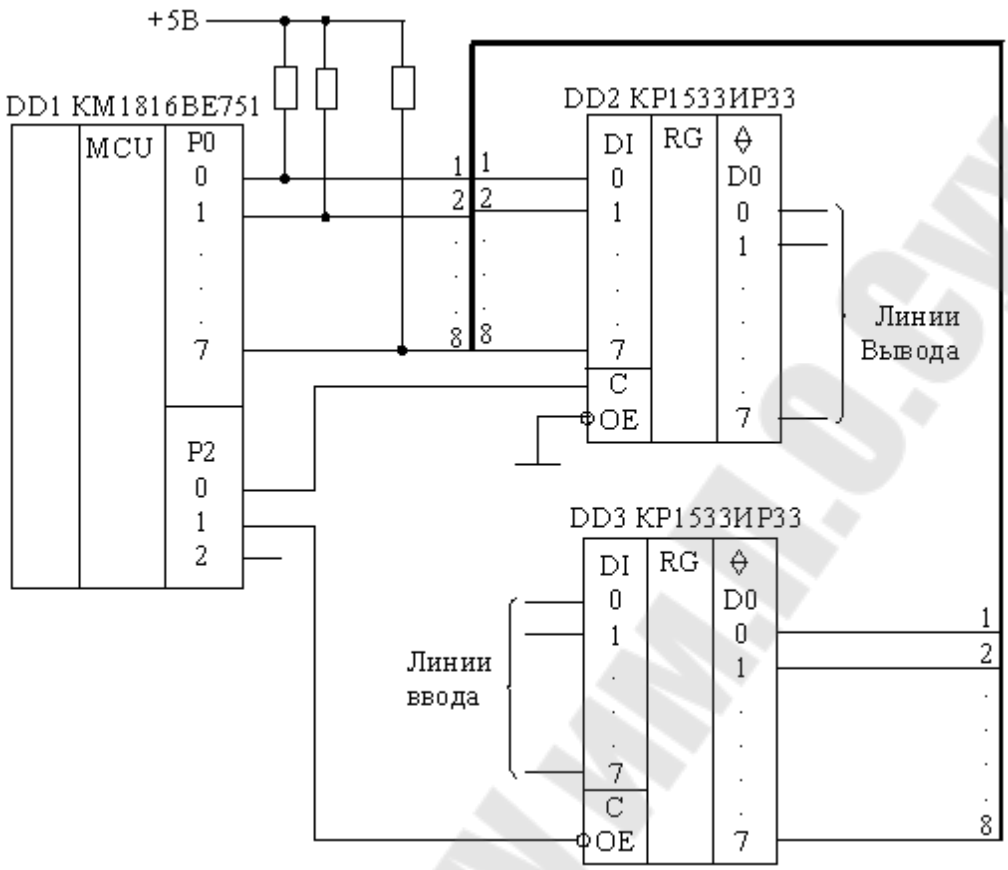


Рис. 3.3. Увеличение линий ввода-вывода МК при статическом режиме работы порта P0

```
MOV  P0,#0   ; Вывод нулей в порт P0 и на выходы DD2
CLR  P2.0    ; Запоминание данных в DD2 (защелкивание триггеров)
```

Для вывода данных, например, из аккумулятора МК в регистр DD2 используются следующие команды:

```
MOV  P0,A    ; Вывод в P0
SETB P2.0    ; Открыть триггеры DD2 (вывод в порт DD2)
CLR  P2.0    ; Запоминание данных в DD2
```

Ввод данных, например, в аккумулятор из регистра DD3 можно выполнить командами:

```
MOV  P0,#0FFH ; Настроить порт P0 на ввод
CLR  P2.1     ; Открыть выходные буферы DD3
MOV  A,P0     ; Пересылка данных из P0 в аккумулятор
SETB P2.1     ; Закрыть выходные буферы DD3 (отключить
              ; выходы DD3 от порта P0)
```

Другой метод подключения регистров для увеличения количества линий ввода-вывода заключается в том, что они адресуются как ячейки внешней памяти данных, доступ к которым осуществляется по командам MOVX.

На рис. 3.4 приведена схема увеличения количества линий ввода-вывода МК при работе порта P0 в динамическом режиме. Регистр DD3 служит портом вывода, с программной точки зрения – это ячейка внешней памяти данных, доступная только для записи. Регистр DD4 служит портом ввода, программно – это ячейка внешней памяти данных, доступная только для чтения. Регистр DD2 служит для запоминания адреса ячеек внешней памяти данных, в данном случае портов ввода-вывода. С его выходов снимаются сигналы разрядов A0,A1,…,A7 адреса. Если МК использует внешнюю память программ, то этот регистр уже имеется в схеме, он используется для запоминания младшего байта адреса ячеек памяти (микросхема DD10 на рис. 3.2). Если же МК работает с внутренней памятью программ, то этот регистр ставится дополнительно.

При выполнении команды записи во внешнюю память данных

MOVX @Ri,A

автоматически на выводе P3.6 микроконтроллера вырабатывается управляющий сигнал (строб) записи WR. Здесь обозначено Ri – это регистры R0 или R1, их содержимое определяет адрес ячейки внешней памяти данных.

При выполнении команды чтения из внешней памяти данных

MOVX A,@Ri

автоматически на выводе P3.7 микроконтроллера вырабатывается управляющий сигнал (строб) чтения RD. Здесь также Ri – это регистры R0 или R1, содержимое которых определяет адрес ячейки памяти.

Для схемы на рис. 3.4 адрес регистра DD3 (порта вывода) как ячейки внешней памяти данных будет 00000001B = 01H, а адрес регистра DD4 (порта ввода) будет 00000010B = 02H.

После сброса МК, например, при включении электропитания на выводе OE микросхемы DD4 будет высокий уровень, поэтому выходные буферы регистра находятся в z-состоянии и выходы DD4 отключены от линий порта P0. На выводе C микросхемы DD3 будет низкий уровень и триггеры регистра закрыты.

Однако на выходах DO регистра после включения питания могут быть случайные значения, поэтому в программе инициализации МКС следует вывести в DD3 нули.

Для вывода данных из МК в регистр DD3 (порт вывода) можно использовать следующие команды:

```
MOV    R0,#01H   ; Адрес DD3
MOVX  @R0,A     ; Вывод в порт DD3 из аккумулятора
```

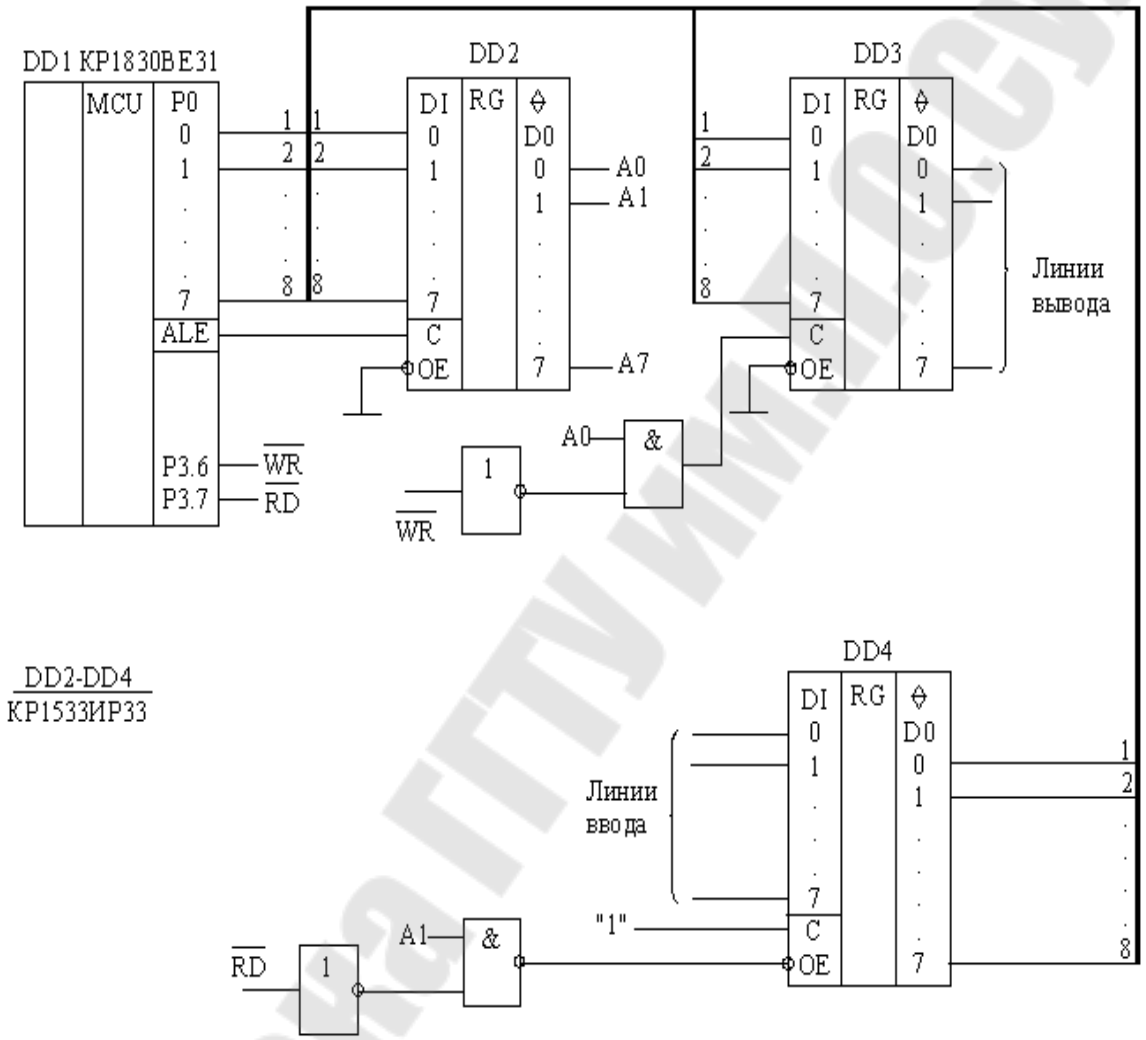


Рис. 3.4. Увеличение линий ввода-вывода МК при динамическом режиме работы порта P0

Для ввода данных в МК из регистра DD4 можно использовать команды:

```
MOV    R0,#02H   ; Адрес DD4
MOVX  A,@R0     ; Ввод из порта DD4 в аккумулятор
```

3.3. Схемы ввода цифровых и аналоговых сигналов

Реализация схем ввода сигналов от датчика МКС зависит от их количества, а также типа заданного АЦП. При разработке схемы подключения АЦП следует учитывать разрядность кода и возможность управления выходными буферами микросхемы.

На рис. 3.5 приведена функциональная схема ввода цифровых и аналоговых сигналов при использовании АЦП типа К1113ПВ1. Микросхема DA1 КР590КН3 является коммутатором аналоговых входных сигналов U1-U4. Переключение входных каналов коммутатора выполняется цифровыми сигналами уровней ТТЛ, подаваемых на адресные входы A1 и A0. Таблица истинности коммутатора имеет вид:

A1	A0	Соединение
0	0	Y=X1
0	1	Y=X2
1	0	Y=X3
1	1	Y=X4

Операционный усилитель DA2 включен по схеме повторителя и служит для устранения влияния низкоомного входа АЦП на коммутатор.

Особенностью микросхемы К1113ПВ1 является то, что в ней отсутствует возможность управления выходными буферами выдачи цифрового кода [3]. После запуска АЦП выходы буферов будут в z-состоянии. Когда преобразование закончится (сигнал на выводе $\overline{DR}$ станет равным 0), выходы буферов переходят в активный режим и на выводах D0-D9 появится цифровой код, который будет неизменным до нового запуска АЦП. В схеме на рис. 3.5 разряды D0-D7 выходного кода АЦП подаются на входы буферного регистра DD2, а два старших разряда D8,D9 – на входы регистра DD3. Сигнал готовности данных $\overline{DR}$ АЦП и цифровые входные сигналы X1-X4 подаются на входы DD3. Выходы регистров DD2 и DD3 соединены с выводами порта P0 МК. Управление выходными буферами регистров осуществляется от линий P3.3 и P3.4. При P3.3=1, P3.4=1 выходы регистров DD2, DD3 находятся в z-состоянии и они отключены от выводов порта P0. Запуск АЦП выполняется сигналом от линии P1.5. Переключение каналов коммутатора производят сигналы с линий P1.6, P1.7 микроконтроллера.

3.4. Схемы вывода управляющих сигналов

В проектируемой МКС имеются 3 управляющих сигнала Y1, Y2, Y3. Эти сигналы представляют собой одиночные импульсы ТТЛ - уровней, поступающие на исполнительные устройства МКС. Управляющие сигналы снимаются с портов МК. Однако следует учитывать, что нагрузочная

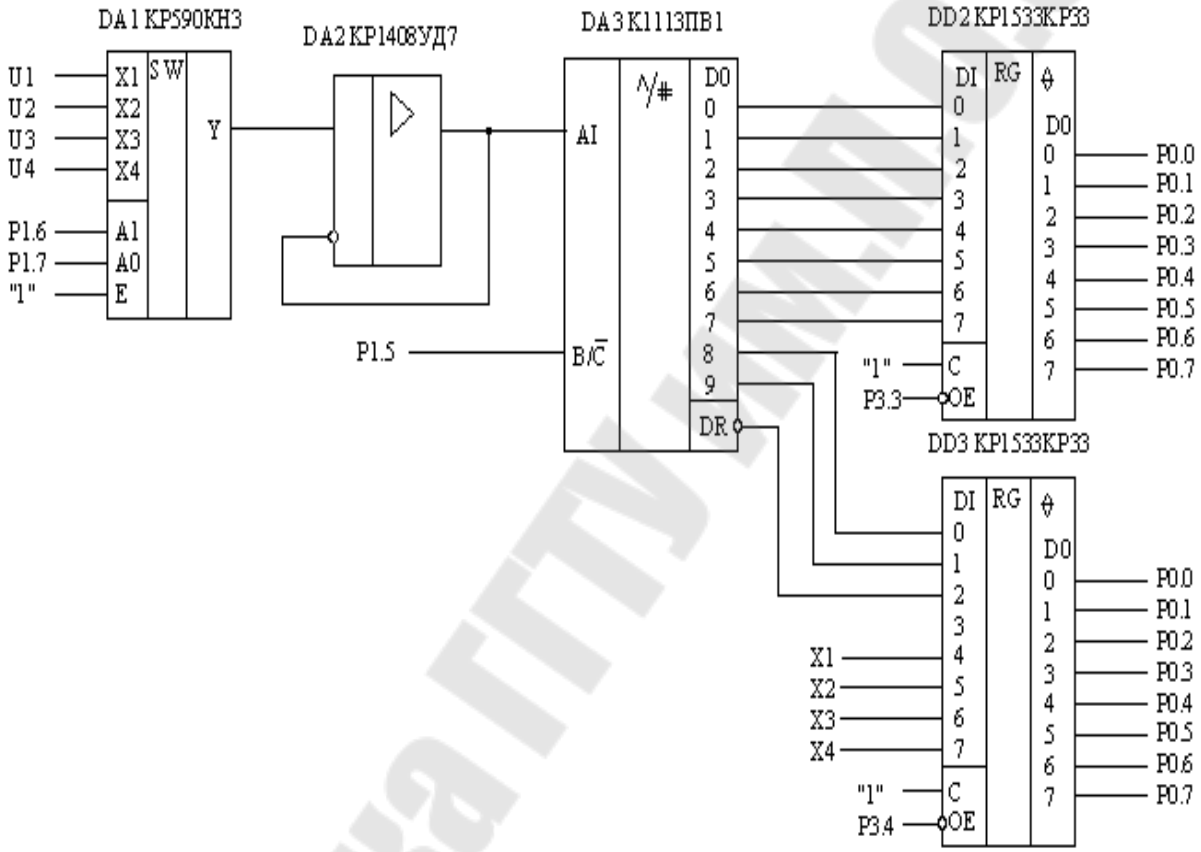


Рис. 3.5. Схема ввода цифровых и аналоговых сигналов с АЦП К1113ПВ1

способность портов Р1, Р2, Р3 мала: $I^0_{вых} \leq$ 1,6 мА, $I^1_{вых} \leq$ 40 мкА [2]. Емкость нагрузки не должна превышать 150 пФ. Для увеличения нагрузочной способности линий портов МК следует использовать усилительные элементы. Это обычно микросхемы, имеющие выходы с открытым коллектором.

На рис. 3.6 приведены схемы формирования управляющих сигналов МКС. В схеме на рис. 3.6,а используется повторитель К155ЛП9. При сигнале на входе уровня логической 1 на выходе будет также 1. В

схеме на рис. 3.6,б используется инвертор К155ЛН3. При сигнале на входе уровня логической 1 на выходе будет логический 0, т. е. такая схема требует инверсных входных сигналов для управления.

В некоторых случаях, при значительном расстоянии исполнительного устройства от МКС, а также исключения влияния помех от наводок на соединительные провода, необходимо гальваническое разделение цепей МК и управляющих сигналов. На рис. 3.8,в приведена схема формирования управляющего сигнала с гальванической развязкой на транзисторной оптопаре АОТ128А. Следует иметь в виду, что источник питания $+U_{П1}$, $-U_{П1}$ должен быть гальванически не связан с источником +5 В питания МК.

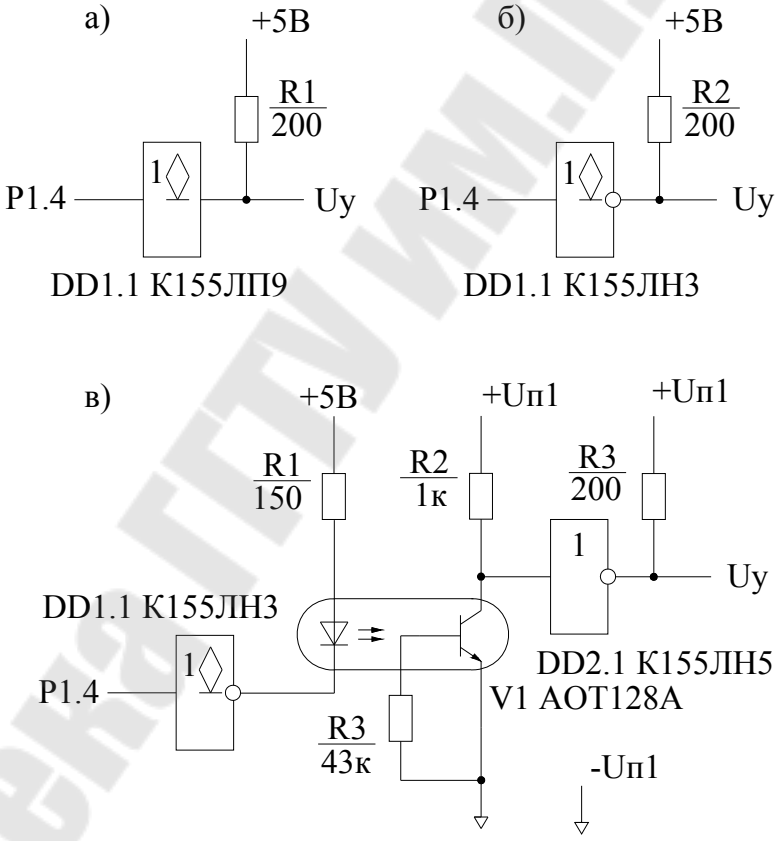


Рис. 3.6. Схемы вывода управляющих сигналов Y1,Y2,Y3

3.5. Схемы подключения светодиодных индикаторов

В проектируемой МКС на пульте управления имеются светодиоды для индикации значений входных сигналов X1, X2, X3, X4 и управляющих Y1, Y2, Y3. На рис. 3.7 приведены два варианта схем подключения светодиодов индикации. В схеме на рис. 3.7,а для управления светодиодами используются мощные инверторы с открытым кол-

лектором К155ЛН3. Светодиод загорается при поступлении логической 1 на соответствующий вход регистра DD5. Управление регистрами осуществляется по входу С. Если С = 1, то триггеры регистра “прозрачны”, и их выходы повторяют сигналы на входах. При С = 0 триггеры “защелкиваются”, т. е. запоминают состояния входов в этот момент. После этого сигналы на входах регистра DD5 могут иметь произвольное значение. Это обстоятельство позволяет использовать порт Р0 для выполнения других функций, например, для ввода данных из АЦП.

