Министерство образования Российской Федерации
Владимирский государственный университет
Муромский институт (филиал)

Н.П.МОЛЬКОВ, В.В.ФАЛИН

МИКРОПРОЦЕССОРЫ И МИКРО-ЭВМ В РЭС

Учебное пособие для специальности 200800

Муром 2000

Авторы: Н.П.Мольков (разд.1,2,3,5), В.В. Фалин (введение, разд.4)

УДК 681.58:681.32

Микропроцессоры и микро-ЭВМ в РЭС: Учеб. Пособие для специальности 200800 / Н.П. Мольков, В.В. Фалин: Муромский инст. Владим. гос. ун-та. – Муром, 2000. – 122с.

Рассматриваются вопросы разработки и применения микропроцессорных устройств в РЭС. Представлены необходимые теоретические сведения о наиболее распространенных микропроцессорах и однокристальных микро-ЭВМ, особенностях их архитектуры. Приведена классификация датчиков, освещены вопросы их применения в МПУ. На примерах рассмотрены особенности программирования МПУ на языке ассемблера.

Предназначено для студентов специальности 200800 – Проектирование и технология радиоэлектронных средств.

Табл. 29. Ил. 37. Библиогр.: 15 назв.

Печатается по решению редакционно-издательского совета Муромского института Владимирского государственного университета.

Рецензенты: кафедра «Конструирование и технология радиоэлектронных средств» (Владимирский государственный университет)

ВВЕДЕНИЕ

Развитие микроэлектронной технологии, обеспечившее появление больших (БИС) и сверхбольших интегральных схем (СБИС), создало предпосылки для резкого снижения стоимости систем управления. Но практическая реализация достижений технологии БИС и СБИС в широких масштабах стала возможной только с изобретением микропроцессора (МП) — универсального компонента, который стал связующим звеном для нового поколения изделий микроэлектроники (БИС и СБИС) и обеспечил его функциональную полноту.

Огромную популярность завоевали однокристальные микроЭВМ (ОЭВМ), объединяющие на одном полупроводниковом кристалле, как сам микропроцессор, так и целый ряд дополнительных устройств, обеспечивающих его функционирование в системе управления: оперативную и программную память, генератор синхроимпульсов, разнообразные устройства ввода-вывода информации и др. Благодаря низкой стоимости систем управления, основанных на ОЭВМ (в простейших случаях для создания системы достаточно одной БИС), более двух третей мирового рынка МП составили ОЭВМ.

В настоящее время развитие МП достигло такого уровня, что сферы применения МП можно считать практически неограниченными, поскольку в любой из них могут быть реализованы потенциальные возможности микропроцессорных систем управления. Это относится как к простейшим объектам управления — датчикам, исполнительным устройствам, регистраторам информации, так и к самым сложным станкам или приборам.

Решение задачи оптимального управления промышленными объектами (технологическим оборудованием, устройствами и про-

цессами, а также параметрами и данными, характеризующими их) возможно лишь на основе широкого внедрения в различные отрасли промышленности микропроцессорных контроллеров и систем управления. Управление технологическим оборудованием, устройствами и процессами относится к нижнему уровню управления производством. Наибольший выигрыш от применения МП и ОЭВМ достигается именно на этом уровне, от эффективности его работы, прежде всего, зависит производительность труда и качество продукции.

Необходимо отметить, что преимущества МП-контроллеров и систем не обеспечиваются сами собой, автоматически, а являются результатом глубокого анализа, грамотной постановки и решения задачи на основе разумных компромиссов, оптимального выбора типа МП или ОЭВМ и элементной базы в целом [1]. Обязательным условием эффективного применения МП и ОЭВМ является использование средств комплексной отладки [2, 3], являющихся необходимым инструментом инженера-разработчика аппаратурных и программных средств МП-контроллеров и систем и значительно сокращающих цикл их разработки.

Подводя итог, можно сказать, что современный специалист разработчик новой аппаратуры немыслим без знаний современных МП и микро-ЭВМ. Он должен уметь правильно оценивать возможности современных МП, видеть их место в конкретном применении и свободно владеть программированием на языке ассемблера, являющегося основным языком программирования современных микроконтроллеров.

Предлагаемое учебное пособие предназначено для студентов радиотехнических специальностей занимающихся проектированием РЭА с применением средств вычислительной техники.

1. ОБЩИЕ СВЕДЕНИЯ, ТЕРМИНОЛОГИЯ, ФУНКЦИИ И ЗАДАЧИ ВЫПОЛНЯЕМЫЕ МП В РЭС

Рассмотрим основные понятия и определения, принятые в микропроцессорной технике и используемые в данной книге [1-7,].

Микропроцессор (МП) - программно-управляемое устройство, осуществляющее процесс обработки цифровой информации и управление им, построенное на одной или нескольких БИС.