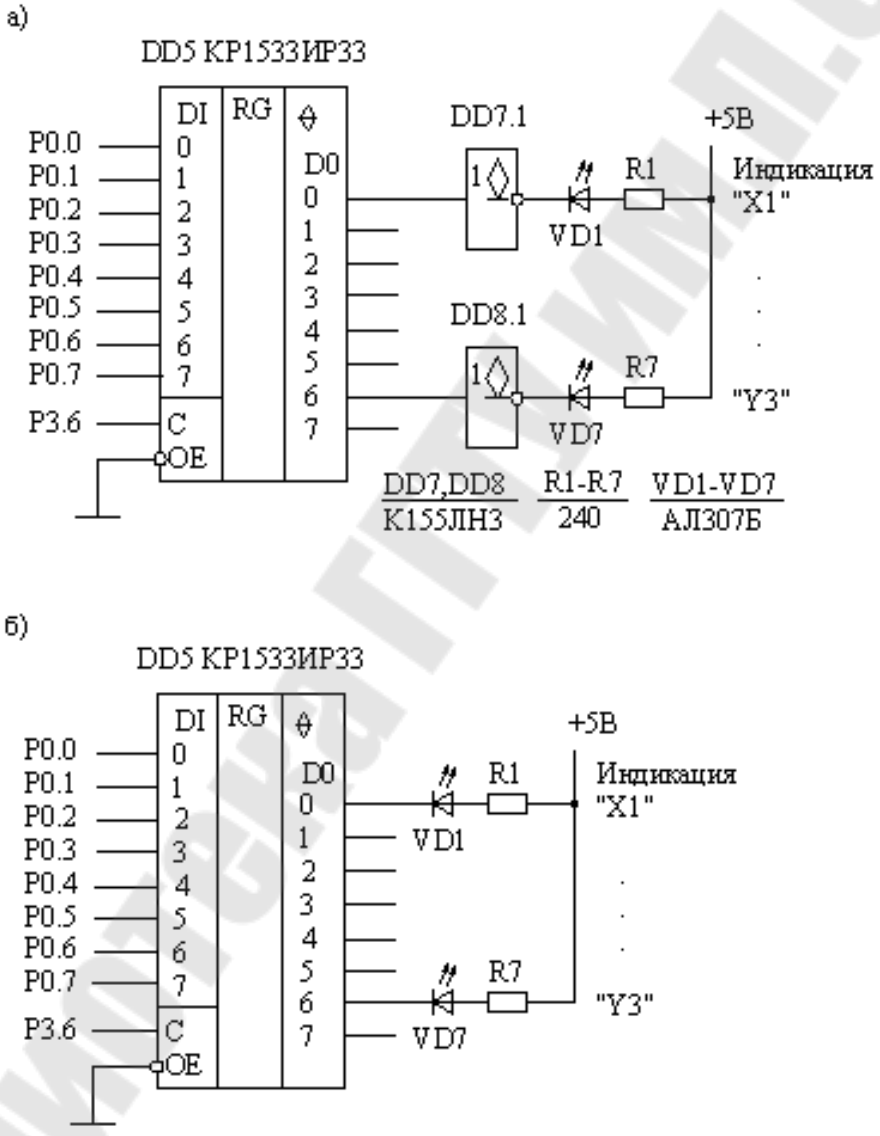


Рис. 3.7. Схемы подключения светодиодов индикации

Регистр КР1533ИР33 имеет довольно мощные выходы: ток низкого логического уровня $I^0_{вых}$: 20 мА, а ток высокого уровня $I^1_{вых}$: 100 мкА [4,5]. Поэтому можно непосредственно управлять светодиодами без использования инверторов, как показано на рис.3.7,б. Светодиоды будут гореть при низком уровне на выходах регистра. Поэтому на входы регистра необходимо подавать также сигналы низкого уровня, т.е. логические 0. Таким образом, схема на рис. 3.7,б требует для управления светодиодами инверсных сигналов, которые должен выдавать порт P0.

3.6. Схемы подключения линейного дисплея и клавиатуры

Семисегментные индикаторы (ССИ) широко используются для отображения цифровой и буквенной информации. Семь отображающих элементов ССИ позволяют высвечивать десятичные и шестнадцатиричные цифры, некоторые буквы русского и латинского алфавитов, а также некоторые специальные знаки. Наиболее распространены индикаторы: светодиодные, вакуумно-люминесцентные, жидкокристаллические, вакуумно-накаливаемые.

Преобразование двоичных позиционных кодов в семисегментные для ССИ может осуществляться либо программно, либо аппаратно с использованием преобразователей кодов.

Для отображения многосимвольной индикации используются линейные (однострочные) дисплеи. Такие дисплеи представляют собой “линейку”, смонтированную из отдельных ССИ. Число знакомест (позиций или разрядов) дисплея определяется в соответствии с требованиями к МКС.

Существует два способа организации МК с линейными дисплеями: статический и динамический [1].

Первый требует наличия на входах каждого индикатора специальных буферных регистров для хранения кодов выводимых символов. Естественно, что с увеличением разрядности дисплея возрастает число дополнительных микросхем, а, следовательно, и стоимости МКС.

Второй способ (динамический) основан на том, что любой световой индикатор является инерционным прибором, а человеческому глазу отображаемая на дисплее информация, если ее обновлять с частотой не менее 20 раз в секунду, представляется неизменной. Динамический способ вывода информации на дисплей требует значитель-

но меньших аппаратных затрат, но более сложного программного обеспечения.

Второй способ (динамический) основан на том, что любой световой индикатор является инерционным прибором, а человеческому глазу отображаемая на дисплее информация, если ее обновлять с частотой не менее 20 раз в секунду, представляется неизменной. Динамический способ вывода информации на дисплей требует значительно меньших аппаратных затрат, но более сложного программного обеспечения.

На рис. 3.8 приведена схема подключения четырехразрядного дисплея на семисегментных светодиодных индикаторах при статическом управлении. Дисплей предназначен для отображения только десятичных цифр, поэтому в качестве преобразователей двоичного кода в семисегментный используются дешифраторы КР514ИД2. Индикаторы АЛС333Б имеют раздельные катоды [8]. Индикатор HG1 отображает младшую десятичную цифру, а HG4 – старшую. Для высвечивания запятой сегмент h индикатора HG3 соединен через резистор R22 с общим проводом. Микросхема DD6 является буферным регистром для хранения кодов, выводимых на индикаторы HG1 и HG2 с порта P0. Коды для индикаторов HG3 и HG4 непосредственно снимаются с порта P2.

В разрабатываемой МКС на пульте управления имеется простейшая клавиатура, состоящая из клавиш для переключения аналоговых сигналов с целью отображения их на индикаторах дисплея, а также переключателя (тумблера) для ввода режима работы МКС. Клавиатуры по методу аппаратурной реализации бывают двух видов: кодирующие и некодирующие [1].

В кодирующих клавиатурах каждый контакт подключается к отдельной линии порта ввода МК. При этом схемным путем формируется код, соответствующий нажатой клавише. Благодаря простоте реализации эти клавиатуры широко применяются при небольшом числе клавиш, как правило, не более 8.

При большом числе клавиш удобнее применять некодирующие (матричные) клавиатуры, которые представляют собой простую матрицу переключателей (требуемой размерности), включенных на пересечении строк и колонок матриц. Идентификация (кодирование) нажатой клавиши в таких клавиатурах выполняется программой. Так как для идентификации применяется метод сканирования, то такие

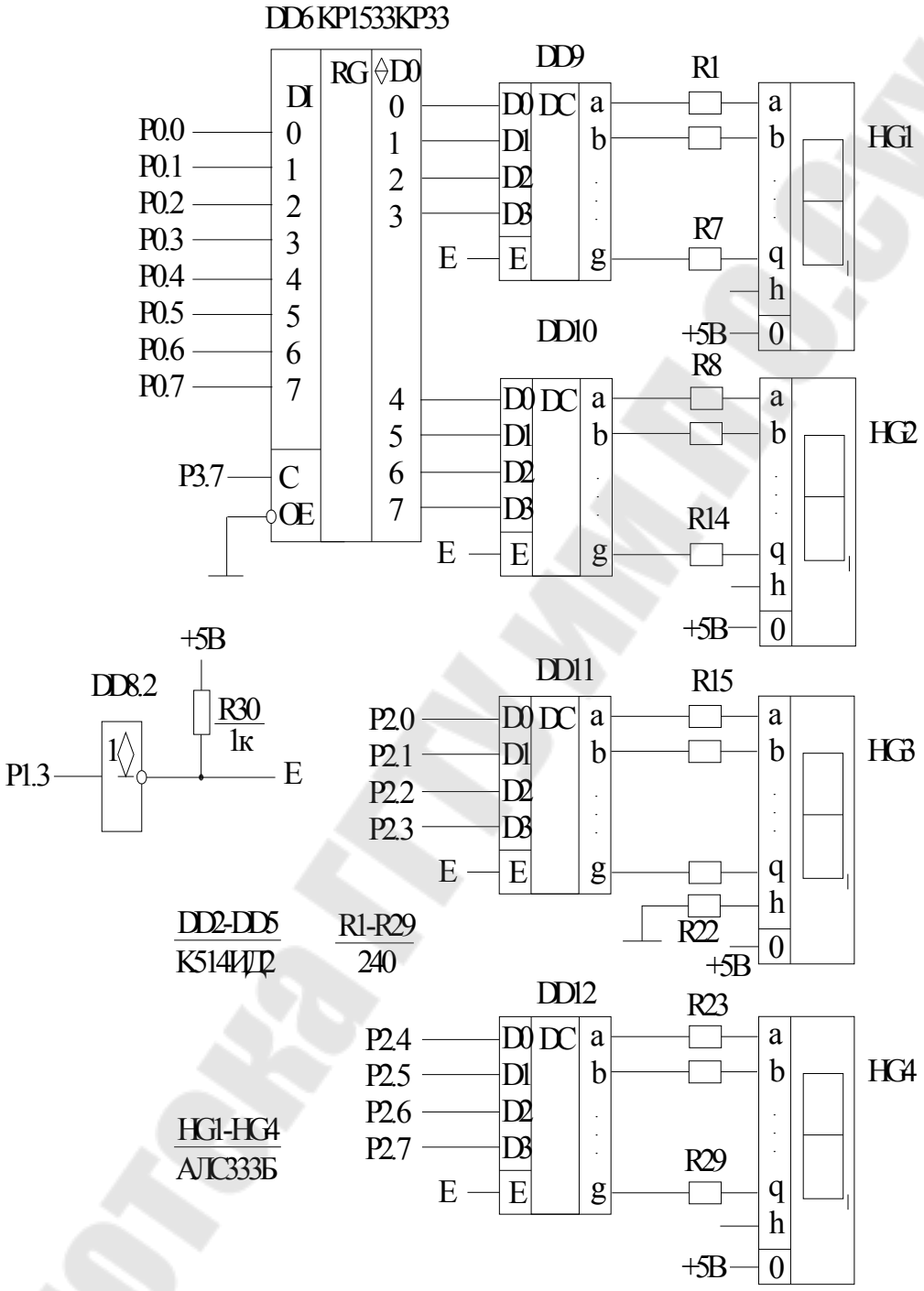


Рис.3.8. Схема подключения семисегментных светодиодных индикаторов при статическом управлении дисплеем

клавиатуры обычно используются совместно с дисплеями, которые используют динамическое управление индикаторами.

Контакты кодирующих клавиатур (переключателей и кнопок) бывают с фиксацией замкнутого состояния и без фиксации. Первые остаются в нажатом состоянии (замкнуты), вторые размыкаются после отжатия. На рис. 3.9 приведена схема подключения кодирующей клавиатуры, состоящей из переключателей с фиксацией, к порту МК. Переключатели SA1-SA4 служат для выбора аналоговых сигналов U1-U4, а переключатель (тумблер) SA5- для выбора режима работы МКС: “Работа-Пульт”. Особенностью
переключателей SA1-SA4 является то, что они взаимозависимы, т.е. в любой момент времени может быть замкнут только один из них. Если свободных линий портов МК недостаточно для подключения клавиатуры, то можно использовать дополнительный регистр, как показано на рис. 3.10.

Если в клавиатуре используется контакты без фиксации (кнопки), то необходимы средства (программные или аппаратурные) для опроса состояния клавиатуры, т.е. определения момента нажатия кнопки. На рис. 3.11 приведена схема подключения клавиатуры с кнопками, в которой формируется сигнал запроса внешнего прерывания МК в случае нажатия любой клавиши. Линия порта P3.2 в схеме используется как вход запроса прерывания $\overline{INT0}$. При разомкнутых контактах SB1-SB4 на входе $\overline{INT0}$=1. При нажатии любой клавиши замыкается контакт и на входе $\overline{INT0}$=0, т.е. поступает запрос прерывания. Подпрограмма обработки запроса прерывания будет производить идентификацию нажатой клавиши. Так как замкнутое состояние нажатой клавиши не фиксируется визуально, то для отображения номера выбранного сигнала U1-U4 необходимо использовать дополнительный индикатор дисплея, который индицирует цифры 1-4, соответствующие сигналам U1-U4.

При большом количестве сигналов (каналов измерения) можно использовать клавиатуру только с одной клавишей. В исходном состоянии (после инициализации) выбирается, например, канал 1, что отображается на индикаторе дисплея. После каждого нажатия номер канала инкрементируется, т.е. увеличивается на 1. При достижении номера последнего канала счет начинается опять с 1.

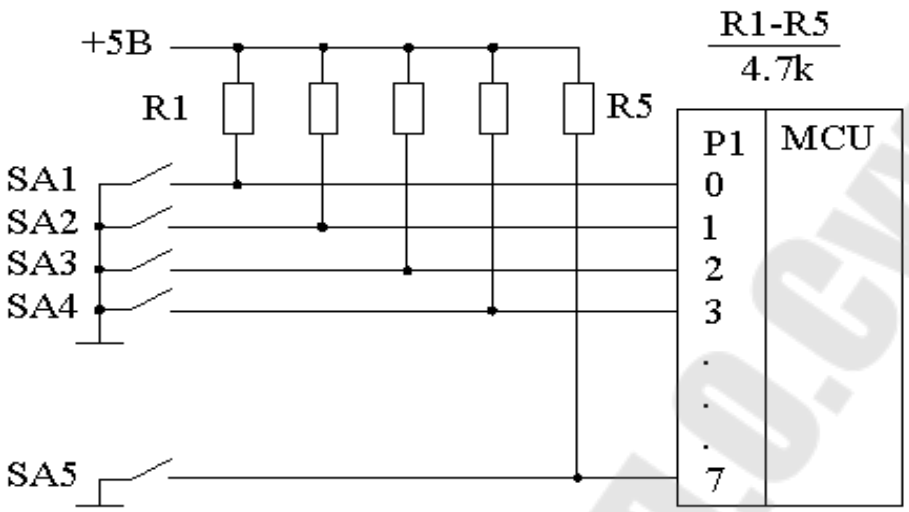


Рис.3.9. Схема подключения переключателей к МК

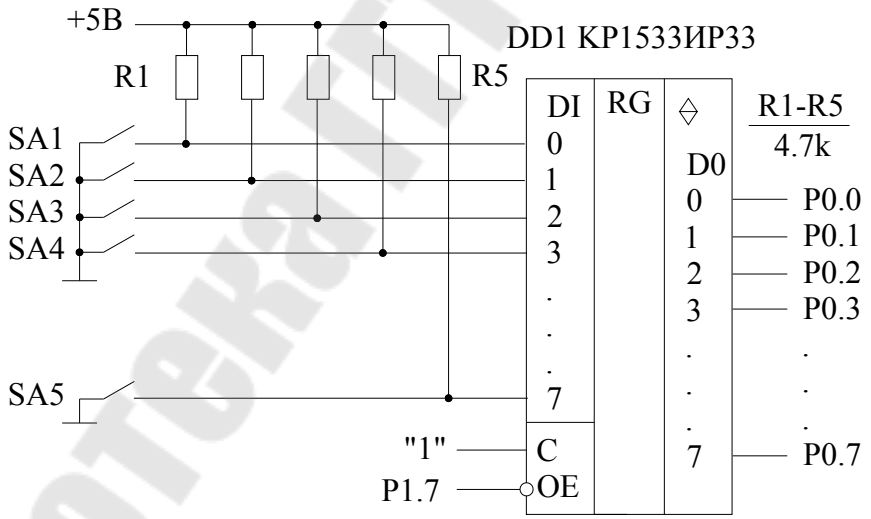


Рис. 3.10. Схема подключения переключателей к регистру

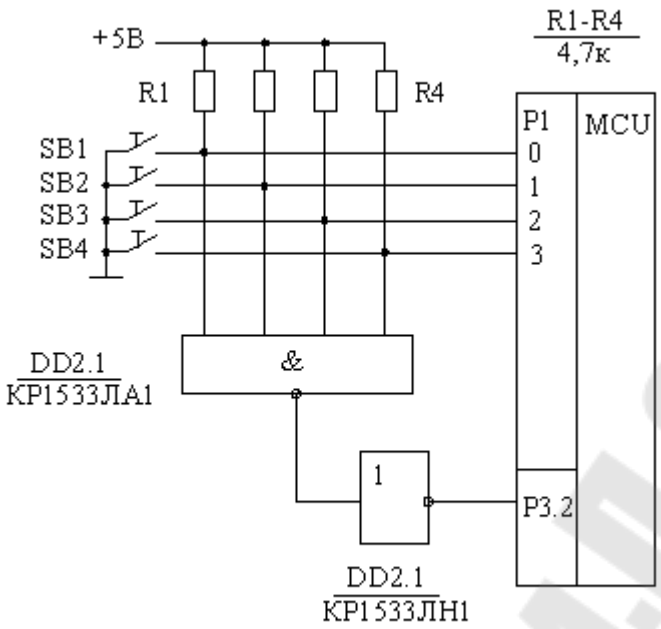


Рис. 3.11. Схема опроса состояния кнопок

На рис. 3.12 приведена схема динамического управления дисплеем со светодиодными индикаторами. Индикаторы HG1-HG4 используются для отображения значений аналоговых сигналов (например, U1-U4) в десятичном коде. Индикатор HG5 отображает номер аналогового канала. Светодиоды VD1-VD7 отображают значение сигналов X1-Y3,они работают также в динамическом режиме. Регистр DD5 служит для запоминания кода, выводимого на сегменты индикаторов HG1-HG5 и светодиоды VD1-VD7 с порта P0. В цепях общих электродов HG1-HG5 и светодиодов VD1-VD7 включены транзисторные ключи VT1-VT6. Они управляются кодом сканирования вида “бегущий нуль”, который поступает с линий P2.0-P2.5 МК. Микросхемы DD6-DD8 служат для усиления сигналов управления по току.

При использовании динамического способа управления дисплеем удобно код сканирования использовать и для управления клавиатурой, т.е. для идентификации нажатой клавиши. На рис. 3.13 приведена схема подключения контактов такой клавиатуры.

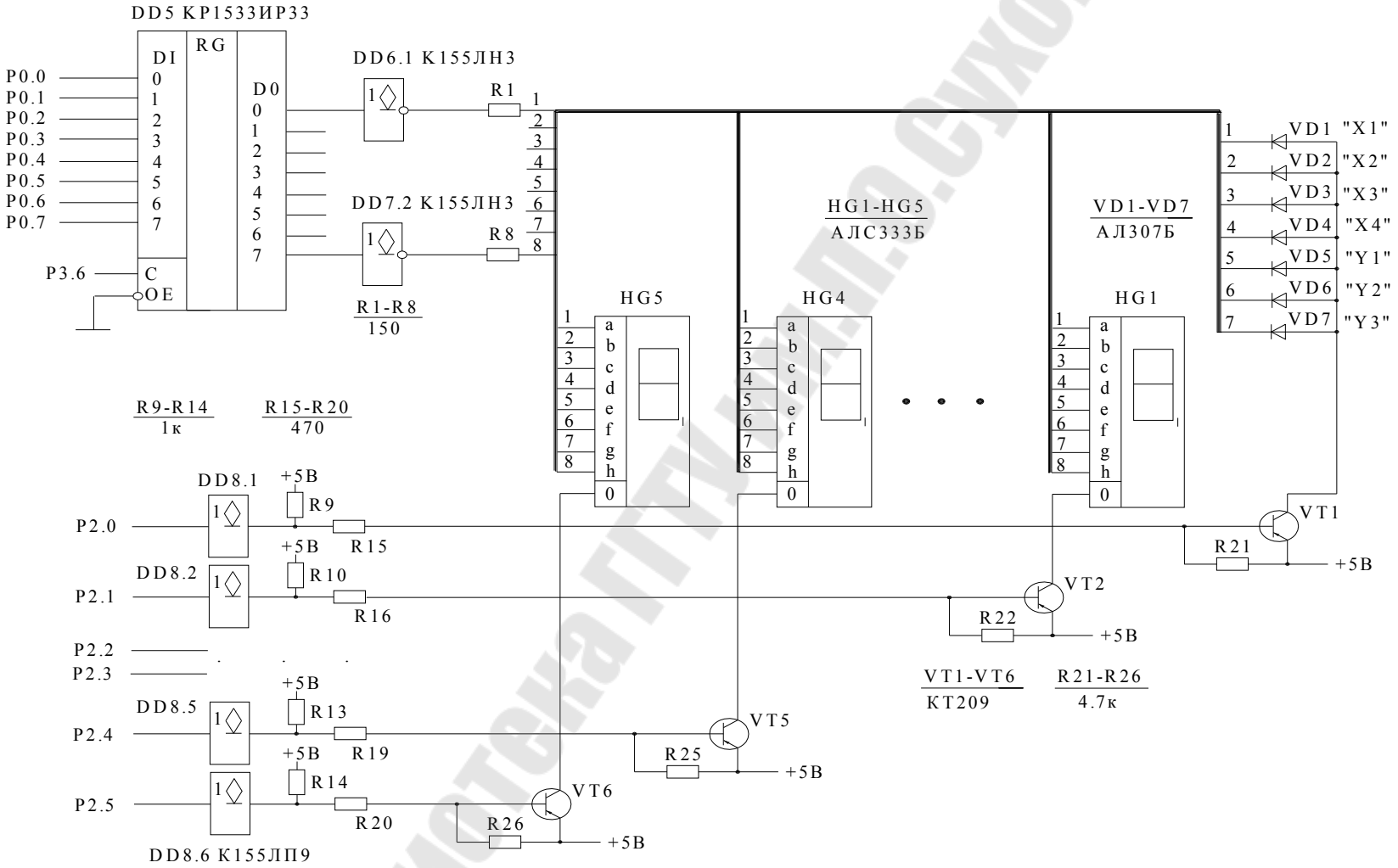


Рис.3.12. Схема динамического управления светодиодными индикаторами.

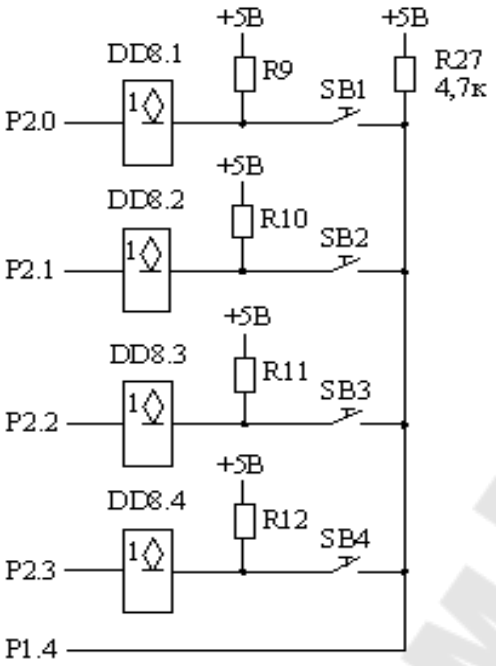


Рис. 3.13. Схема сканирования кнопок

3.7.Схемы сопряжения с последовательным интерфейсом

Разрабатываемая микроконтроллерная система должна иметь связь с внешним удаленным компьютером через последовательный интерфейс RS-232C или ИРПС (токовая петля). По запросу внешнего компьютера МК должен передавать данные о сигналах X1-Y3 и коды аналоговых сигналов U1-U4, получаемых с помощью АЦП. Обычно для связи МКС с внешним компьютером используют программный метод управления пере-
дачей, так как это позволяет уменьшить число линий связи [1]. В этом случае интерфейс должен обеспечить двустороннюю передачу, т.е. от компьютера к МК (запрос) и от МК к компьютеру (передача данных).

При сопряжении МК со стандартным последовательным интерфейсом необходимо решать следующие проблемы [1]:

- согласование уровней сигналов интерфейса с уровнями ТТЛ МК;
- поддержание стандартной скорости приемо-передачи;
- поддержание стандартных форматов посылки;
- поддержание стандартных протоколов обмена

На рис. 3.14 приведена схема сопряжения МК51 с интерфейсом RS-232C. Согласование уровней выполняется с помощью микросхемы DD1 КР559ИП20 (прием) и микросхемы DD2 КР559ИП19 (пере-

дача), специально разработанных для этой цели. При этом потребуются дополнительные источники электропитания напряжением +12 и -12 В. В схеме используется стандартный разъем DB9S (розетка). Сигналы RxD - принимаемые данные (от компьютера), TxD - передаваемые данные (в компьютер).

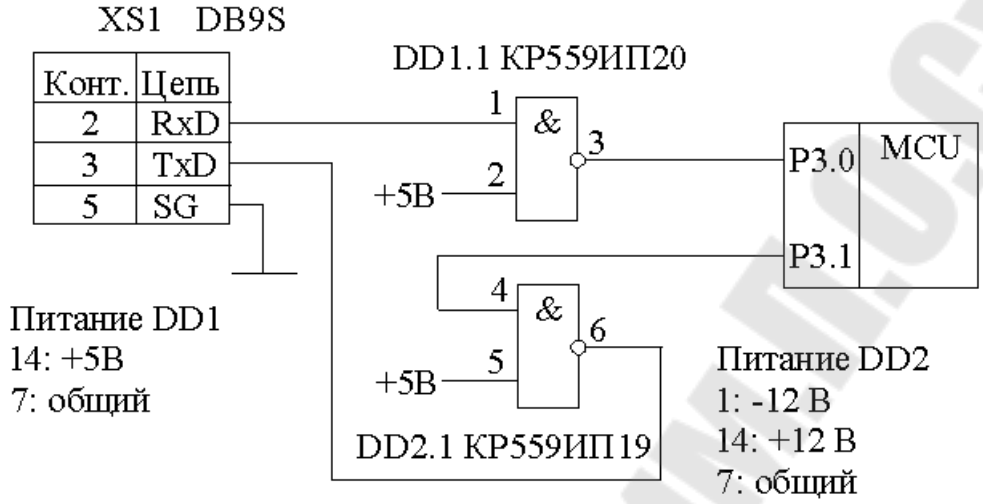


Рис.3.14. Схема сопряжения МК с интерфейсом RS-232C

На рис. 3.15 приведена схема сопряжения с интерфейсом ИРПС (токовая петля 20 мА). Принимаемые данные (импульсы тока 20 мА в цепи +ПрД/-ПрД) преобразуются в уровни ТТЛ с помощью оптрона V1 и инвертора DD1. Оптрон также обеспечивает гальваническую развязку передатчика (компьютера) и приемника (МК). Передаваемые данные (импульсы тока 20 мА в цепи +ПД/-ПД) образуются из уровня ТТЛ с помощью инвертора DD2 и оптрона V2, который также обеспечивает гальваническую развязку передатчика (МК) и приемника (компьютера). Схема требует дополнительный источник питания Uп1 напряжением 12 В, гальванически не связанный с источником +5В питания МК.

3.8. Пример разработки микроконтроллерной системы

Рассмотрим пример разработки МКС при следующих исходных данных:

Период опроса $T_{опр} = 2$ с.

Микроконтроллер КМ1816ВЕ751.

Тактовая частота работы МК: 12 МГц.

АЦП: К1113ПВ1 (10 разрядов).

Аналоговые сигналы: U1,U2,U3,U4.
Цифровые сигналы: X1,X2,X3,X4.
Логическая функция f( ) = $\overline{X1} \wedge (\overline{X2 \vee X3}) \vee X4$.
Функция g( ) =5W1 + W2 – K1
Длительности импульсов: t1 = 80 мкс, t2 = 24 мс, t3 = 75 мс.
Дисплей на семисегментных светодиодных индикаторах.
Управление дисплеем: статическое.
Метод получения семисегментного кода: аппаратурный.
Интерфейс: RS-232C.
Скорость обмена: 4800 бит/с.

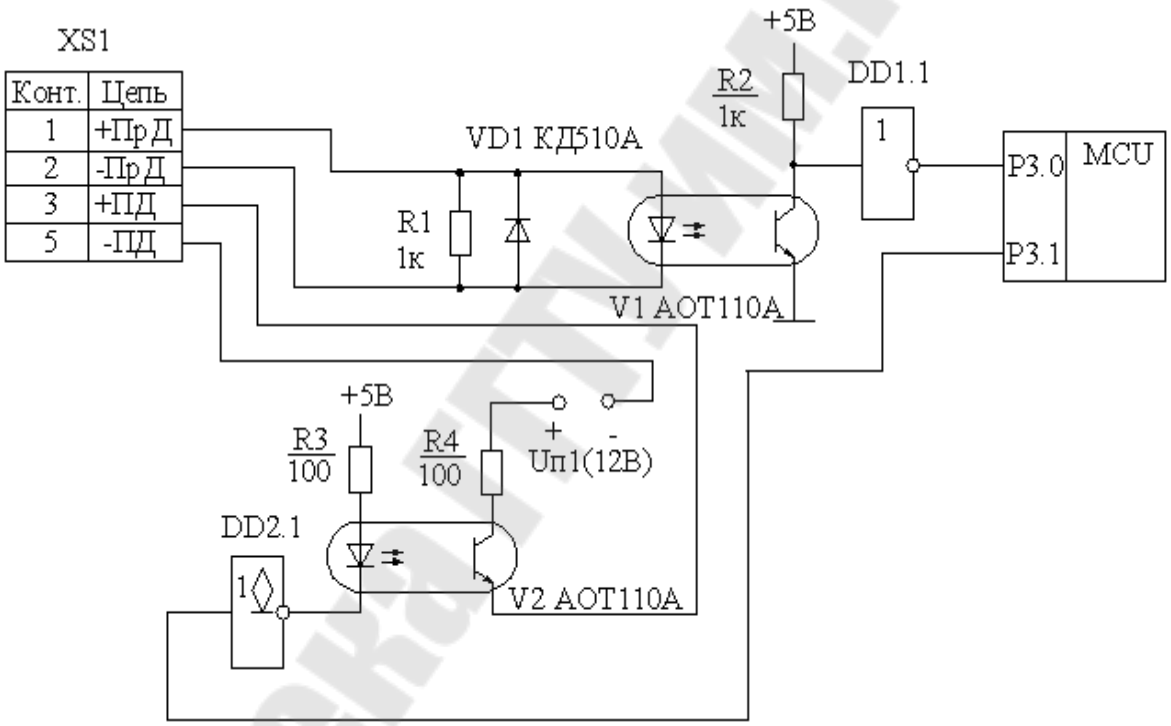


Рис.3.15. Схема сопряжения МК с интерфейсом ИРПС

3.8.1. Разработка структурной схемы МКС

Структурная схема составляется исходя из описания работы МКС (см. раздел 2) и исходных данных для проектирования.

Структурная схема разрабатываемой МКС приведена на рис. 3.16. Ее центральной частью является однокристальный микроконтроллер КМ1816ВЕ751, имеющий внутреннюю память программ объемом 4 Кбайт и 32 двунаправленных линий ввода-вывода. МК имеет 2 таймера: Т/С0 и Т/С1. Таймер Т/С0 используется для отсчета времени $T_{опр}$, а таймер Т/С1 используется для задания требуемой скорости обмена 4800 бит/с по интерфейсу RS-232C.

Аналоговые сигналы U1,U2,U3,U4 через коммутатор (аналоговый мультиплексор) поступают на АЦП. С выхода АЦП 10-разрядный код поступает в МК. МК выдает адрес для выбора канала коммутатора, а также сигнал для запуска АЦП.

Цифровые (двоичные) сигналы X1,X2,X3,X4 поступают на входы МК. Управляющие сигналы Y1,Y2 и Y3, представляющие собой одиночные импульсы определенной длительности, вырабатываются микроконтроллером. Они усиливаются по току с помощью формирователей сигналов ФС1, ФС2, ФС3.

Сигналы последовательного интерфейса TxD (передаваемые данные) и RxD (принимаемые данные) вырабатываются последовательным портом (УАПП) микроконтроллера. Для преобразования их в сигналы интерфейса RS-232C служит блок сопряжения.

Пульт оператора включает в себя: 7 светодиодов для индикации значений сигналов X1,X2,X3,X4,Y1,Y2,Y3; четырехпозиционный линейный дисплей на семисегментных светодиодных индикаторах для отображения аналоговых сигналов (напряжений) U1,U2,U3,U4; 4 переключателя для выбора аналоговых каналов; переключатель режима работы МКС; кнопку “Сброс”, при нажатии на которую производится сброс МК.

3.8.2. Разработка принципиальной схемы МКС

При разработке принципиальной схемы МКС можно использовать схемы отдельных блоков (модулей), которые были рассмотрены в п.п. 3.1 – 3.7.

Для рассматриваемого примера проектирования МКС, согласно исходным данным выбираем схему включения МК по рис. 3.1. Для

ввода аналоговых и цифровых сигналов применим схему на рис. 3.5. Переключатели присоединим по схеме на рис.3.10. Для вывода управляющих сигналов используем схему на рис. 3.6,а. Светодиоды индикации включим по рис.3.7,а, светодиодные индикаторы – по рис. 3.8. И, наконец, для сопряжения с интерфейсом RS-232C используем схему на рис. 3.14.

На основании этого выбора следует составить полную принципиальную схему МКС. На этой схеме нужно выполнить новую нумерацию всех ее элементов (микросхем, резисторов и т.д.), так как она, как правило, отличается от нумерации элементов на схемах отдельных модулей.

Необходимо дать описание выполняемых функций в принципиальной схеме МКС элементов, таких как АЦП, коммутаторы, регистры, светодиоды и индикаторы, микросхемы памяти, дешифраторы, а также переключателей и кнопок. Следует также дать краткое описание (справочные данные) этих элементов: условное графическое изображение, назначение выводов, основные электрические параметры, таблицы функционирования и т.п.

3.8.3. Разработка программного обеспечения работы МКС

Алгоритм работы МКС был приведен на рис. 2.2. По нему составим блок-схему алгоритма (БСА) главной программы работы МКС, которая будет иметь символическое имя CONTROL. БСА этой программы приведена на рис. 3.17.

Из нее видно, что главная программа состоит из нескольких подпрограмм, которые вызываются по мере надобности. Такое построение упрощает структуру программы, делает ее более наглядной. Кроме того, использование подпрограмм упрощает разработку и отладку всей программы работы МКС.

Текст главной программы на языке Ассемблера для МК51 имеет следующий вид:

```
;*******************************************************
; CONTROL – главная программа работы МКС
;*******************************************************
DATA_W:   EQU    30H      ; Начальный адрес массива в ПД для
                          ; хранения кодов АЦП
          ORG    0000H    ; Начальный адрес программы
```

```
CONTROL:  JMP     MAIN     ; Переход на основную программу

          ORG   000BH      ; Вектор прерывания от T/C0
          CALL  TIMER      ; Подпрограмма отсчета времени Tопр
          RETI


          ORG   0023H      ; Вектор прерывания от УАПП
          CALL  TRANSMIT   ; Подпрограмма передачи данных в
                           ;  интерфейс
          RETI

          ORG   0030H    ; Таблица констант
ADR_K1:   DW    XXXXH    ; Константа  K1
ADR_K2:   DW    XXXXH    ; Константа  K2
ADR_K3:   DW    XXXXH    ; Константа  K3
ADR_K4:   DW    XXXXH    ; Константа  K4
ADR_Q:    DW    XXXXH    ; Константа  Q

          ORG   0050H    ; Основная программа
MAIN:     MOV   SP, #70H ; Определить стек
          CALL   INIT    ; Подпрограмма инициализации МКС
AGAIN:    CALL DIGIT     ; Подпрограмма ввода и обработки
                         ; цифровой информации
          CALL  ANALOG ; Подпрограмма ввода и обработки
                         ; аналоговой информации
          SETB   ES      ; Разрешить прерывание от УАПП
          SETB   PS      ; Присвоить прерыванию от УАПП
                         ; высший приоритет
```

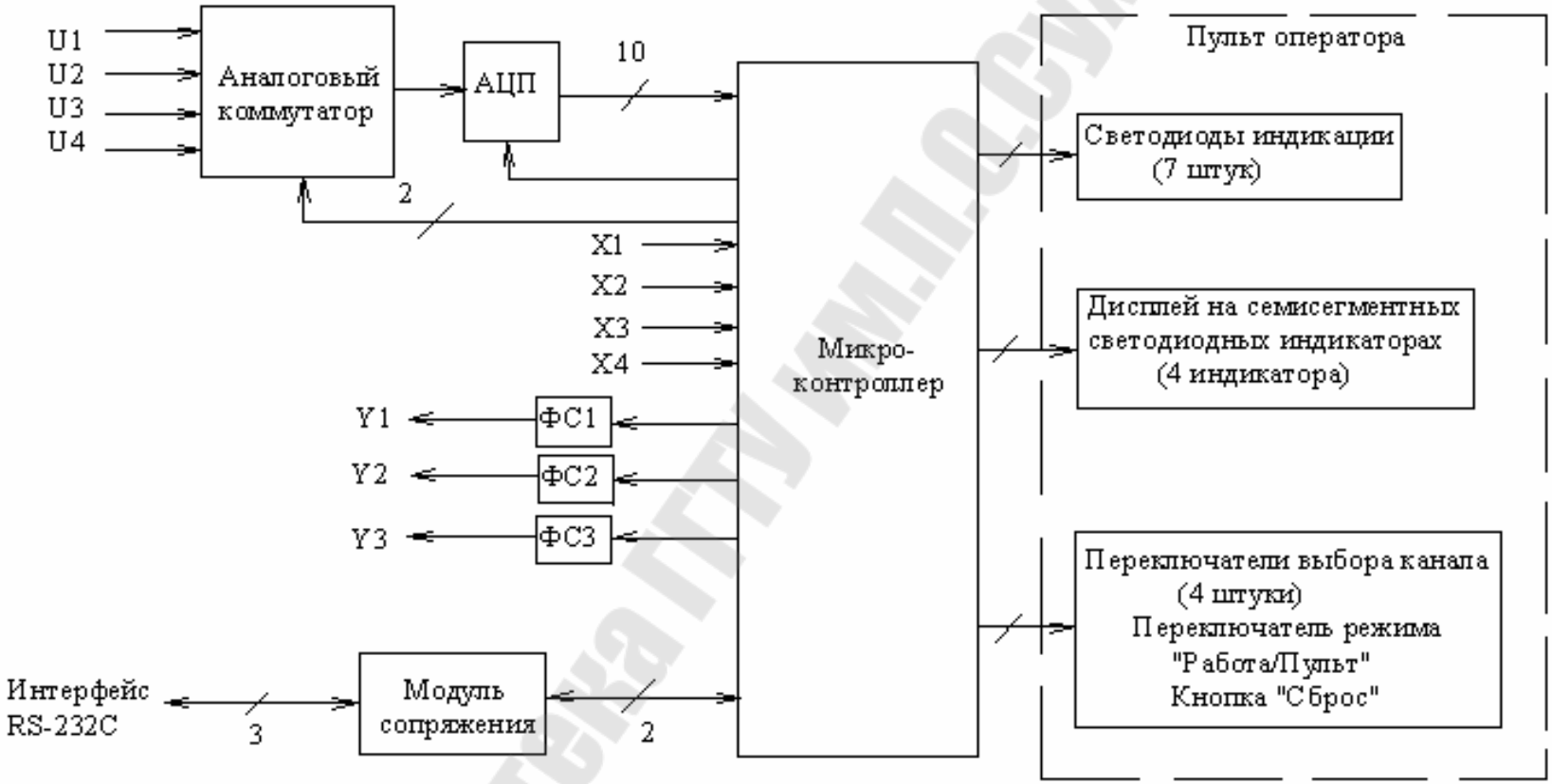


Рис.3.18. Структурная схема микроконтроллерной системы управления объектом

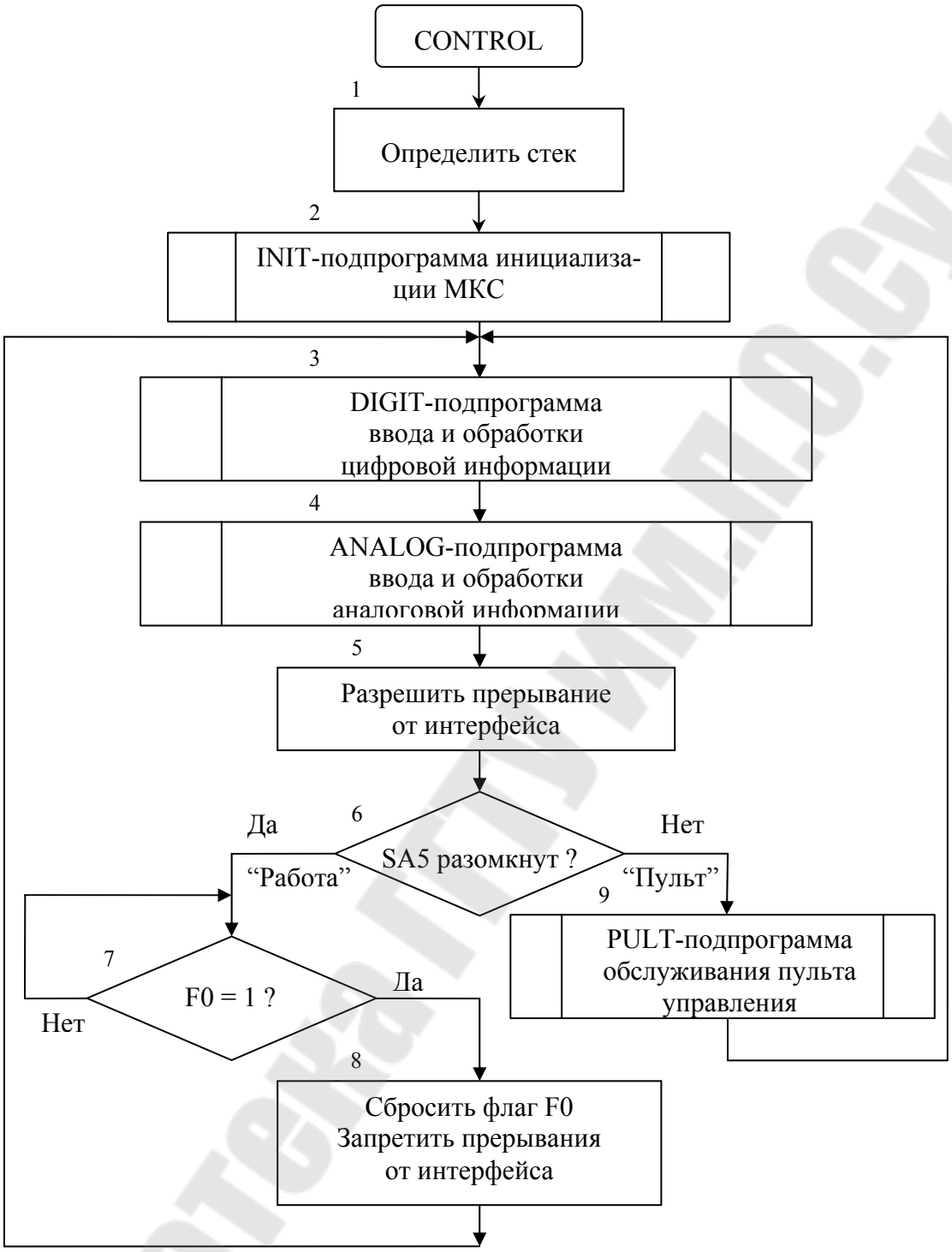


Рис. 3.17. БСА главной программы работы МКС

```
; Опрос состояния переключателя SA5 “Работа/Пульт”
            MOV   P0, #0FFH   ; Настроить порт P0 на ввод
            CLR    P3.5       ; Открыть выходные буферы DD4
            MOV  A,P0         ; Ввод из P0
            SETB  P3.5        ; Закрыть буферы  DD4
            JB  ACC.7, COUNT  ; Переход, если SA5 разомкнут
                              ; (Работа)
            CALL   PULT       ; Подпрограмма обслуживания
                              ;пульта управления
            JMP    AGAIN      ; Зацикливание программы
COUNT:      JNB    F0, COUNT  ; Цикл отсчета времени Tопр = 2с
            CLR   F0          ; Сбросить флаг окончания счета Tопр
            CLR   ES          ; Запретить прерывания от УАПП
            JMP   AGAIN       ; Зацикливание программы
```

В программе определен массив ячеек памяти данных МК с начальным символическим адресом DATA_W для хранения кодов с выхода АЦП. Так как входных переменных 4, то массив будет иметь 8 ячеек. В памяти программ, начиная с адреса 0030H, размещаются двухбайтные константы, которые будут использоваться при вычислениях функций. Значения констант условно обозначено XXXXH. Константы занимают в памяти две ячейки, например, по адресу ADR_K1 размещается младший байт (МлБ), а по следующему адресу ADR_K1+1 – старший байт (СтБ) константы. В главной программе используются 2 прерывания: от таймера T/C0 и от УАПП. Флаг F0 используется для индикации окончания отсчета времени опроса Tопр. Он устанавливается, когда истечет время Tопр = 2с.