Микропроцессорная БИС - интегральная микросхема, выполняющая функцию МП или его части. По существу - это БИС с процессорной организацией, разработанная для построения микропроцессорных систем.

Микропроцессорный комплект (МПК) - совокупность микропроцессорных и других интегральных микросхем, совместимых по конструктивно-технологическому исполнению и предназначенных для совместного применения при построении МП, микро-ЭВМ и других средств вычислительной техники.

Микропроцессорная электронная вычислительная машина (микро-ЭВМ) - ЭВМ, состоящая из микропроцессора (микропроцессоров), полупроводниковой памяти, средств связи с периферийными устройствами и при необходимости пульта управления и источника питания, объединенных общей несущей конструкцией. МикроЭВМ — универсальное вычислительное устройство с законченной конструкцией и развитым стандартным программным обеспечением.

Однокристальная микро-ЭВМ **-** микро-ЭВМ, выполненная в виде одной БИС. В этом случае на одном кристалле размещаются процессор, постоянное запоминающее устройство (ПЗУ) для хранения программы, оперативное запоминающее устройство (ОЗУ) для хранения

промежуточных результатов, каналы ввода — вывода, в ряде случаев таймер.

Одноплатная микро-ЭВМ - микро-ЭВМ, выполненная в виде одной печатной платы, предназначенная для встраивания в радиоэлектронную аппаратуру. Как правило, однокристальные и одноплатные микро-ЭВМ допускают подключение дополнительных плат памяти и контроллеров устройств ввода - вывода.

Микропроцессорная система - управляющая, информационная или иная специализированная цифровая система, построенная на базе микропроцессорных средств и включающая микро-ЭВМ и средства сопряжения с обслуживаемым объектом.

Алгоритм **–** конечный набор предписаний, определяющий решение задачи посредством конечного количества операций или схематическое изображение последовательности шагов реализующих поставленную задачу.

Программное обеспечение микропроцессорной системы (микро-ЭВМ) — последовательность команд (программа или набор программ), размещенная в ЗУ микропроцессорной системы и реализующая требуемый алгоритм ее функционирования.

Исходная программа **–** текстовый файл, написанный на выбранным языке программирования с применением текстовых редакторов, для выполнения на МП требуется трансляция в машинные коды.

Транслятор – специальная системная программа, преобразующая исходную программу в объектную программу.

Объектная программа – программа в машинных кодах предназначенная для загрузки в ПЗУ МП и выполнения её.

Таким образом, микропроцессорная система - это комплекс технических средств и программного обеспечения. Области применения

микропроцессоров и микро-ЭВМ постоянно расширяются. Применительно к РЭС можно выделить три основные области: математическая обработка сигналов, автоматизация измерительных приборов и систем, автоматизация управления технологическими процессами. Сточки зрения конструктора наиболее важными являются две последние.

Применение микропроцессоров (МП) преобразует измерительные приборы в «интеллектуальные» устройства, способные производить необходимую математическую обработку измерительной информации и представлять ее в наиболее удобном для восприятия виде.

Кроме задач обработки измеряемых параметров микропроцессор выполняет функции управляющего устройства, обеспечивающего подключение необходимых элементов приборов, прием командных сигналов, передачу выходных данных и др. В зависимости от типов измерительного прибора и примененного микропроцессора возможна различная структура «интеллектуального» прибора.

В приборах для измерения электрических и неэлектрических величин микропроцессоры выполняют следующие основные функции:

1) автоматическую установку пределов измерения, корректировку аддитивных и мультипликативных погрешностей;

2) автоматическое управление процессом уравновешивания в приборах сравнения постоянного и переменное токов;

3) первичную обработку данных - определение отклонений от номинальных значений, определение момента приближения к граничным условиям, вычисление отношений максимума-минимума, именованных значений отсчетов умножение и деление на константы;

4) статистическую обработку данных: определение средних значений контролируемых величин за определенные интервалы времени,

вычисление вариаций, дисперсий средних квадратических значений и др.;

5) обработку данных по упрощенным алгоритмам определение контролируемых параметров по измеренным значениям и известным зависимостям, сглаживание полученных значений и др.;

6) обработку данных: по алгоритмам, реализующие метод измерения: определение скоростей движения и значений расхода на основе корреляционных методов, определение параметров объекта на основе спектрального анализа сигналов и др.;

7) регистрацию данных в буферных (транзитных) регистраторах: управление частотой отсчетов, рациональное использование буферной памяти, подготовку данных к передаче в блоки основной регистрации;

8) визуализацию и регистрацию данных на осциллографах и дисплеях: управление процессом визуализации организацию памяти, формирование знаков, управление цветом, формирование маркерных меток и др.;

9) диагностику функциональных узлов приборов: определение перед началом измерения исправности основные узлов сложных приборов, организацию тестов контроля индикацию неисправностей;

10) управление работой узлов, выполняющих отдельные функции измерительного преобразования, в частности работой аналого-цифрового преобразователя и др.;

11) полное управление процессом измерения по заданной программе, включая управление внешними блокам и дополнительными устройствами (в том числе переключателями, вентилями, микродвигателями), для приборов, измеряющих неэлектрические величины.