Подпрограмма INIT используется для инициализации МКС после включения электропитания. БСА подпрограммы приведена на рис. 3.18. При сбросе МК все порты настраиваются на ввод [1,2], т.е. на их выводах будут высокие уровни (логические 1). Поэтому необходимо вывести P1.0 = P1.1 = P1.2 = 0 для установки начальных значений управляющих сигналов Y1 = Y2 = Y3 = 0. Необходимо также вывести нули в порт P0 для гашения светодиодов индикации VD1- VD7, так как после включения электропитания на выходах регистра DD5 (см. рис. 3.7,а) могут быть случайные значения. Индикаторы дисплея HG1 – HG4 после сброса МК будут автоматически погашены, так как на входы гашения E дешифраторов DD9 –DD12 с выхода инвертора DD8.2 поступает низкий уровень (см. рис. 3.8).

В подпрограмме инициализации необходимо настроить УАПП и таймеры Т/С0, Т/С1 на заданные режимы работы. Для двустороннего обмена по интерфейсу RS-232C выбираем для УАПП режим 1. Управляющее слово, которое следует загрузить в регистр SCON, будет 52Н [1,2]. Скорость обмена УАПП задает таймер Т/С1, который должен работать в режиме 2. Управляющее слово, которое нужно загрузить в регистр TMOD, будет 20Н. Для получения заданной скорости обмена в регистр TH1 таймера
Т/С1 необходимо загрузить число, которое вычисляется из соотношения [2]:

$$V_n = \frac{2^{SMOD} \cdot f_{CLK}}{32 \cdot 12 \cdot [256 - (TH1)]}. \qquad (3.1)$$

При SMOD = 0, f_{CLK} = 12 МГц и V_n = 4800 бит/с получаем из формулы число FAH.

Для отсчета времени $T_{опр}$ будем использовать таймер Т/С0 в 1-м режиме работы. Управляющее слово для этого режима равно 01Н. При тактовой частоте работы МК, равной 12 МГц, максимальное время задержки будет 65536 мкс [1,2], что значительно меньше требуемой 2 с. Поэтому на таймере Т/С0 реализуем задержку на 50 мс, а в регистре общего назначения (РОН) будем накапливать количество переполнений таймера с целью получения требуемой задержки в 2 с. Для получения задержки 50 мс в регистры таймера TH0,TL0 первоначально необходимо загрузить число 65536 – 50000 = 15536 = B03CH. Количество переполнений таймера для отсчета времени 2 с будет: 2000 мс / 50 мс = 40. Для счета переполнений будем использовать регистр R0 1-го банка РОН.

```
;*****************************************************
;  INIT – подпрограмма инициализации МКС
;*****************************************************
INIT:       ANL  P1, #11111000B  ; Вывести Y1 = Y2 = Y3 = 0
            MOV  P0, #0          ; Вывести нули в порт P0 (погасить
                                 ; VD1-VD7)
            CLR  P3.6            ; Отключить  DD5
            MOV  SCON, #52H      ; Режим 1 для УАПП,
```

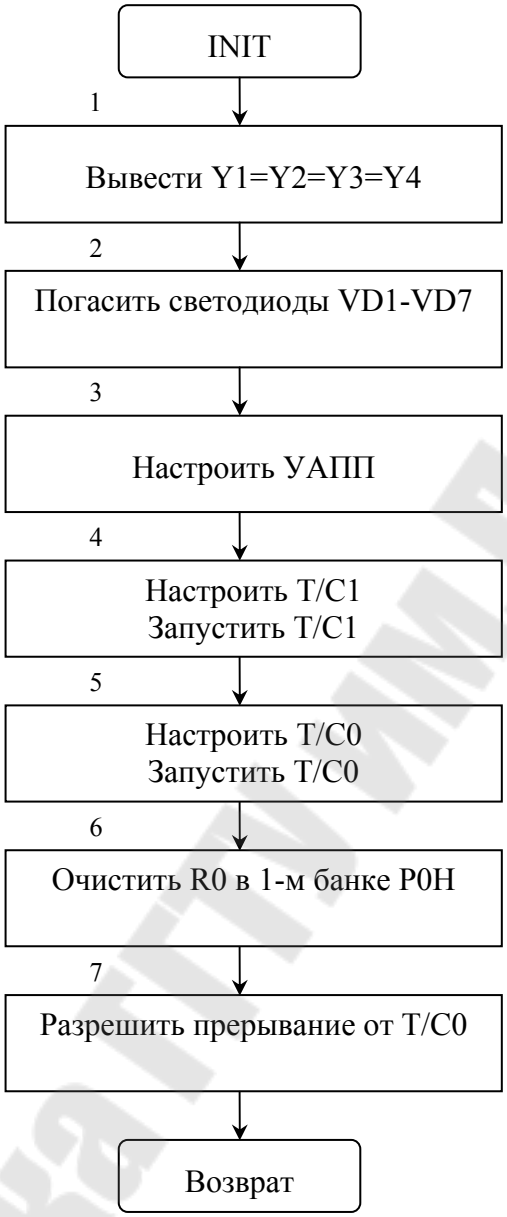


Рис. 3.18. БСА подпрограммы инициализации МКС

```
                        ; двусторонний  обмен
MOV   TMOD, #20H   ; Режим 2 для Т/С1
MOV   TH1, #0FAH    ; Загрузить TH1 для скорости
                        ; обмена 4800 бит/с
SETB   TR1              ; Запустить Т/С1
ORL    TMOD, #01H  ; Режим 1 для Т/С0
MOV   TH0, #0B0H    ; Загрузить регистры Т/С0
MOV   TL0, #3CH      ; для отсчета задержки 50 мс
```

```
        MOV   08H, #0       ; Очистка регистра R0 в 1–м
                             ; банке P0H
        SETB  TR0           ; Запустить T/C0
        SETB  EA            ; Общее разрешение прерываний
        SETB  ET0           ; Разрешить прерывание от T/C0
        RET
```

Подпрограмма TIMER является обработчиком прерывания при переполнении таймера T/C0, ее БСА приведена на рис. 3.19. Эта подпрограмма производит отсчет времени опроса $T_{опр}$. При каждом вызове подпрограммы вследствие переполнения таймера (истекло 50 мс) инкрементируется регистр R0 в 1-м банке РОН. Если содержимое R0 станет равным 40, то это означает, что истекло время $T_{опр}$ = 2 с. После окончания отсчета времени 2 с подпрограмма TIMER устанавливает флаг F0 = 1, что является сигналом для главной программы CONTROL начать новый цикл опроса датчиков и выполнения других функций управления.

```
;*************************************************************
;TIMER – подпрограмма отсчета времени Tопр = 2 с
; Выходной параметр: F0 – флаг окончания отсчета времени
; Используется 1-й банк P0H
;*************************************************************
TIMER:  CLR   TR0           ; Остановить T/C0
        SETB  RS0           ; Перейти в 1-й банк P0H
        INC   R0
        CJNE  R0, #40,EXIT      ; Переход, если (R0)<40
        SETB  F0            ; Установить флаг F0 окончания
                             ;счета 2 с
        MOV   R0, #0        ; Очистить R0
EXIT:   MOV   TH0, #0B0H        ; Перезагрузить  T/C0
        MOV   TL0, #3CH
        SETB  TR0           ; Запустить T/C0
        CLR   RS0           ; Перейти в 0-й банк P0H
        RET
```

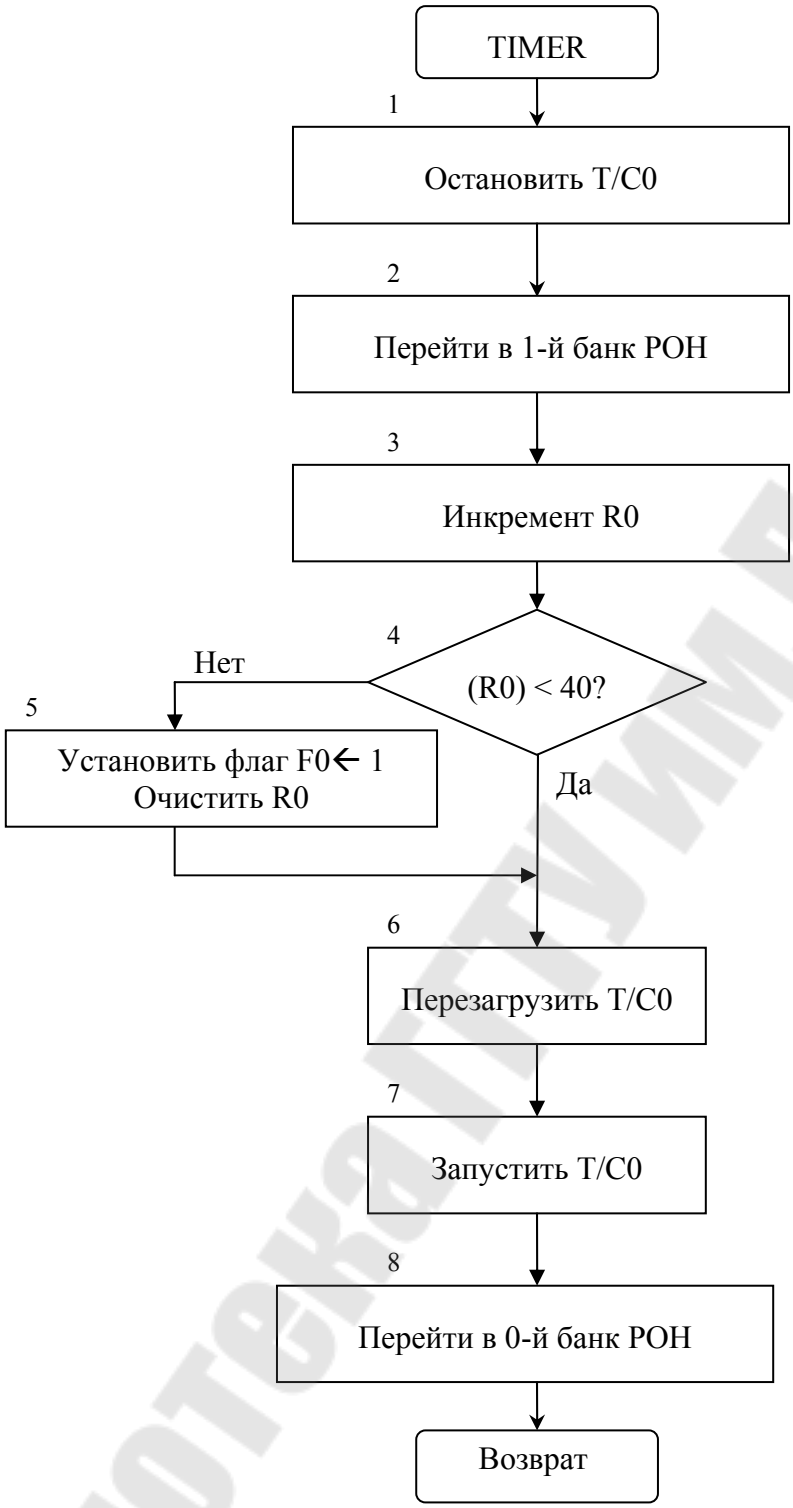


Рис. 3.19. БСА подпрограммы отсчёта времени $T_{опр}$

Подпрограмма DIGIT выполняет обработку цифровых сигналов X1,X2,X3 и X4, поступающих в МКС от датчиков. Обобщенная БСА этой подпрограммы приведена на рис. 2.3. По ней составим детальную БСА подпрограммы DIGIT, которая изображена на рис. 3.20.

Ввод цифровых сигналов осуществляется через регистр DD3 (см. рис. 3.5). Перед вводом данных из DD3 необходимо настроить порт P0 МК на ввод, а затем открыть выходные буферы DD3 подачей на управляющий вход OE низкого уровня с линии P3.4 порта МК (блоки 1 и 2 БСА). После ввода данных в аккумулятор МК надо вновь закрыть выходные буферы DD3 (блоки 3 и 4 БСА).

Ячейка ПД с адресом 24H используется в программе для хранения значений входных сигналов X1,X2,X3 и X4, а также для запоминания результата вычисления логической функции f() – управляющего сигнала Y1. Выбор этого адреса обусловлен тем, что эта ячейка, как и другие с адресами от 20H до 2FH, допускают обращение к отдельным битам с помощью команд SETB bit и CLR bit [1,2], что упрощает программирование. После выполнения подпрограммы DIGIT в ячейке 20H будет:

	7	6	5	4	3	2	1	0	- разряды
24H	0	0	0	Y1	X4	X3	X2	X1	

Для формирования одиночного управляющего импульса Y1 длительностью t1 = 80 мкс используется подпрограмма задержки DEL80MKS, текст которой будет приведен ниже.

```
;*****************************************************
; DIGIT – подпрограмма ввода и обработки цифровой информации
; Используется ячейка ПД с адресом 24H для хранения значений
; X1,X2,X3,X4,Y1
;*****************************************************
DIGIT:      MOV   P0, #0FFH   ; Настроить P0 на ввод
            CLR    P3.4       ; Открыть выходные буферы DD3
            MOV   A, P0       ; Ввод сигналов X1-X4
            SETB   P3.4       ; Закрыть буферы DD3
            SWAP   A
            ANL  A, #00001111B ; Обнулить СтТ аккумулятора
            MOV  24H, A       ; Переслать аккумулятор в
                              ; ячейку 24H
            MOV  C, 24H.1     ; Переслать C ← X2
            ORL   C, 24H.2    ; C ← X2∨X3
            CPL    C          ; C ← X2∨X3
```

```
        ORL   C, 24H.3          ; C ← X2 ∨ X3 ∨ X4
        ANL    C, /24H.0        ; C ← f( ) = X1 ∧ X2 ∨ X3 ∨ X4
        MOV  24H.4, C           ; Переслать результат в 24H.4 ← f( )
        JNC    EXIT             ; Переход, если C = f( ) = 0
        SETB   P1.0             ; Вывести Y1=1
        CALL  DEL80MKS  ; Подпрограмма задержки на 80мкс
        CLR   P1.0              ; Вывести Y1=0
EXIT:   RET
```

Подпрограмма ANALOG выполняет ввод и обработку аналоговой информации – напряжений U1,U2,U3 и U4, поступающих в МКС от датчиков. Обобщенная БСА этой подпрограммы приведена на рис. 2.5. По ней составим детальную БСА подпрограммы ANALOG, которая изображена на рис. 3.21. Первый блок БСА – это подпрограмма OPROS, которая производит ввод аналоговых сигналов U1 – U4, их преобразование в цифровые коды W1 – W4 и размещение в памяти данных МК. В результате выполнения подпрограммы OPROS в ПД в ячейках с адресами 30H – 37H будет следующая информация:

		7	6	5	4	3	2	1	0	- разряды
	37H	0	0	0	0	0	0	x	x	СтБ W4
	36H	x	x	x	x	x	x	x	x	МлБ W4
	35H	0	0	0	0	0	0	x	x	СтБ W3
	34H	x	x	x	x	x	x	x	x	МлБ W3
	33H	0	0	0	0	0	0	x	x	СтБ W2
	32H	x	x	x	x	x	x	x	x	МлБ W2
	31H	0	0	0	0	0	0	x	x	СтБ W1
(DATA_W)	30H	x	x	x	x	x	x	x	x	МлБ W1

Здесь обозначено: x – произвольное значение (0 или 1).

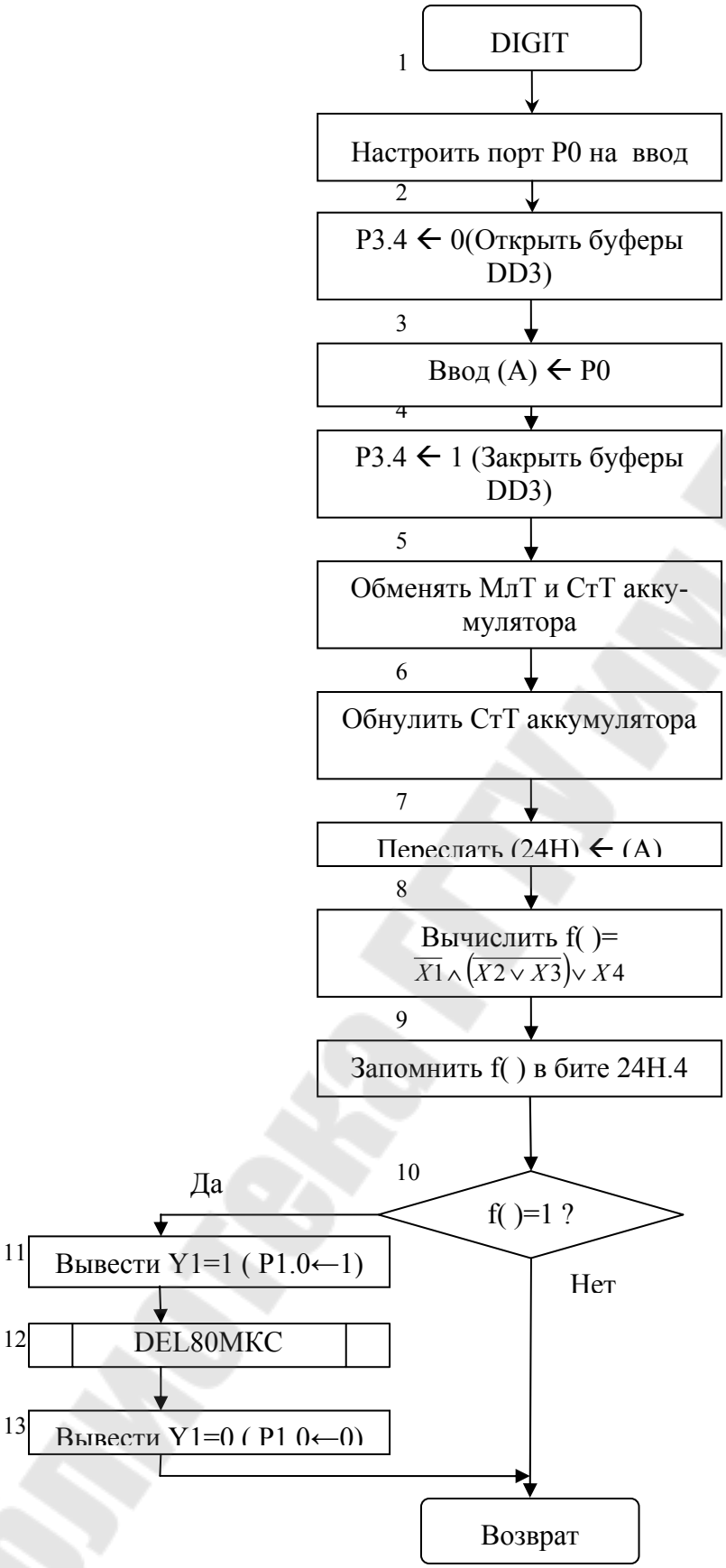


Рис. 3.20. БСА подпрограммы ввода и обработки цифровой информации

В блоках 2 – 4 подпрограммы ANALOG производится вычисление заданной функции g() = 5*W1 + W2 – K1.

Для выполнения арифметических операций умножения и деления можно воспользоваться готовыми подпрограммами, которые приведены в приложении данного руководства.

Примечания:

1.При выполнении операции умножения кода АЦП на постоянный коэффициент, например, 5*W1, надо использовать подпрограмму M16_8 – умножения двухбайтного числа на однобайтное. В общем случае результат (произведение) должен быть трехбайтным двоичным числом, которое размещается в регистрах R5, R4 и R3. Но так как максимальный код АЦП – это 10-разрядное двоичное число (03FFH), то даже при умножении на коэффициент 64 в результате будет только двухбайтное число, которое размещается в регистрах R4 и R3.

2.При выполнении операции деления на постоянное число используется подпрограмма деления D16_8. При этом следует учитывать только целое частное в регистре R3, а остатком в регистре R4 можно пренебречь.

3.Постоянные коэффициенты в выражениях для функции g() в исходных данных подобраны таким образом, что при всех вычислениях (промежуточных и окончательных) получаются только двухбайтные двоичные числа.

В блоке 5 БСА полученное значение функции g() сравнивается с двухбайтной константой Q, хранящейся в памяти программ по символическому адресу ADR_Q. Сравнение можно выполнить с помощью программы вычитания двухбайтных целых чисел g() – Q. Результат сравнения определяется значением флага переноса C микроконтроллера после вычитания старших байтов. Если C=1, то g() < Q. Если же C=0, то g() >=Q. В блоке 6 подпрограммы ANALOG производится анализ флага C, и в зависимости от его значения формируются одиночные импульсы Y1 или Y2 длительностью t1 и t2 на выводах P1.1 и P1.2 порта МК. Подпрограммы задержек DEL24MS на 24 мс и DEL75MS на 75 мс будут приведены ниже. После выполнения подпрограммы ANALOG в ячейке с адресом 24H будет

	7	6	5	4	3	2	1	0	- разряды
24H	0	Y3	Y2	Y1	X4	X3	X2	X1	

```
;*****************************************************
```

```
; ANALOG – подпрограмма ввода и обработки аналоговой информа-
ции
;**************************************************
ANALOG:    CALL  OPROS              ; Подпрограмма ввода
   ;аналоговых сигналовU1-U4, преобразования их в коды W1-W4 и
   ; запоминания в массиве DATA_W
                ; Вычисление 5·W1
              MOV   R0, #DATA_W  ; Адрес массива DATA_W
              MOV   R3, @R0        ; МлБ множимого
              INC    R0
              MOV   R4, @R0        ; СтБ множимого
              MOV   R2, #5          ; Множитель
              CALL  M16_8             ; Подпрограмма умножения,
                ; результат: R4-СтБ, R3-МлБ  произведения
      ; Вычисление 5·W1+ W2
              INC    R0             ; Адрес МлБ кода W2
              MOV  R6, @R0     ; МлБ кода W2
              INC    R0
              MOV  R7, @R0    ; СтБ кода W2
              MOV  A, R3
              ADD   A, R6         ; Сложение младших байтов
              MOV   R6, A         ; МлБ суммы
              MOV   A, R7
              ADDC  A, R4         ; Сложения старших байтов
              MOV   R7, A         ; СтБ суммы
          ; Вычисление g( ) = 5·W1+ W2 - K1
              MOV  DPTR, #ADR_K1 ; Адрес константы K1
              CLR    A
              MOVC       A, @A+DPTR ; В аккумуляторе – МлБ
                                    ; константы K1
              MOV   R4, A            ; В R4 – МлБ константы K1
              INC   A
              MOVC  A, @A+DPTR ; В аккумуляторе – СтБ
                                  ; константы K1
              MOV  R5, A             ; В R5 – СтБ константы K1
              MOV  A, R6
              CLR    C               ; Сбросить флаг C
              SUBB  A, R4
              MOV   R4, A            ; МлБ   g( )
```

```
            MOV   A, R7
            SUBB  A, R5
            MOV   R5, A            ; СтБ  g( )
        ; Вычисление g( ) – Q
            MOV   DPTR, #ADR_Q ; Адрес константы Q
            CLR  A
            MOVC  A, @A+DPTR
            MOV  R6, A        ; В регистре R6 - МлБ константы Q
            INC  A
            MOVC  A, @A+DPTR
            MOV  R7, A        ; В регистре R7 – СтБ константы Q
            CLR   C
            MOV   A, R4
            SUBB   A, R6       ; Вычитание младших байтов
            MOV   A, R5
            SUBB  A, R7
            JC   FORM_Y2       ; Переход, если меньше (C=1)
            SETB  P1.2         ; Вывести Y3=1
            SETB   24H.6       ; Запомнить Y3=1 в ячейке 24H
            CALL  DEL75MS   ; Подпрограмма задержки на 75 мс
            CLR   P1.2         ; Вывести Y3=0
            RET
FORM_Y2:    SETB   P1.1        ; Вывести Y2=1
            SETB   24H.5       ; Запомнить Y2=1 в ячейке 24H
            CALL  DEL24MS  ; Подпрограмма задержки на 24 мс
            CLR   P1.1         ; Вывести Y2=0
            RET
```

Примечание. В некоторых заданиях для выполнения курсового проекта требуется вычислять функции g() вида:

$$g() = \min \{5*W1 + W2;\ W3 - K3\} \qquad (3.2)$$

или

$$g() = \max \{5*W1 + W2;\ W3 - K3\}. \qquad (3.3)$$

Функцию g() в формуле (3.2) можно условно записать в виде:

$$g() = \min \{N;\ M\}.$$

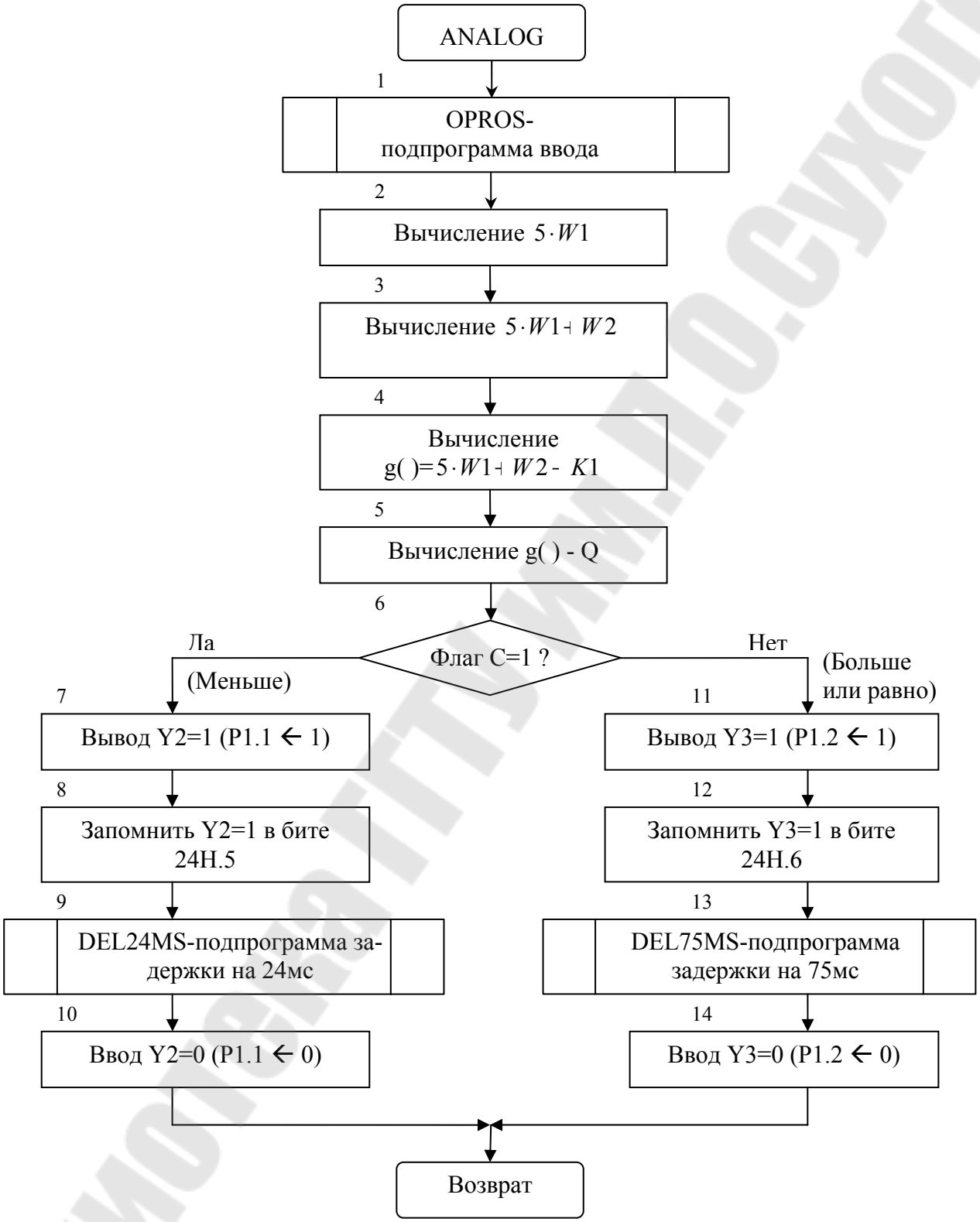


Рис. 3.20. БСА подпрограммы ввода и обработки аналоговой информации

Сначала следует вычислить выражения N = 5*W1 + W2 и M = W3 – K3, которые будут двухбайтными двоичными целыми числами. Затем нужно их сравнить между собой с целью определения минимального значения. Оно и будет являться искомым значением функции g(). Например, если N < M, то g() = N.

Во втором случае формулу (3.3) можно условно записать в виде:

$$g() = \max \{ N; M\}.$$

После вычисления выражений N = 5*W1 + W2 и M = W3 – K3, их нужно сравнить между собой с целью определения максимального значения. Оно и будет являться искомым значением функции g(). Например, если N >= M, то g() = N.

Сравнение N и M можно выполнить с помощью программы вычитания двухбайтных двоичных целых чисел, т.е. N – M. Результат сравнения определяется флагом C микроконтроллера после вычитания старших байтов. Если C = 1, то N < M. Если же C = 0, то N >= M.

Подпрограмма OPROS производит переключение аналоговых каналов коммутатора DA1 (см. рис. 3.5), обслуживание АЦП и размещение в ПД в массиве DATA_W кодов W1,W2,W3,W4. Она использует подпрограмму ADCONV – аналого-цифрового преобразования. БСА подпрограммы OPROS приведена на рис. 3.22.

```
;*****************************************************
;OPROS – подпрограмма ввода аналоговых сигналов U1-U4.
;*****************************************************
OPROS:    MOV  R0, #DATA_W      ; Начальный адрес массива
                                ;DATA_W
          MOV  R7, #4           ; Счетчик аналоговых сигналов
NEXT:     MOV  R2, #11111100B ; Регистр R2 - код для выбора
                                ; канала коммутатора DA1
          MOV   P1, R2          ; Вывод в P3 - выбор канала
                                ;  коммутатора
          CALL  ADCONV          ; Подпрограмма
                                ; АЦ-преобразования,
    ; в регистре B –  МлБ, в регистре A – СтБ кода АЦП
          MOV   @R0, B          ; Пересылка МлБ кода АЦП в ПД
          INC   R0
          MOV  @R0, A           ; Пересылка СтБ кода АЦП в ПД
          INC        R0
          INC        R2
```

```
        DJNZ  R7, NEXT      ; Цикл, если (R7) ≠ 0
        RET
```

Подпрограмма ADCONV выполняет обслуживание АЦП, ее БСА приведена на рис. 3.23. Для запуска АЦП К1113ПВ1 формируется импульс вида 0->1->0 на линии P1.5 порта МК. После этого МК ожидает момента окончания преобразования кода АЦП, анализируя сигнал готовности данных на выводе DR, который поступает из регистра DD3 на линию P0.2 порта МК (блок 4). Когда сигнал DR станет равным 0, производится ввод старшего байта кода АЦП из регистра DD3 в аккумулятор МК, а затем младшего байта из регистра DD2 в регистр B микроконтроллера.

```
;*************************************************
; ADCONV – подпрограмма аналого-цифрового преобразования
; Выходные параметры: регистры A, B – СтБ, МлБ кода АЦП
;*************************************************
ADCONV: CLR   P1.5              ; Запустить  АЦП
        SETB P1.5
        CLR         P1.5
        MOV  P0, #0FFH      ; Настроить порт P0 на ввод
        CLR         P3.4        ; Открыть выходные буферы DD3
TEST:   JB   P0.2, TEST ; Цикл ожидания готовности данных АЦП
        MOV A, P0               ; Ввод СтБ кода АЦП
        ANL  A, #00000011B  ; Выделить разряды D1,D0
                                ;аккумулятора
        SETB  P3.4              ; Закрыть буферы DD3
        CLR   P3.3              ; Открыть буферы DD2
        MOV  B, P0              ; Ввод МлБ кода АЦП
        SETB  P3.3              ; Закрыть буферы DD2
        RET
```

Подпрограмма PULT используется для обслуживания пульта управления. Она вызывается, если переключатель режима SA5 будет в положении “Пульт”. БСА подпрограммы приведена на рис. 3.24. Рассмотрим программную реализацию некоторых блоков алгоритма.

В блоке 2 подпрограммы PULT выводится информация на светодиоды индикации VD1 – VD7 из ячейки ПД с адресом 24H, где хранятся значения переменных X1-X4, Y1-Y3.

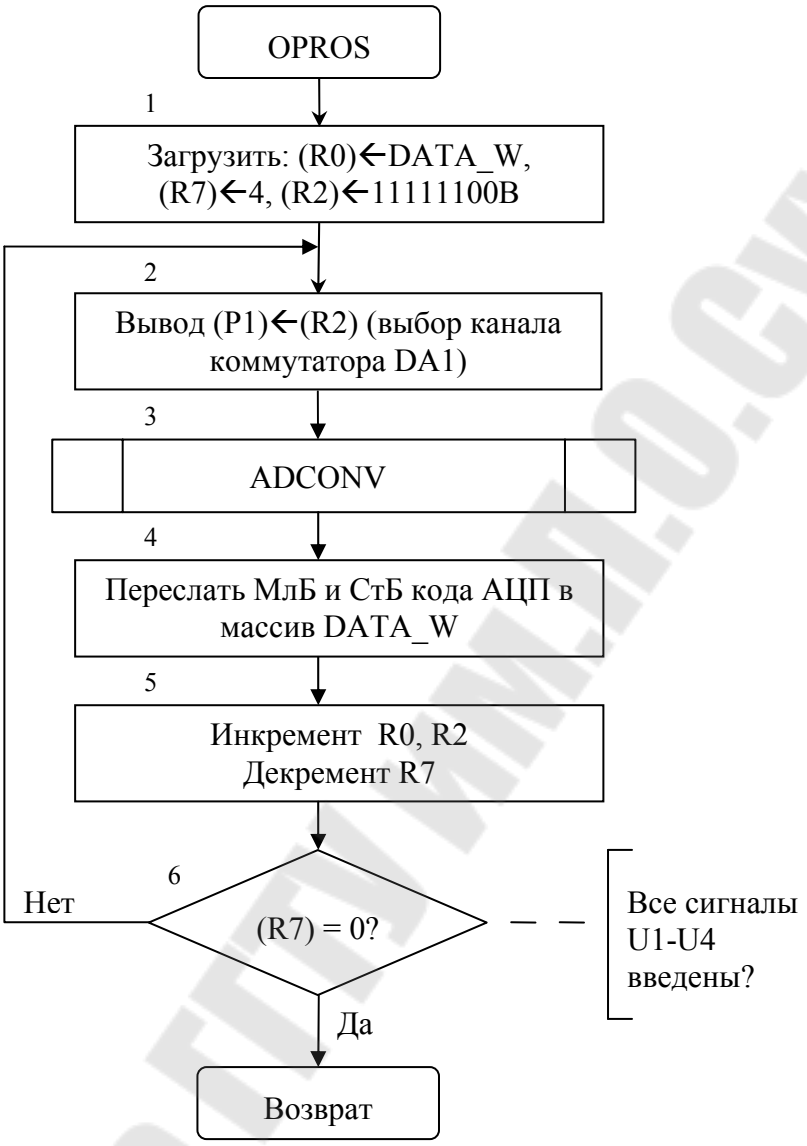


Рис. 3.23. БСА ввода аналоговых сигналов U1-U4

Блок 3 – это подпрограмма IDNKEY – идентификации замкнутых контактов переключателей аналоговых каналов SA1-SA4. Затем по коду переключателя в блоке 4 вычисляется адрес кода Wi из массива DATA_W, и в блоке 5 код Wi пересылается в регистры R2 и R3.

Блок 6 – это подпрограмма CALC, в которой вычисляется значение кода Ui для вывода на дисплей.

Блок 7 – это подпрограмма вывода на индикаторы дисплея.

В блоке 8 программа проверяет положение переключателя SA5 “Работа/Пульт” на пульте управления. Если SA5 замкнут, то продолжается программа обслуживания пульта управления. Если же SA5 разомкнут, то программа гасит светодиоды индикации и дисплей (блок 9) и запускает таймер T/C0 для отсчета времени Tопр (блок 10). Затем

происходит переход на блок 3 главной программы CONTROL (см. рис. 3.17).

```
;*************************************************************
; PULT – подпрограмма обслуживания пульта управления
;*************************************************************
PULT:       CLR  ES          ; Запретить прерывания
            CLR  ET0         ; от УАПП и Т/С0
            CLR  TR0         ; Остановить Т/С0
   ; Вывод информации X1-X4, Y1-Y3  на светодиоды VD1-VD7
            MOV  P0, 24H   ; Вывод в порт P0
            SETB   P3.6            ; Открыть триггеры регистра
                                   ;DD5 (Вывод на VD1-VD7)
            CLR   P3.6       ; Защелкивание данных в DD5
KEY:        CALL  INDKEY ; Подпрограмма идентификации
      ;замкнутых SA1-SA4 ,в регистре R7-двоичный код SA1-SA4
          ; Вычисление адреса кода Wi АЦП по коду SA1-SA4
            MOV   A, R7
            ADD    A, R7     ; Удвоение кода переключателей
                             ;SA1-SA4
            MOV    R7, A
            MOV   A, #DATA_W; Начальный адрес массива
                              ;DATA_W
            ADD    A, R7
            MOV    R0, A    ; В регистре R0-адрес МлБ кодаWi
        ; Пересылка кода Wi в регистры R3 и R2
            MOV  R3, @R0 ; МлБ кода АЦП
            INC   R0
            MOV   R2, @R0; СтБ кода АЦП
            CALL  CALC    ; Подпрограмма вычисления кода
                          ; Ui для вывода на HG1-HG4
            CALL DISPLAY       ; Подпрограмма вывода кода Ui
                               ;на HG1-HG4
```

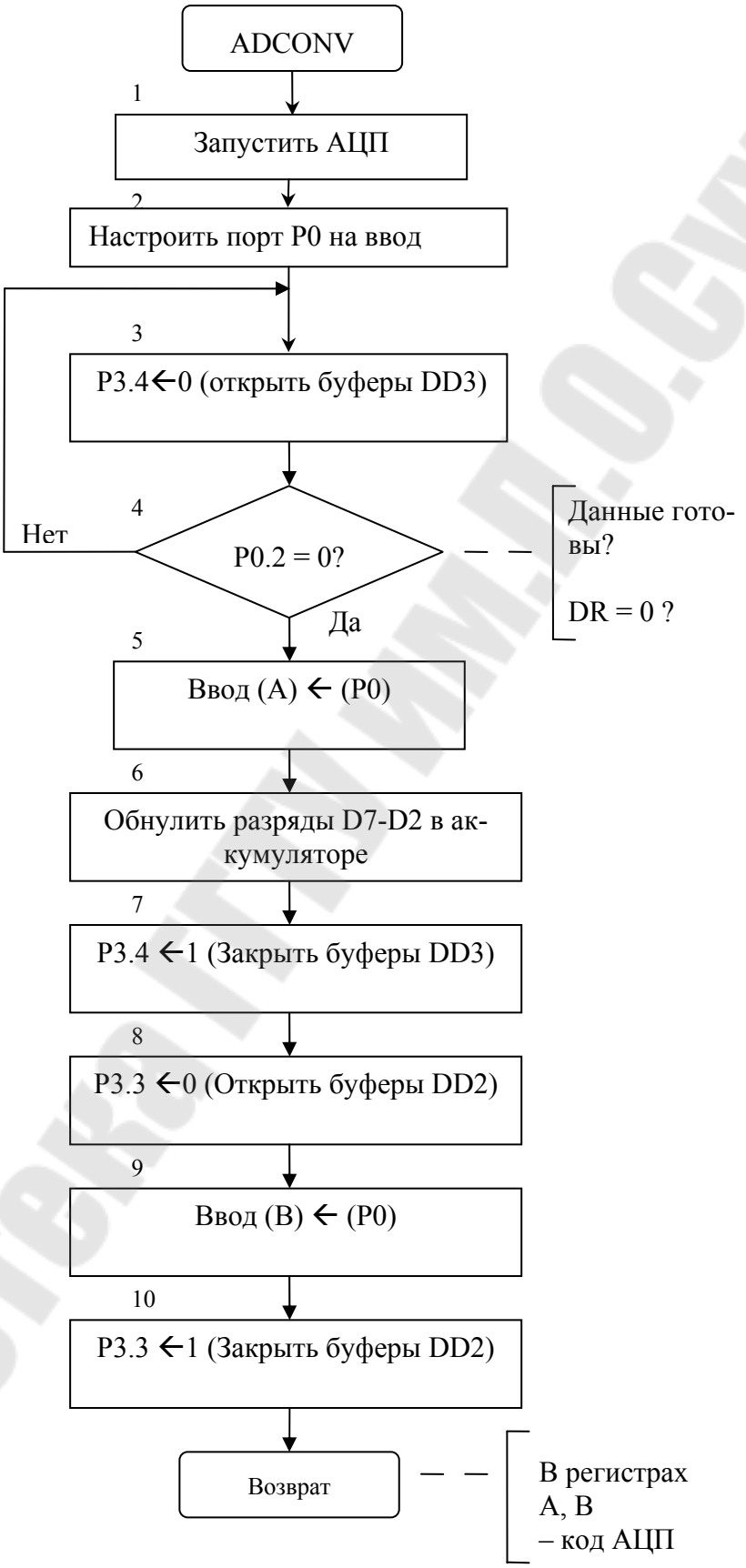


Рис. 3.23. БСА подпрограммы обслуживания АЦП

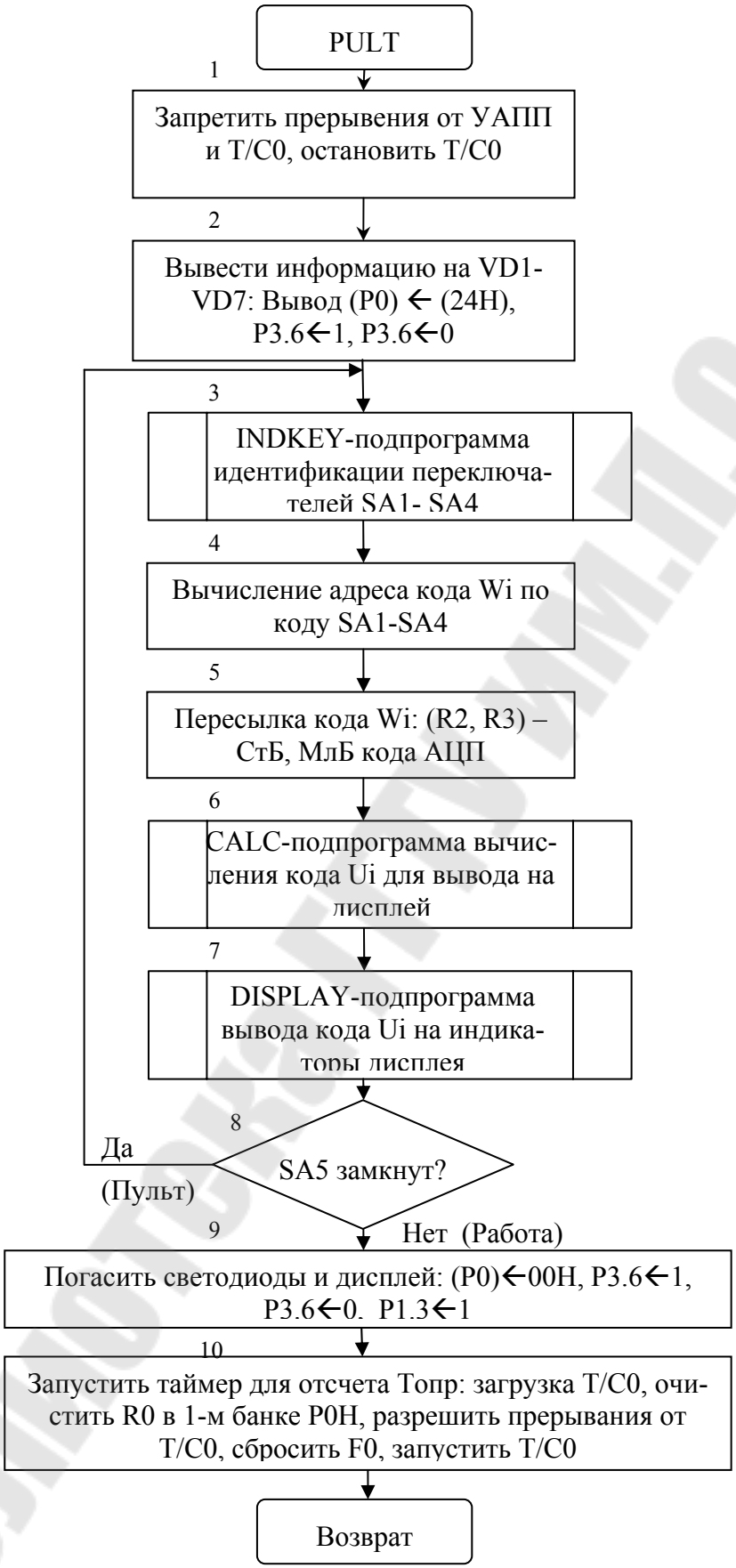


Рис. 3.24. БСА подпрограммы обслуживания пульта управления

```
; Проверка состояния переключателя SA5
        MOV  P0,#0FFH; Настроить порт P0 на ввод
        CLR   P3.5          ; Открыть выходные буферы DD4
        MOV  A,P0           ; Ввод из P0
        SETB  P3.5          ; Закрыть буферы DD4
        JNB  ACC.7, KEY; Переход, если SA5 замкнут
                            ; (“Пульт”)
          ; Переключатель SA5 разомкнут
          ;( положение “Работа”)
        MOV  P0, #0       ; Вывести нули в порт P0
        SETB  P3.6        ; Погасить светодиоды VD1-VD7
        CLR   P3.6        ; Закрыть триггеры DD5
        SETB  P1.4        ; Погасить индикаторы HG1 - HG4
        MOV TH0,#0B0H; Загрузить T/C0
        MOV  TL0,#3CH
        MOV  08H,#0       ; Очистить регистр R0 банка 1
        CLR    F0           ; Сбросить флаг окончания счета
                            ; времени
        SETB          ET0  ; Разрешить прерывание от T/C0
        SETB          TR0        ; Запустить T/C0
        RET
```

Определение номера замкнутого контакта переключателя SA1 – SA4 выполняет подпрограмма IDNKEY, которая присваивает ему позиционный двоичный код в соответствии с табл. 3.1.

Таблица 3.1

Коды для переключателей SA1-SA4

Замк-нут	Унитарный код	Позиционный код, HEX
SA1	XXXX1110	00
SA2	XXXX1101	01
SA3	XXXX1011	02
SA4	XXXX0111	03

БСА подпрограммы приведена на рис. 3.25. Унитарный код с входов регистра DD4 пересылается в аккумулятор (блоки 1 – 4), а затем сдвигается вправо до появления нуля во флаге переноса C. Количест-

во сдвигов накапливается в регистре R7. При выходе из подпрограммы в нем будет позиционный двоичный код замкнутого переключателя. Для корректной работы подпрограммы необходимо, чтобы соблюдалось условие: всегда имеется только один замкнутый контакт, т.е. контакты переключателей SA1 – SA4 должны быть взаимозависимы.

```
;*************************************************
; IDNKEY – подпрограмма идентификации замкнутого переключателя ;SA1-SA4
; Выходной параметр: регистр R7 - позиционный код переключателя
;*************************************************
IDNKEY:   MOV P0, #0FFH      ; Настроить порт P0 на ввод
          CLR  P3.5          ; Открыть выходные буферы DD4
          MOV A, P0                ; Ввод из P0
          SETB  P3.5          ; Закрыть буферы DD4
          ANL  A,#00001111B ; Выделить разряды D3-D0
          MOV R7, #0          ; Счетчик сдвигов
SHIFT:    RRC A               ; Сдвиг  аккумулятора вправо
          JNC   EXIT          ; Выход,  если C=0
          INC   R7
          JMP  SHIFT          ; Цикл сдвига
EXIT:     RET
```

Подпрограмма CALC вычисляет коды, соответствующие аналоговым сигналам (напряжениям) U1 – U4, для последующего вывода на дисплей.

Для АЦП К1113ПВ1 при 10-разрядном включении диапазон изменения выходного кода будет 0000H … 03FFH, что соответствует входному однополярному напряжению в диапазоне 0 … +10,24 В [3]. Чтобы найти напряжение U_i, соответствующее коду АЦП W_i, составим пропорцию:

$$U_i\ B \quad ---- \quad W_i$$
$$10,24\ B \quad ---- \quad 03FFF. \qquad (3.4)$$

Откуда получим:

$$U_i = 10,24\frac{W_i}{03FF}, B. \qquad (3.5)$$

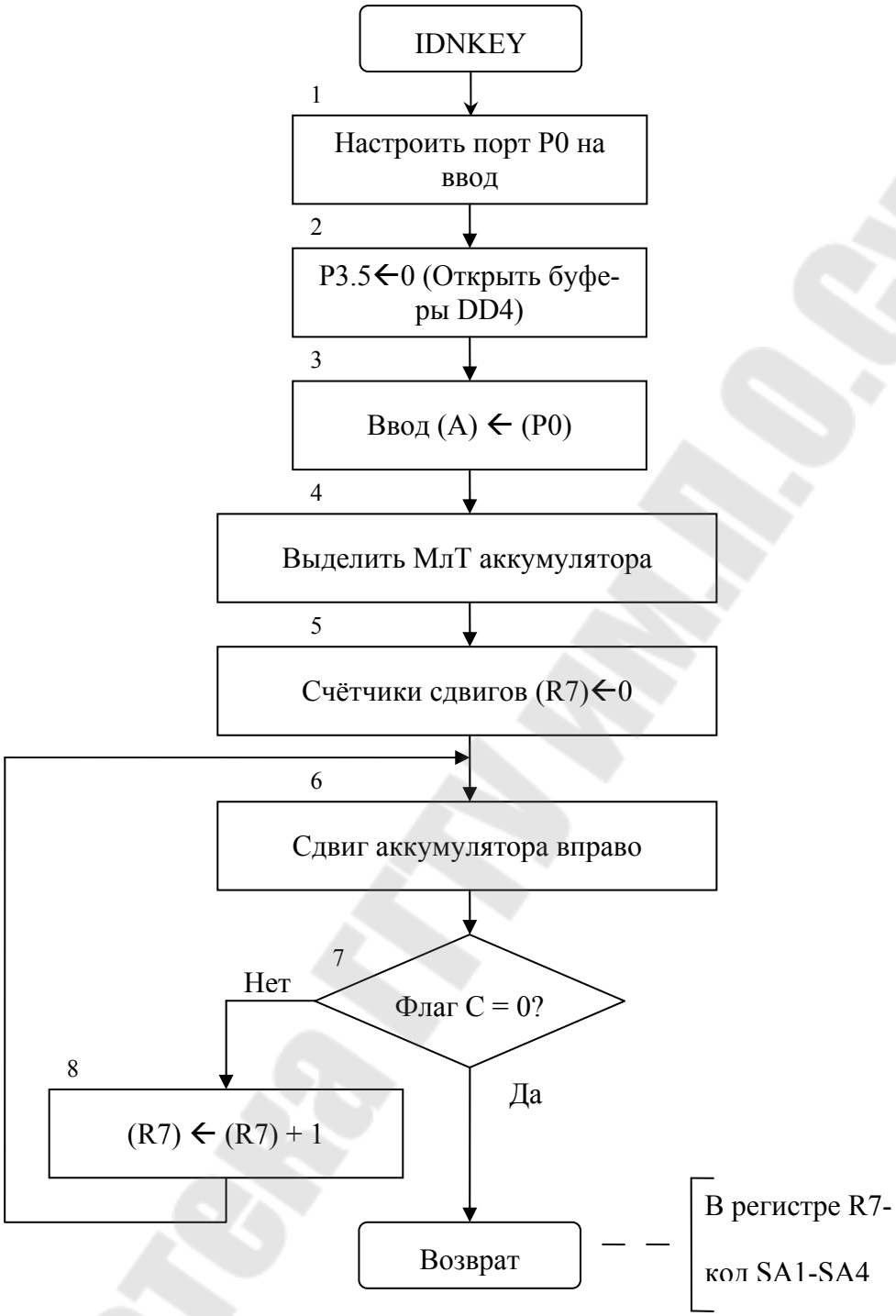


Рис. 3.25. БСА идентификации замкнутого SA1-SA4

Все входящие в формулу (3.5) числа надо преобразовать в дробные в формате с фиксированной запятой [10].

Имеем:

$$U_i = 10^2 \cdot 0{,}1024 \frac{W_i \cdot 2^{16}}{03FF \cdot 2^{16}} = 10{,}24 \frac{W_i}{03FF} = 10^2 \cdot, YYYY = YY, YY \quad , (3.6)$$

где условно обозначено: *,YYYY* – дробное число в формате с фиксированной запятой в двоично-десятичном коде.

БСА подпрограммы CALC приведена на рис. 3.25. Вычисление кода Ui производится по формуле (3.6). Для выполнения арифметических операций с дробными числами в формате с фиксированной запятой (ФЗ) можно воспользоваться готовыми подпрограммами из приложения данного пособия.

Деление дробных двоичных чисел с ФЗ ,Wi / ,03FF выполняется подпрограммой с именем D16_16F. Для ее работы необходимо загрузить в регистры R2 и R3 старший и младший байты (СтБ и МлБ) кода Wi (делимое), а в регистры R4 и R5 – СтБ и МлБ числа 03FF (делитель). Эта операция выполняется в блоке 1 БСА. Результатом деления (блок 2) будет дробное двоичное число с ФЗ в регистрах R6,R7 – СтБ, МлБ частного. В блоке 3 частное пересылается для временного хранения в регистр DPTR.

В блоке 4 посредством подпрограммы с именем C10_16AF производится преобразование дробного двоично-десятичного числа ,1024 в дробное двоичное число с ФЗ, которое размещается в регистрах R3,R4.

В блоке 6 производится умножение полученных в блоках 2 и 4 дробных двоичных чисел с помощью подпрограммы M16_16F. Результат умножения будет находиться в регистрах R7,R6.

В блоке 7 посредством подпрограммы с именем C16_10F производится преобразование дробного двоичного числа в дробное двоично-десятичное число, которое разместится в регистрах R2,R3 в виде:

,

R2		R3	
P1	P2	P3	P4

Здесь обозначено: P1 – 1-й (старший) разряд десятичной дроби, P4 – 4-й (младший) разряд дроби.

```
;**************************************************
; CALC – подпрограмма вычисления кода сигналов U1-U4 для вывода
; на  дисплей
```

```
; Входные параметры: (R2, R3) - СтБ, МлБ кода АЦП
; Выходные параметры: (R2, R3) – двоично-десятичное число
; с Ф3 формата (,P1P2P3P4)
;*************************************************
CALC:       MOV   R4, #03H      ; Загрузить
            MOV   R5, #0FFH     ; делитель
```

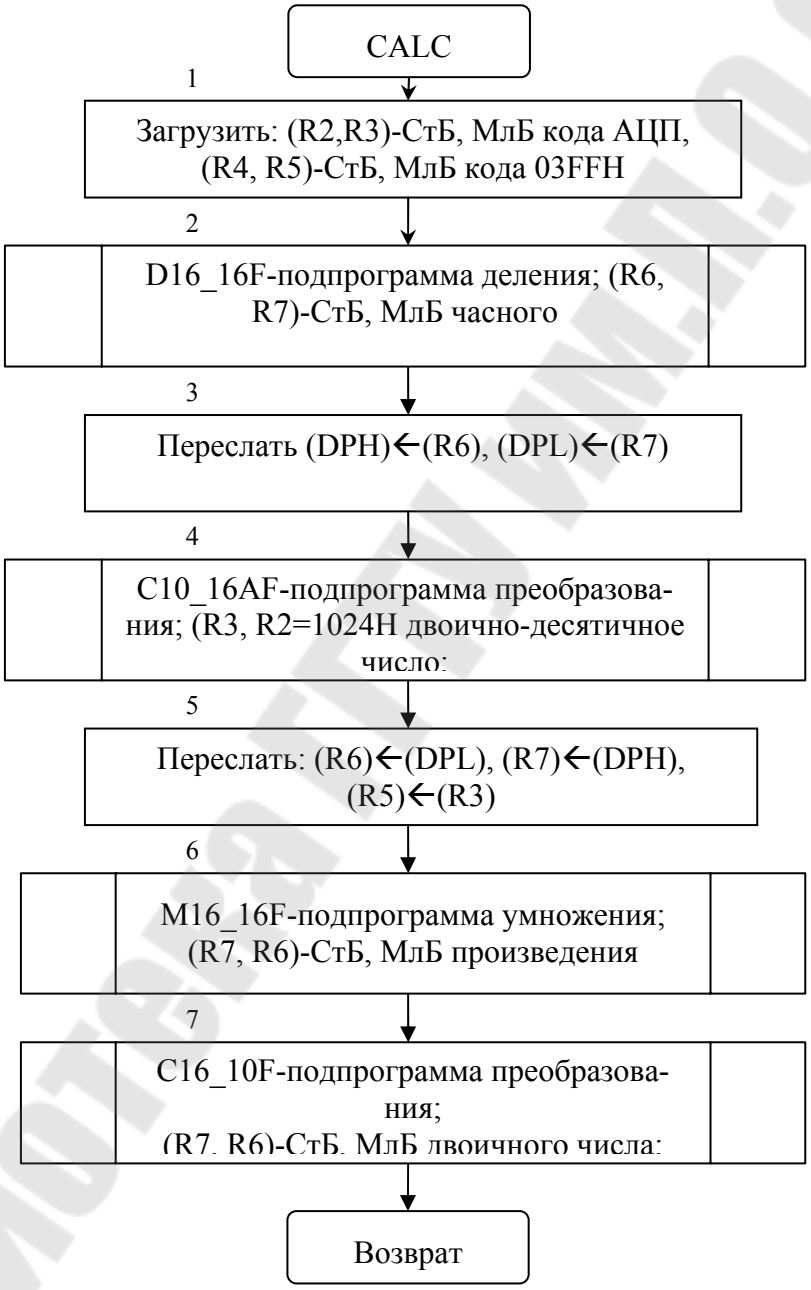


Рис. 3.24. БСА подпрограммы вычисления кода сигнала U1-U4 для вывода на дисплее

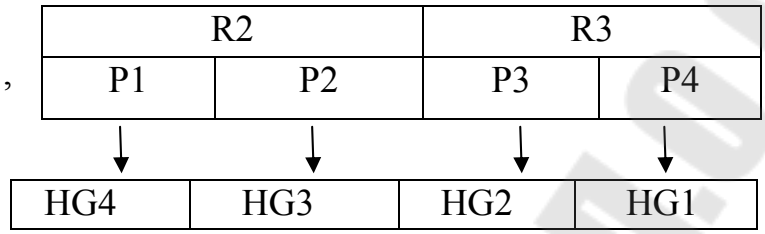
```
            CALL  D16_16F       ; Подпрограмма деления:
```

```
                    ; в (R6,R7) - СтБ, МлБ результата
        MOV     DPH, R6         ; Запоминание результата
                                ; деления
        MOV     DPL, R7         ; в регистре DPTR
; Преобразование двоично-десятичного числа с Ф3  0,1024
                ; в двоичное
        MOV    R3, #10H
        MOV    R2, #24H
        CALL   C10_16AF   ; Подпрограмма преобразования:
    ; результат в регистрах R3,R4 - СтБ, МлБ двоичного числа с Ф3
            ; Умножение двоичных чисел с Ф3
        MOV     R7, DPH       ; В регистрах
        MOV     R6, DPL       ; R7, R6 – множимое
        MOV    A, R3
        MOV    R5, A        ; В регистрах R5, R4 – множитель
        CALL   M16_16F      ; Подпрограмма умножения
    ;двоичных чисел с Ф3, результат: R7, R6 –
                    ; СтБ, МлБ произведения
; Преобразование двоичного числа с Ф3 в двоично-десятичное
        CALL   C16_10F     ; Подпрограмма преобразования,
    ;результат: в регистрах R2, R3 - двоично-десятичное число
                                ; формата    ,P1P2P3P4
        RET
```

Подпрограмма DISPLAY используется для вывода значений напряжений U1 – U4 на индикаторы HG1 – HG4 дисплея. БСА ее приведена на рис. 3.26.

В блоке 1 производится вывод в порт P0 и, следовательно, на входы регистра DD6 содержимого регистра R3, в котором размещены 3-й и 4-й разряды десятичной дроби, полученной в подпрограмме CALC. В блоках 2 и 3 производится запоминание выведенных данных в триггерах регистра DD6. В блоке 4 в порт P2 выводится содержимое регистра R2, в котором размещены 1-й и 2-й разряды десятичной дроби, полученной в подпрограмме CALC. И, наконец, в блоке 5 производится вывод семисегментных кодов десятичных цифр из дешифраторов DD9 – DD12 на индикаторы HG1 – HG4 дисплея. Вывод осуществляется подачей высокого уровня на входы разрешения E дешифраторов. Запятая высвечивается постоянно на индикаторе HG3 дисплея (см. рис. 3.8). Положение запятой соответствует умножению десятич-

ной дроби формата ,P1P2P3P4 на коэффициент 100, т.е. фактически на дисплей выводится значение P1P2,P3P4. Это соответствует значению U_i, определяемому формулой (3.6). Соответствие информации, находящейся в регистрах R2,R3 микроконтроллера и выведенной на индикаторы HG1 – HG4 дисплея можно изобразить в виде:



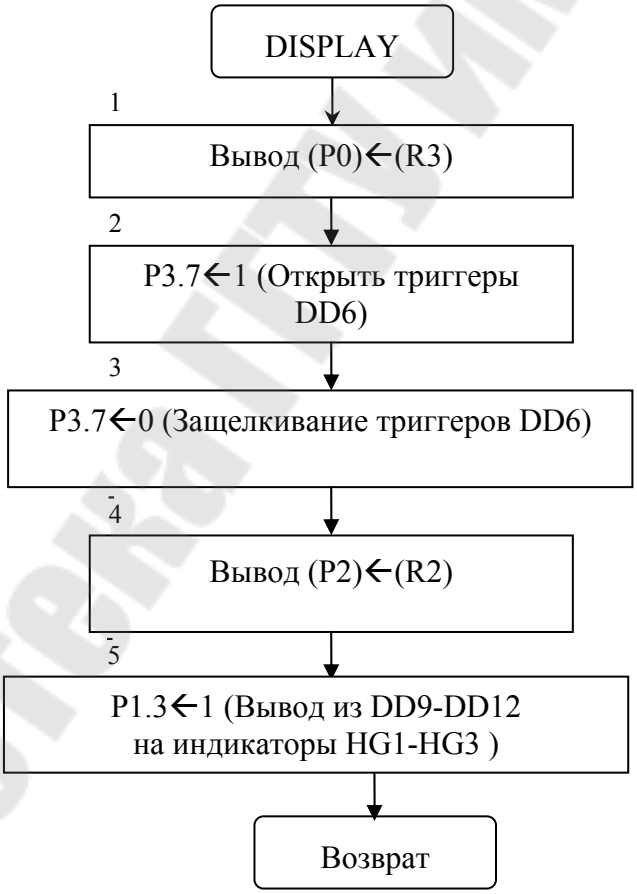


Рис. 3.26. БСА подпрограммы вывода кода U1-U4 на индикаторы дисплея

```
;*****************************************************
; DISPLAY – подпрограмма вывода сигналов U1-U4 на индикаторы HG1-HG4 дисплея
; Входные параметры: регистры (R2, R3) - двоично-десятичное число
; с Ф3 формата   ,P1P2P3P4
;*****************************************************
DISPLAY:    MOV   P0,R3    ; Вывод МлБ кода в порт P0
            SETB  P3.7     ; Открыть триггеры регистра  DD6
            CLR   P3.7     ; Защелкивание триггеров
                           ; регистра DD6
            MOV   P2,R2    ; Вывод СтБ кода в порт P2
            SETB  P1.4     ; Разрешить вывод с дешифраторов
          ; DD9 – DD12 на индикаторы HG1 – HG4 дисплея
            RET
```

В разрабатываемой МКС предусматривается связь системы с удаленным центральным компьютером по последовательному интерфейсу RS-232C или ИРПС. Оба этих интерфейса используют асинхронный режим обмена.

В микроконтроллерах семейства МК51 имеется последовательный порт, представляющий собой универсальный асинхронный приемопередатчик (УАПП), через который осуществляется прием и передача информации, представленной последовательным кодом (младшими разрядами вперед). В состав УАПП входят принимающий и передающий сдвигающие регистры, а также специальный буферный регистр SBUF приемопередатчика. Запись байта в регистр SBUF приводит к автоматической переписи байта в сдвигающий регистр передатчика и инициирует начало передачи байта. Последовательный порт может работать в 4-х режимах.

При использовании режима 1 передается через вывод P3.1 (TxD) или принимается из P3.0 (RxD) 10 бит информации: старт-бит (логический 0), 8 бит данных и стоп-бит (логическая 1). Скорость приемо/передачи – величина переменная и задается таймером T/C1.

МКС передает информацию центральному компьютеру только после получения от него запроса. Запрос вызывает прерывание выполняемой программы и переход на подпрограмму передачи данных в интерфейс. Прерывания от интерфейса разрешаются только в режиме “Работа” МКС после ввода и обработки цифровой и аналоговой информации в очередном цикле опроса $T_{опр}$ (см. рис. 2.2 и рис.3.17). За-

просом является любой байт информации, который МК принимает из интерфейса. Если этот байт успешно принят микроконтроллером, то УАПП устанавливает флаг запроса прерывания RI=1. Если прерывание разрешено (флаг ES=1), то происходит переход выполнения программы на адрес 0023H, и затем вызывается подпрограмма передачи данных в интерфейс TRANSMIT. Ее БСА приведена на рис. 3.26. Особенностью УАПП МК51 является то, что флаги запросов прерывания приемника RI и передатчика TI устанавливаются аппаратурно, но сбрасываться они должны программно.

В блоке 1 БСА производится сброс флага RI приемника. Затем в блоке 2 проверяется готовность передатчика путем опроса его флага TI. Если он готов к передаче, то флаг TI сбрасывается и в регистр SBUF пересылается байт из ячейки ПД с адресом 24H, где находятся данные о сигналах X1 – X4 и Y1 – Y3 (блоки 3 и 4). Затем в интерфейс передаются 8 байт данных из массива DATA_W, где находятся коды W1 – W4 АЦП (блоки 5 – 10 БСА). Регистр R7 используется в качестве счетчика байтов. Текст программы передачи данных:

```
;*************************************************************
; TRANSMIT – подпрограмма передачи данных в интерфейс
;*************************************************************
TRANSMIT:   CLR   RI          ; Сбросить флаг RI
WAIT1:      JNB   TI,WAIT1 ; Ожидание готовности  передатчика
            CLR    TI          ; Сбросить флаг готовности
            MOV  SBUF, 24H ; Передать байт в интерфейс
            MOV  R0, #DATA_W; Начальный адрес массива
            MOV  R7, #8        ; Счетчик байтов
WAIT2:      JNB  TI, WAIT2  ; Ожидание готовности передатчика
            CLR   TI
            MOV  SBUF, @R0
            INC  R0
            DJNZ   R7, WAIT2 ; Цикл передачи восьми байтов
            RET
```

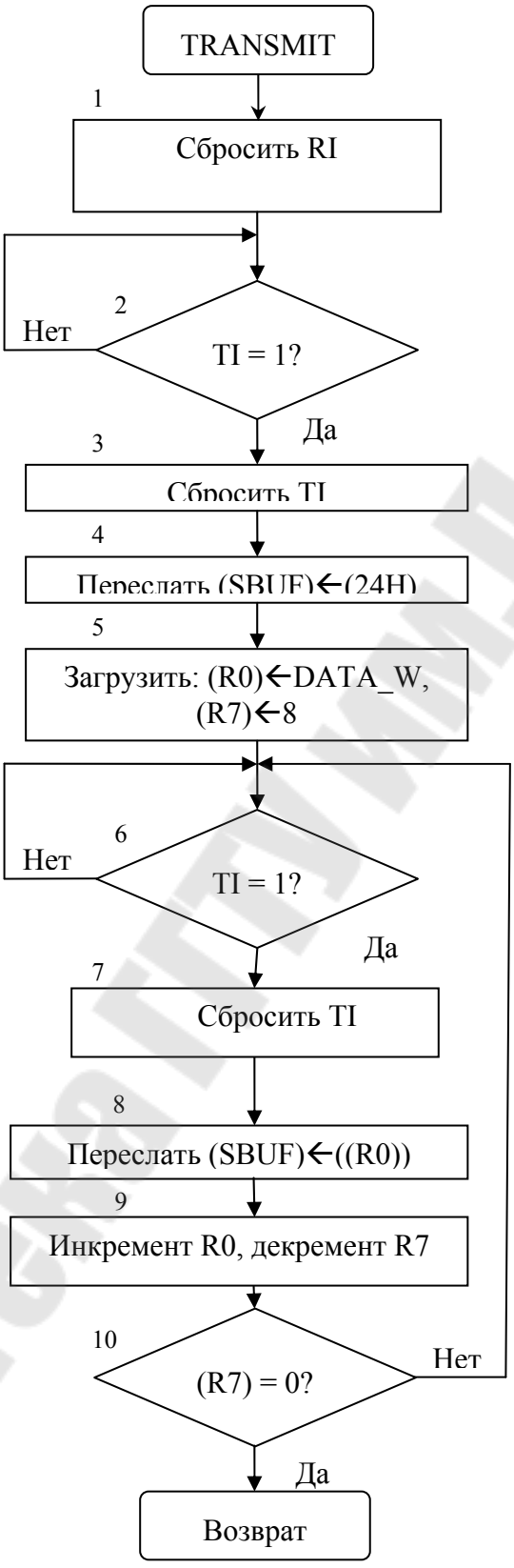


Рис. 3.26. БСА подпрограммы передачи данных в интерфейс

Для формирования управляющих сигналов Y1, Y2, Y3 в программу работы МК необходимо ввести временные задержки различной длительности. Так как оба таймера МК заняты, то для реализации временных задержек используем метод программных циклов [1]. Для задержки длительностью t1 = 80 мкс используем подпрограмму DEL, которая вызывается командой CALL DEL, ее текст:

```
DEL:    MOV    R7, #N
LOOP:   DJNZ   R7, LOOP
        RET
```

В описании команд МК51 [1,2] указывается, за сколько машинных циклов (МЦ) выполняется каждая команда: MOV R7,#N - 1 МЦ, DJNZ – 2 МЦ, RET – 2 МЦ, CALL – 2 МЦ.

Время машинного цикла $T_{МЦ}$ связано с тактовой частотой f_{CLK} работы МК соотношением:

$$T_{МЦ} = \frac{1}{f_{МЦ}} = \frac{12}{f_{CLK}} \qquad (3.7)$$

При f_{CLK} =12 МГц имеем:

$$T_{МЦ} = \frac{12}{12 \cdot 10^{6}} = 1 \text{ мкс.}$$

Время задержки

$$t_{зд} = (2 + 1 + 2 + 2 \cdot N) \cdot T_{МЦ} \qquad (3.8)$$

При $t_{зд} = t_1 = 80$ мкс имеем

$$N = \frac{80 - 5}{2} = 37.5 \approx 38.$$

Для реализации задержки длительностью t_2 =24 мс и t_3 =75 мс используем подпрограмму с двумя вложенными циклами:

```
DELAY:  MOV    R6, #EXTR
LOOP2:  MOV    R7, #INTR
LOOP1:  DJNZ   R7, LOOP1        ; Внутренний цикл
        DJNZ   R6, LOOP2        ; Внешний цикл
        RET
```

Необходимо определить значение констант INTR и EXTR, задающих времена, соответственно, $t_{внутр.цикла}$ и $t_{внеш.цикла}$. Можно записать для времени внутреннего цикла:

$$t_{внутр\ .цикла} = 2 \cdot 1 \cdot INTR \ . \qquad (3.9)$$

Примем $t_{внутр\ .цикла} = 400$ мкс, тогда

$$INTR = \frac{400}{2} = 200 \ .$$

Для времени внешнего цикла

$$t_{внеш.цикла} = (2 + 1 + 2) \cdot 1мкс + [(1 + 2) \cdot 1мкс + 400мкс] \cdot EXTR \qquad (3.10)$$

При $t_{внеш\ .цикла} = t_2 = 24\ мс$

$$EXTR = \frac{24000 - 5}{403} = 59.54 \approx 60.$$

При $t_{внеш.цикла} = t_3 = 75\ мс$

$$EXTR = \frac{75000 - 5}{403} = 186.09 \approx 186.$$

Окончательно подпрограммы временных задержек будут иметь следующий вид:

```
;**********************************************************
; DEL80MKS – подпрограмма задержки на 80 мкс
;**********************************************************
**
DEL80MKS:   MOV     R7, #38
LOOP:       DJNZ    R7, LOOP
            RET
;**********************************************************
; DEL24MS – подпрограмма задержки на 24 мс
;**********************************************************
DEL24MS:    MOV     R6, #60
LOOP2:      MOV     R7, #200
LOOP1:      DJNZ    R6, LOOP1
            DJNZ    R6, LOOP2
            RET

;**********************************************************
; DEL75MS – подпрограмма задержки на 75 мс
;**********************************************************
DEL75MS:    MOV     R6, #186
LOOP2:      MOV     R7, #200
LOOP1:      DJNZ    R7, LOOP1
```

```
        DJNZ    R6, LOOP2
        RET
```

3.8.4. Применение программной перекодировки для получения семисегментного кода

Программная перекодировка использует особую подпрограмму для получения семисегментного кода отображаемого символа на индикаторах дисплея. Примером может служить подпрограмма CROSSCOD, которая позволяет получить семисегментные коды десятичных цифр 0,1,…,9.

```
;**************************************************
; CROSSCOD – подпрограмма получения семисегментного кода
; Входной параметр: регистр A – двоичный код символа (цифры)
; Выходной параметр: регистр A – семисегментный код символа
;**************************************************
CROSSCOD:  MOV  DPTR, #TABLE ; Загрузить начальный адрес
                             ; таблицы семисегментных кодов
           MOVC  A, @A+DPTR ; Пересылка кода в аккумулятор
           RET
TABLE:     DB   3FH          ; Код цифры “0”
           DB   06H          ; Код цифры “1”
           DB   5BH          ; Код цифры  “2”
           DB   4FH          ; Код цифры  “3”
           DB   66H          ; Код цифры  “4”
           DB   6DH          ; Код цифры  “5”
           DB    7DH         ; Код цифры  “6”
           DB   07H          ; Код цифры  “7”
           DB    7FH         ; Код цифры  “8”
           DB    6FH         ; Код цифры  “9”
```

Схема подключения светодиодов индикации VD1 – VD7 и семисегментных светодиодных индикаторов HG1 – HG4 приведена на рис. 3.28. Ее особенность является то, что элементы индикации непосредственно подключены к выходам регистров DD5 – DD9, без использования инверторов с открытым коллектором. Дело в том, что регистры КР1533ИР33

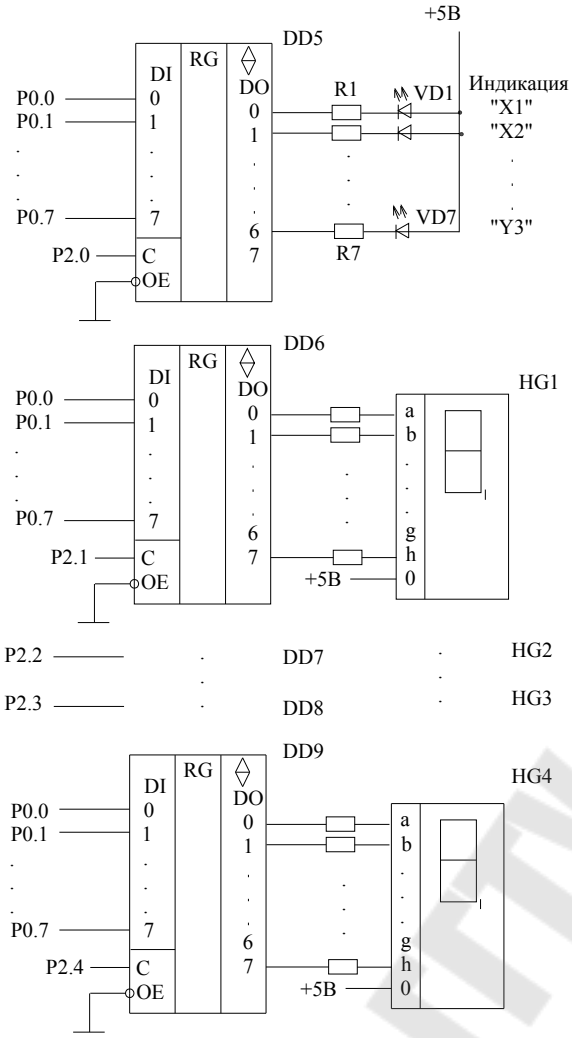


Рис.3.28. Схема подключения индикаторов при использовании программной перекодировки

обладают высокой нагрузочной способностью при низком уровне сигнала на выходах: $I_{вых} <= 20$ мА, что вполне достаточно для яркого свечения светодиодов. Однако для работы элементов индикации нужно на входы регистров DD5 – DD9 подавать инверсные значения данных с порта P0 микроконтроллера для их отображения на VD1 – VD7 и HG1 – HG4.

При использовании программной перекодировки необходимо внести коррективы в некоторые программы работы МКС, рассмотренные в п.3.8.3.

1. Нужно внести изменения в блок 2 БСА подпрограммы инициализации INIT (см. рис.3.18). Его содержание должно быть: закрыть регистры DD5 – DD6.

После сброса МК на выводах всех его портов будут высокие уровни (логические 1). Поэтому регистры DD5 –DD9 будут открыты, на их выходах – высокие уровни, светодиоды VD1 – VD7 и индикаторы HG1 – HG4 будут погашены. Для сохранения этого состояния необходимо в подпрограмме INIT закрыть триггеры регистров DD5 – DD9:

```
INIT:   ANL  P1, #11111000B  ; Вывод P1.0=P1.1=P1.2=0
        ANL  P2, #11100000B  ; Закрыть триггеры регистров
                             ;DD5 – DD9
          ; Настроить УАПП
        ……..
        …….
```

2. Нужно скорректировать блоки 2 и 9 БСА подпрограммы PULT (см. рис.3.24). Текст подпрограммы PULT должен быть:

```
PULT:   ……
         ; Вывести информацию на VD1 – VD7 (блок 2 БСА)
        MOV  A, 24H; Переслать в аккумулятор содержимое ячейки
24H
        CPL    A          ; Инвертировать аккумулятор
        MOV   P0, A       ; Вывод в порт P0
        SETB   P2.0       ; Открыть триггеры регистра DD5 – вывод
                          ; на VD1 – VD7
        CLR    P2.0       ; Защелкивание триггеров регистра DD5
        ……
        ……

         ; Погасить светодиоды и дисплей (блок 9 БСА)
        MOV   P0, #0FFH     ; Вывод единиц в порт P0
        ORL  P2, #00011111B ; Открыть триггеры регистров
                            ; DD5 – DD9,
                            ; погасить элементы индикации
        ANL   P2, #11100000B ; Защелкивание триггеров регистров
        …….
        …….
```

3. Полностью изменяется подпрограмма вывода на дисплей DISPLAY, БСА ее приведена на рис. 3.29. Текст программы будет следующий.

```
;*****************************************************
; DISPLAY – подпрограмма вывода на индикаторы дисплея при
; программной перекодировке
; Входные параметры: регистры R2,R3 – двоично-десятичное число
; с ФЗ формата  ,P1P2P3P4

;*****************************************************
DISPLAY:  MOV   A, R3       ; Пересылка МлБ кода в аккумулятор
          ANL   A, #00001111B ; Маскировать старшую тетраду
                              ; (СтТ) аккумулятора
          CALL  CROSSCOD  ; Подпрограмма перекодировки,
      ; в аккумуляторе – семисегментный код цифры разряда P4
      ;  десятичной дроби
          CPL   A           ; Инвертировать аккумулятор
          MOV   P0, A
          SETB  P2.1        ; Открыть триггеры регистра DD6 –
                            ; вывод на индикатор HG1
          CLR   P2.1        ; Защелкивание триггеров регистра DD6
          MOV   A, R3       ; Пересылка МлБ кода в аккумулятор
          SWAP  A           ; Обмен тетрад в аккумуляторе
          ANL  A,#00001111B ; Маскировать СтТ аккумулятора
          CALL  CROSSCOD ; Подпрограмма перекодировки, в
       ; аккумуляторе – семисегментный код цифры разряда P3
       ; десятичной дроби
          CPL   A           ; Инвертировать аккумулятор
          MOV   P0, A
          SETB  P2.2        ; Открыть триггеры регистра DD7 –
                        ; вывод на индикатор HG2
          CLR   P2.2   ; Защелкивание триггеров регистра DD7
          MOV   A, R2       ; Пересылка СтБ кода в аккумулятор
          ANL  A,#00001111B  ; Маскировать СтТ аккумулятора
```

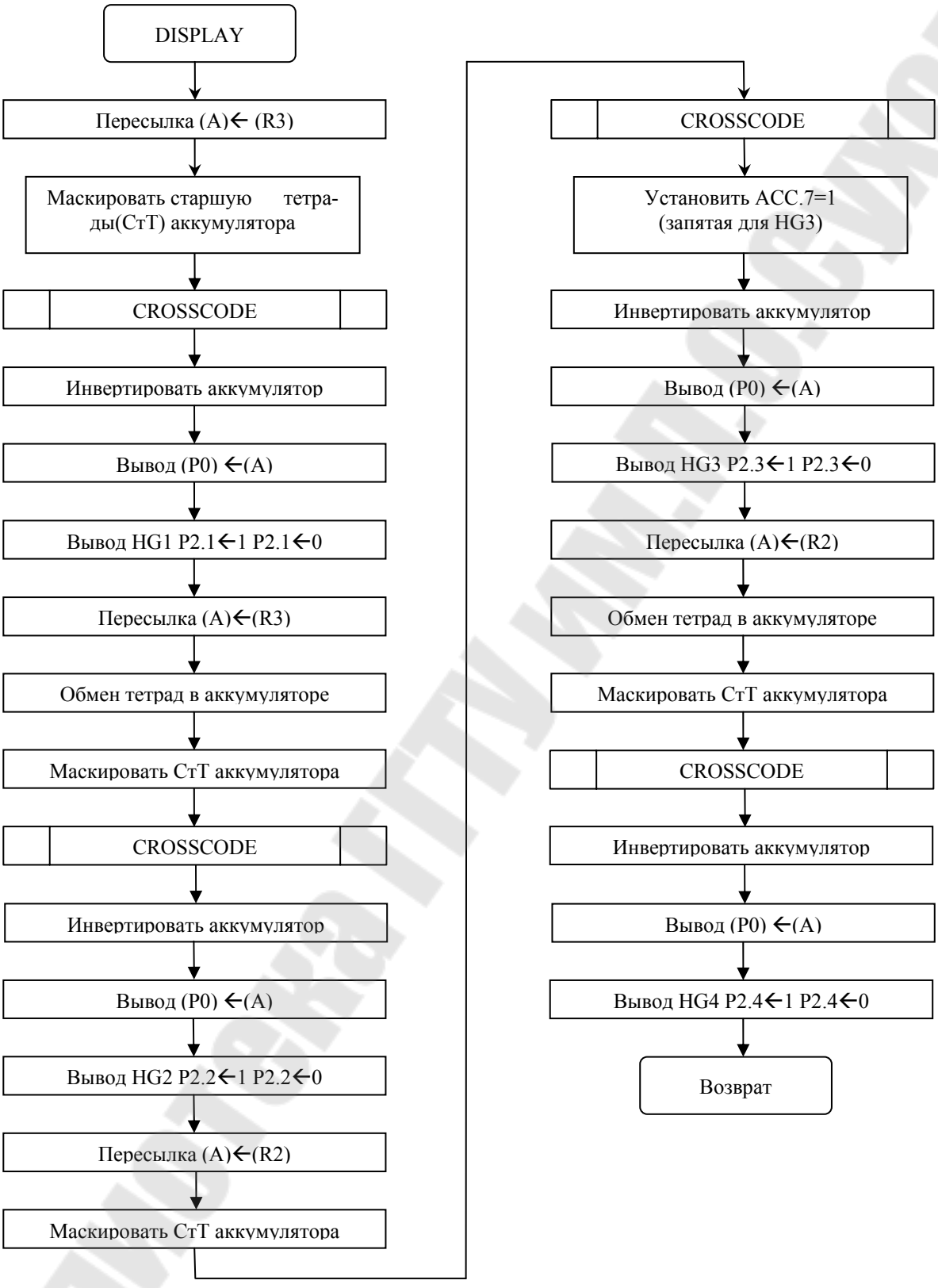


Рис.3.29. БСА вывода на индикаторы дисплея при программной перекодировке

```
        CALL  CROSSCOD  ; Подпрограмма перекодировки, в
; аккумуляторе – семисегментный код цифры разряда P2
; десятичной дроби
        SETB  ACC.7    ; Установить запятую на индикаторе HG3
        CPL    A
        SETB  P2.3          ; Открыть триггеры регистра DD8 –
                            ; вывод на индикатор HG3
        CLR  P2.3          ; Защелкивание триггеров регистра DD8
        MOV  A, R2
        SWAP  A
        ANL  A,#00001111B ; маскировать СтТ аккумулятора
        CALL  CROSSCOD  ; подпрограмма перекодировки, в
; аккумуляторе – семисегментный код цифры разряда P1
; десятичной дроби
        CPL  A
        MOV  P0, A
        SETB  P2.4       ; Открыть триггеры регистра DD9 –
                            ; вывод на индикатор HG4
        CLR    P2.4       ; Защелкивание триггеров регистра DD9
        RET
```

4. Оформление курсового проекта

Курсовой проект состоит из расчетно-пояснительной записки и графической части.

4.1. Расчетно-пояснительная записка

Расчетно-пояснительная записка должна содержать:

- Титульный лист.
- Задание на курсовой проект.
- Оглавление (содержание) с указанием страниц.
- Введение.
- Разработка структурной схемы МКС.
- Разработка принципиальной схемы МКС.
- Разработка программного обеспечения работы МКС.
- Заключение.
- Список использованной литературы.
- Приложение.

Во введении должны быть сформулированы цели и задачи курсового проекта, кратко охарактеризовано содержание работы.

Структурная схема разрабатывается в соответствии с индивидуальным заданием. Необходимо дать краткое описание состава и назначения основных элементов системы.

В разделе “Разработка принципиальной схемы МКС” необходимо разработать принципиальные схемы отдельных модулей микроконтроллерной системы: схему подключения микроконтроллера, при необходимости модуля внешней памяти программ, модуля ввода-вывода цифровой и аналоговой информации, пульта управления с элементами индикации. Нужно дать краткое описание микросхем, входящих в модули (заданных или выбранных самостоятельно). Привести их условное изображение, назначение выводов, таблицы функционирования и т.п.

В разделе “ Разработка программного обеспечения работы МКС ” необходимо привести карту распределения адресного пространства памяти: подпрограммы, промежуточные данные, стек, константы. Следует привести блок-схемы алгоритмов программ и подпрограмм, дать их краткое описание. Если блок-схемы алгоритмов будут изображены на чертеже формата А1 графической части проекта, то дублировать их в пояснительной записке не нужно. Необходимо привести тексты всех разработанных программ на языке Ассемблера. Следует привести тексты всех подпрограмм, выполняющих арифметические операции, даже если они применялись как готовые. При составлении программ надо использовать символические адреса ячеек памяти, портов и имена констант. Листинги программ на Ассемблере должны включать директивы, метки, мнемонику команд и обязательно комментарии. Перевода команд в машинные коды делать не надо.

В заключении необходимо привести основные результаты выполнения курсового проекта, параметры разработанной МКС (например, количество микросхем, объем занимаемой постоянной и оперативной памяти, напряжения и токи, требуемые от источников питания и т.п.)

В приложении помещается перечень элементов принципиальной схемы разработанной МКС.

Пояснительная записка оформляется компьютерными средствами на листах формата А4 (210×297 мм) в соответствии с требованиями ГОСТ 2.105 - 95, должна быть переплетена и подписана. Примерный объем пояснительной записки – 35-40 страниц.

4.2. Графическая часть

Графическая часть курсового проекта состоит из двух чертежей формата А1:

1. Принципиальная электрическая схема МКС.
2. Структурные схемы (блок-схемы) алгоритмов программ работы МКС.

Все элементы принципиальной схемы должны иметь нумерацию, а микросхемы - нумерацию выводов.

Примечание. Нумерация микросхем и других элементов на общей принципиальной схеме, как правило, отличается от нумераций на схемах отдельных модулей.

Графическая часть должна быть выполнена компьютерными средствами или вручную с применением чертежных инструментов и оформлена в соответствии с требованиями ЕСКД.

Список рекомендуемой литературы

1. Сташин В.В., Урусов А.В., Мологонцева О.Ф. Проектирование цифровых устройств на однокристальных микроконтроллерах. – М.: Энергоатомиздат, 1990.
2. Однокристальные микроЭВМ. Справочник/ А.В.Боборыкин, Т.П.Липовецкий и др. – М.: МИКАП, 1994.
3. Федорков Б.Г., Телец В.А., Дегтяренко В.П. Микроэлектронные цифроаналоговые и аналогоцифровые преобразователи. - М.: Радио и связь, 1985.
4. Цифровые интегральные микросхемы. Справочник / М.И.Богданович, И.Н.Грель, В.А.Прохоренко, В.В. Шалимо. – Мн.: Беларусь, 1991.
5. Цифровые и аналоговые интегральные микросхемы: Справочник / Под ред. С.В. Якубовского. - М.: Радио и связь, 1990.
6. Полупроводниковые БИС запоминающих устройств: Справочник / Под ред. А.И. Гордонова и М.Н. Дьякова. - М.: Радио и связь, 1987.
7. Лебедев О.Н. Микросхемы памяти и их применение. - М.: Радио и связь, 1990.
8. Иванов В.И.,Аксенов А.И., Юшин А.М. Полупроводниковые оптоэлектронные приборы: Справочник. - Энергоатомиздат, 1989.

9. Резисторы, конденсаторы, трансформаторы, дроссели, коммутационные устройства РЭА. Справочник / Н.Н.Акимов, Е.П.Ващуков, В.А.Прохопенко, Ю.П.Ходоренко. – Мн.: Беларусь, 1994.
10. Гуртовцев А.Л., Гудыменко С.В. Программы для микропроцессоров: Справ. пособие. – Мн.: Выш. шк., 1989.

Приложение

Программы арифметических операций для микроконтроллеров семейства МК51

Программы умножения двоичных чисел

```
; ---------------------------- M16_8 ----------------------------------------
; Подпрограмма умножения целых двоичных чисел без знака
; формата 16×8=24.
; Входные параметры: (R4,R3) – СТБ,МЛБ множимого, (R2) –
;множитель.
; Выходные параметры: (R5,R4,R3) – СТБ,СРБ,МЛБ произведения.
M16_8:      MOV   A, R2
            MOV   B, R3
            MUL   AB
            MOV   R3, A       ;  (R3) – МЛБ произведения
            MOV   R7, B
            MOV   A, R2
            MOV   B, R4
            MUL   AB
            ADD   A, R7
            MOV   R4, A       ;  (R4) – СРБ произведения
            MOV   A, B
            ADDC   A, #0
            MOV   R5, A       ;  (R5) – СТБ произведения
            RET

; ---------------------------- M16_16 ---------------------------------------
; Подпрограмма умножения целых двоичных чисел без знака
; формата 16×16=32.
; Входные параметры: (R7,R6) – СТБ,МЛБ множителя,
;                    (R5,R4) – СТБ,МЛБ множимого.
```

```
; Выходные параметры: (R7,R6,R3,R2) – СТБ,…,МЛБ произведения.
M16_16:  MOV   A, R4
         MOV   B, R6
         MUL   AB          ;  Хмл.×Yмл.
         MOV   R3, B
         MOV   R2, A
         MOV   A, R5
         MOV   B, R6
         MUL   AB          ;  Хст.×Yст.
         ADD   A, R3
         MOV   R3, A       ;  (R3)=(R3)+(A)
         CLR   A
         ADDC  A, B
         MOV   R6, A       ;  (R6)=0+(B)+C
         MOV   A, R4
         MOV   R1, A
         CLR   A
         ADDC  A, #0
         MOV   R4, A       ;  (R4)=0+C
         MOV   A, R1
         MOV   B, R7
         MUL   AB
         ADD   A, R3
         MOV   R3, A       ;  (R3)=(R3)+(A)
         MOV   A, R6
         ADDC  A, B
         MOV   R6, A       ;  (R6)=(R6)+(B)+C
         CLR   A
         ADDC  A, R4
         MOV   R4, A       ;  (R4)=(R4)+C
         MOV   A, R5
         MOV   B, R7
         MUL   AB
         ADD   A, R6
         MOV   R6, A       ;  (R6)=(R6)+(A)
         MOV   A, R4
         ADDC  A, B        ;  (A)=(R4)+(B)+C
         MOV   R7, A       ;  Результат в (R7,R6,R3,R2)
         RET
```

```
; ------------------------------ M16_16F ------------------------------------
; Подпрограмма умножения  дробных двоичных чисел без знака
; с фиксированной запятой формата 16×16=16.
; Входные параметры: (R7,R6) – СТБ,МЛБ множителя,
;                    (R5,R4) – СТБ,МЛБ множимого.
; Выходные параметры: (R7,R6) – СТБ,МЛБ произведения.
; Используется подпрограмма M16_16.
M16_16F:  CALL  M16_16  ;  (R7,R6) – произведение
          RET
```

Программы деления двоичных чисел

```
; ------------------------------ D16_8 --------------------------------------
; Подпрограмма деления целых двоичных чисел без знака
; формата 16:8=(8,8).
; Входные параметры: (R4,R3) – СТБ,МЛБ делимого, (R2) – делитель.
; Выходные параметры: (R3) – частное, (R4) – остаток.
D16_8:    MOV   B, #8          ;  Счетчик циклов
    ; Сдвиг влево остатка и частного в R4,R3
CYCLE:    MOV   A, R3
          ADD   A, R3
          MOV   R3, A
          MOV   A, R4
          ADDC  A, R4          ;  (A) – СТБ остатка
          JC    PER1           ;  Если переполнение остатка
  ; Вычитание делителя из остатка
          SUBB  A,R2
          JNC   PER2           ;  Если разность > 0
  ; Разность < 0, восстановление остатка
          ADD   A, R2          ;
          MOV   R4, A
          JMP   PER3
; Переполнение остатка, разряд частного = 1
PER1:     CLR   C
          SUBB  A, R2
PER2:     MOV   R4, A
          INC   R3             ;  +1 в частное
PER3:     DJNZ  B, CYCLE       ;  Зацикливание
          RET
```

```
; ------------------------------ D16_16 ----------------------------------------
; Подпрограмма деления целых двоичных чисел без знака
; формата 16:16 (делимое меньше делителя).
; Входные параметры: (R2,R3) – СТБ и МЛБ делимого (ДМ),
;                    (R4,R5) – СТБ и МЛБ  делителя (ДЛ).
; Выходные параметры: (R6,R7) – СТБ и МЛБ  частного (ЧСТ),
;                     (R2,R3) – СТБ и МЛБ  остатка (ОСТ).
; Используются регистры 0-го банка
D16_16:     MOV   B, #16
; Сдвиг частного (R6,R7) влево
LOOP:       MOV   A, R7
            ADD   A, R7
            MOV   R7, A
            MOV   A, R6
            ADDC   A, R6
            MOV   R6, A          ;  Сдвинутое частное (R6,R7)
; Сдвиг остатка (промежуточного делимого) (R2,R3) влево
            MOV   A, R3
            ADD   A, R3
            MOV   R3, A
            MOV   A, R2
            ADDC   A, R2
            MOV   R2, A          ;  (R2,R3) – сдвинутый остаток
            JC   PER1            ;  Если переполнение остатка (C=1)
; Сложение остатка с дополнительным кодом делителя
            PUSH   03H           ;  Сохранение МЛБ остатка – R3
            PUSH   02H           ;  Сохранение СТБ остатка – R2
            MOV   A, R3
            ADD   A, R5
            MOV   R3, A
            MOV   A, R2
            ADDC   A, R4
            MOV   R2, A
            JNC   PER2           ;  Переход если сумма < 0
; Сумма > 0, в разряд частного записать 1
            DEC   SP             ;
            DEC   SP             ;  Баланс стека
```

```
                MOV   A, R7
                ADD   A, #1
                MOV   R7, A
                MOV   A, R6
                ADDC   A, #0
                MOV   R6, A
                JMP   PER3
; Переполнение остатка, в разряд частного записать 1
PER1:           MOV   A, R3
                ADD   A, R5
                MOV   R3, A             ;  МЛБ нового остатка
                MOV   A, R2
                ADDC   A, R4
                MOV   R2, A             ;  СТБ нового остатка
                MOV   A, R7
                ADD   A, #1
                MOV   R7, A
                MOV   A, R6
                ADDC   A, #0
                MOV   R6, A
                JMP   PER3
; Сумма < 0, восстановление остатка
PER2:           POP   02H               ;  (R2) – СТБ остатка
                POP   03H               ;  (R3) – МЛБ остатка
; Проверка конца цикла
PER3:           DJNZ   B, LOOP
                RET

; ------------------------------ D16_16F----------------------------------------
; Подпрограмма деления дробных двоичных чисел без знака с ФЗ
; формата 16:16=16.
; Входные параметры: (R2,R3) – СТБ,МЛБ делимого,
;                    (R4,R5) – СТБ,МЛБ  делителя.
; Выходные параметры: (R6,R7) – СТБ,МЛБ  частного.
; Используется подпрограмма D16_16.
D16_16F:    CALL   D16_16       ;  (R6,R7) – частное
            RET
```

Программы преобразования представления чисел

```
; ---------------------------- C10_16A ----------------------------------------
; Подпрограмма преобразования целого беззнакового двоично-
десятичного
; числа формата 4*4 в двоичное число формата 16.
; Входные параметры: (R3,R2) – двоично-десятичное число
;(P3P2P1P0).
; Выходные параметры: (R7,R6) – СТБ, МЛБ двоичного числа.
C10_16A:  CLR   A              ;  Обнуление аккумулятора
          MOV   R6, A
          MOV   R7, A
          MOV   R5, #0FH
          MOV   A, R3
          CALL   PPL1          ;  Q1=(P4×10+P3)×10
          MOV   A, R3
          ANL   A, R5
          CALL   PPL           ;  Q2=(Q1+P2)×10
          MOV   A, R2
          CALL   PPL1          ;  Q3=(Q2+P1)×10
; Вычисление Q4=Q3+P0
          MOV   A, R2
          ANL   A, R5
          ADD   A, R6
          MOV   R6, A
          MOV   A, R7
          ADDC   A, #0
          MOV   R7, A
          RET
PPL1:     ANL   A, #0F0H
          SWAP   A
PPL:      ADD   A, R6
          MOV   R6, A
          MOV   A, R7
          ADDC   A, #0
          MOV   R7, A
          CLR   C
          CALL   P10A          ;  Умножение на 2
          MOV   R0, A
          MOV   A, R7
          MOV   R1, A
```

```
            CALL   P10A
            CALL   P10A
            ADD    A, R0
            MOV    R6, A
            MOV    A, R7
            ADDC   A, R1
            MOV    R7, A
            RET
P10A:       MOV    A, R6
            RLC    A
            XCH    A, R7
            RLC    A
            XCH    A, R7
            MOV    R6, A
            RET
```

; ---------- **C16_10F** ----------

; Подпрограмма преобразования дробного двоичного беззнакового
;числа с ФЗ формата 16 в двоично-десятичное число формата 4×4.
; Входные параметры: (R7,R6) – СТБ,МЛБ двоичного числа.
; Выходные параметры: (R2,R3) – двоично-десятичное число
;(,P1P2P3P4).
; Используется подпрограмма M16_10F.

```
VREM:       EQU    20H          ;  Адрес ПД для временного хранения
C16_10F:    CLR    F0
            CALL   M16_10F
            PUSH   ACC          ;  Сохранение цифры P1 (С.ц. P1)
            CALL   M16_10F
            PUSH   ACC          ;  С.ц. P2
            CALL   M16_10F
            PUSH   ACC          ;  С.ц. P3
            CALL   M16_10F
            PUSH   ACC          ;  С.ц. P4
            CALL   M16_10F
            MOV    R7, A        ;  (R7)=P5
```

; Упаковка цифр результата в регистры

```
            POP    ACC          ;  Восстановление цифры P4 (В.ц. P4)
            SWAP   A
            ORL    A, R7
            MOV    R7, A        ;  (R7)=P4P5
```

```
POP   ACC          ;  В.ц.  P3
MOV   R6, A
POP   ACC          ;  В.ц.  P2
SWAP   A
ORL   A, R6
MOV   R6, A        ;  (R6)=P2P3
POP   ACC          ;  (A)=P1
MOV   R5, A        ;  Сохранение A
MOV   A, R7
ANL   A, #0FH
ADD   A, #0FAH
MOV   A, R5
SWAP  A
MOV   DPL, R0
MOV   R0,
#VREM
MOV   @R0, A
MOV   A, R6
SWAP   A
XCHD   A, @R0
MOV   A, @R0
MOV   R2, A
MOV   A, R6
MOV   @R0, A
MOV   A, R7
SWAP   A
XCHD   A, @R0
SWAP   A
MOV   R3, A
MOV   A, R7
SWAP   A
MOV  A, @R0
MOV   A, R3
XCHD   A, @R0
MOV   R3, A
JNC   CON
CLR   A
ADDC   A, R3
DA   A
```

```
                MOV   R3, A
                CLR   A
                ADDC   A, R2
                DA   A
                MOV   R2, A
CON:            MOV   R0, DPL
                RET

; ------------------------ M16_10F ----------------------------------------------
; Подпрограмма умножения дробного двоичного беззнакового числа с
;ФЗ формата 16 на основание десятичной системы 10.
; Входные параметры: (R7,R6) – СТБ,МЛБ  двоичного числа.
; Выходные параметры: (A) – целая часть произведения,
;                          (R7,R6) – дробная  часть произведения.
M16_10F:    MOV   A, R6
                MOV   R2, A
                MOV   R4, A
                MOV   A, R7
                MOV   R3, A
                MOV   R5, A
                CLR   A
                MOV   R6, A
                MOV   R7, A
; Сдвиг числа с ФЗ влево на 1 разряд в (R7,R5,R4)
                MOV   A, R4
                ADD   A, R4
                MOV   R4, A
                MOV  A, R5
                ADDC   A, R5
                MOV   R5, A
                CLR   A
                ADDC   A, R7
                MOV   R7, A             ;   (R7,R5,R4) – число с ФЗ×2
; Сдвиг числа с ФЗ влево на 3 разряда в (R6,R3,R2)
                MOV   A, R2             ;   Первый
                ADD   A, R2             ;
                XCH   A, R4             ;
                MOV   R2, A             ;
                MOV   A, R3             ;
```

```
        ADDC   A, R3        ;
        XCH   A, R5         ;
        MOV   R3, A         ;  сдвиг
        MOV   A, R6
        RLC   A
        MOV   R6, A
        MOV   A, R4         ;  Второй
        ADD   A, R4         ;
        MOV   R4, A         ;
        MOV   A, R5         ;
        ADDC   A, R5        ;
        MOV   R5, A         ;  сдвиг
        MOV   A, R6
        RLC   A
        MOV   R6, A
        MOV   A, R4         ;  Третий
        ADD   A, R4         ;
        MOV   R4, A         ;
        MOV   A, R5         ;
        ADDC   A, R5        ;
        MOV   R5, A         ;  сдвиг
        MOV   A, R6
        RLC   A
        MOV   R6, A
        MOV   A, R4
        XCH   A, R2
        MOV   R4, A
        MOV   A, R5
        XCH   A, R3
        MOV   R5, A
; Сложение сдвинутых чисел: умножение на 10
        MOV   A, R4
        ADD   A, R2
        MOV   R4, A
        MOV   A, R5
        ADDC   A, R3
        MOV   R5, A
        MOV   A, R7
        ADDC   A, R6
```

```
                MOV   R2, A
                MOV   A, R5
                MOV   R7, A
                MOV   A, R4
                MOV   R6, A
                MOV   A, R2
                RET

; ------------------------------ C10_16AF ------------------------------------
; Подпрограмма преобразования дробного двоично-десятичного
; беззнакового числа с ФЗ формата 4*4 в двоичное число с ФЗ
;формата 16.
; Входные параметры: (R3,R2) – двоично-десятичное число
;
 ( ,P1P2P3P4).
; Выходные параметры: (R3,R4) – СТБ,МЛБ двоичного числа.
; Используется подпрограмма C10_16A.
KL:             EQU   22H               ;  Ячейка для временного хранения
; Двоичное преобразование числа (0,P1P2P3P4)*10*4
C10_16F:  CALL   C10_16A     ;  (R7,R6) – двоичное число
; Деление двоичного целого числа на 10**4=2710H
                MOV   R3, #27H
                MOV   R2, #10H
                MOV   R4, #0
                MOV   R5, #0
; Дополнение делителя
                MOV   A, R2
                CPL   A
                ADD   A, #1
                MOV   R2, A
                MOV   A, R3
                CPL   A
                ADDC   A, #0
                MOV   R3, A
                MOV   R1, #16         ;  Счетчик циклов
; Сдвиг частного (R5,R4) и остатка (R7,R6) влево
RED:        MOV   A, R4
                ADD   A, R4
                MOV   R4, A
```

```
        MOV   A, R5
        ADDC   A, R5
        MOV   R5, A
        MOV   A, R6
        ADD   A, R6
        MOV   R6, A
        MOV   A, R7
        ADDC   A, R7
        MOV   R7, A
        JC  MET1            ;  Если переполнение остатка
; Сложение остатка с дополнительным кодом делителя
        MOV   R0, #KL       ;  Сохранение
        MOV   A, R6         ;
        MOV   @R0, A        ;
        INC   R0            ;
        MOV   A, R7         ;
        MOV   @R0, A        ;
        MOV   A, R2         ;  остатка
        ADD   A, R6
        MOV   R6, A
        MOV   A, R3
        ADDC   A, R7
        MOV   R7, A
        JNC   MET2
; Сумма >0, разряд частного=1
        DEC   R0
        MOV   A, R4
        ADD   A, #1
        MOV   R4, A
        MOV   A, R5
        ADDC   A, #0
        MOV   R5, A
        JMP   MET3
; Переполнение остатка, разряд частного=1
MET1:   MOV   A, R2
        ADD   A, R6
        MOV   R6, A
        MOV   A, R3
        ADDC   A, R7
```

```
                MOV   R7, A
                MOV   A, R4
                ADD   A, #1
                MOV   R4, A
                MOV   A, R5
                ADDC   A, #0
                MOV   R5, A
                JMP   MET3
;Сумма <0, восстановление остатка
MET2:     MOV   A, @R0
                MOV   R7, A
                DEC   R0
                MOV   A, @R0
                MOV   R6, A
; Проверка конца цикла
MET3:     DJNZ   R1, RED
                RET
```

Оглавление

Виноградов Эдуард Михайлович

ПРОЕКТИРОВАНИЕ МИКРОКОНТРОЛЛЕРНОЙ СИСТЕМЫ УПРАВЛЕНИЯ

Методические указания
к курсовому проекту
по дисциплине «Микропроцессорная техника»
для студентов специальности 1-36 04 02
«Промышленная электроника»
дневной и заочной форм обучения

Подписано в печать 26.10.09.
Формат 60x84/16. Бумага офсетная. Гарнитура «Таймс».
Ризография. Усл. печ. л. 5,35. Уч.-изд. л. 4,4.
Изд. № 134.
E-mail: ic@gstu.gomel.by
http://www.gstu.gomel.by

Отпечатано на цифровом дуплекаторе
с макета оригинала авторского для внутреннего использования.
Учреждение образования «Гомельский государственный
технический университет имени П. О. Сухого».
246746, г. Гомель, пр. Октября, 48.