В приборах, входящих в измерительную систему, микропроцессо-

ры используются также для связи приборов в единый комплекс, кодирования и декодирования данных, передаваемых по каналам связи, повышения надежности системы путем защиты данных от искажений, сжатия данных и других задач, характерных для информационно-измерительных систем.

При применении МП для автоматизации технологических процессов нижнего уровня позволяет освободить оператора от рутинной работы, повысить качество регулирования и снизить себестоимость выпускаемой продукции. К ТП процессам нижнего уровня можно отнести процессы:

1) входного контроля качества радиоэлементов;

2) поддержание значения одного или нескольких параметров ТП в заданном диапазоне (например, температуры, давления, оборотов двигателей, напряжения, частоты, освещенности и.т.д.), в пределах одного рабочего места;

3) управления отдельными операциями (сборки, контроля качества сборки, настройки, диагностики неисправностей);

4) управление отдельными инструментами и манипуляторами.

В общем виде области применения МП можно сгруппировать в таблицу 1 [8].

Стремительное развитие средств вычислительной техники привело к тому, что применение МП становится рентабельным даже в том случае если его вычислительные мощности загружены всего на несколько процентов. Это даёт возможность применять МП там, где ранее о МП не могла идти речь. Одной из задач современного конструктора РЭС является - уметь видеть эти возможности МП и эффективно их использовать.

Таблица 1

Классификация областей применения МП и ОМЭВМ в промышленности

Управление оборудованием, устройствами	Управление процессами	Управление параметрами	Управление данными
Станками (металлообрабатывающими, намоточными, вязальными...)	Перемещением (сырья, заготовок изделий)	Температурой, (плавления, испарения, термостата, помещений,..)	Сбором (линеаризацией характеристик датчиков, масштабированием и усреднением значений параметров...)
Роботами	Дозировкой (жидкостей, .газов, сыпучих материалов)	Давлением (жидкой, газовой фазы)	Хранением (вводом-выводом на магнитные, полупроводниковые носители...)
Генераторами (электрическими, лазерными, ультразвуковыми...)	Сортировкой (по габаритам, массе, электрическим и магнитным параметрами.	Расходом (жидкостей, газов, сыпучих материалов)	
Преобразователями (одного параметра в другой)	Упаковкой (готовой продукции)	Уровнем (жидкостей, сыпучих материалов)	
Распознавателями (текстовыми, речевыми...)	Раскроем (различных материалов)	Скоростью (линейной, угловой)	Обработкой (цифровой фильтрацией, сортировкой, преобразованием...)
Электроприводами (постоянного и переменного тока, шаговыми...)	Сборкой (БИС, деталей, приборов...)	Влажностью (сырья, материалов, помещений...)	
Гидроприводами	Формообразованием (литьем, штамповкой, прокаткой, резкой, сверлением...)	Током	Регистрацией (на магнитных, бумажных носителях...)
Пневмоприводами	Сваркой (ультразвуковой, дуговой. ..)	Напряжением	
Ректорами (ядерными, химическими...)	Пайкой (индукционной, инфракрасной...)	Мощностью	Передачей (коммутацией, мультиплексированием...)
Печами (плавильными, кристаллизационными, диффузионными…)	Нанесением (красок, паст, пленок...)	Частотой	

2. ПРОГРАММИРОВАНИЕ МИКРОПРОЦЕССОРОВ И МИКРО-ЭВМ

2.1. Разработка алгоритмов программы

Разработка алгоритма, является одним из основных этапов в процессе разработки программного обеспечения. Особенностью программного обеспечения микроконтроллеров является то, что для их написания используется язык низкого уровня – Ассемблер. Это связано с тем, что вычислительные ресурсы и объём памяти обычно сильно ограничены, к тому же большинство контроллеров работают в режиме реального времени. Все это предъявляет высокие требования к быстродействию и объёму программ. Часто Ассемблер является единственно приемлемым языком программирования микроконтроллеров. Особенно это актуально при автоматизации технологических процессов нижнего уровня, где функции МП ограничиваются управлением несколькими операциями.

Сложность программирования на Ассемблере заключается в том, что разработчик должен детально представлять программную структуру проектируемого устройства, хорошо знать используемый микропроцессор, его архитектуру и систему команд. Программирование на уровне команд значительно отличается от программирования на языках высокого уровня, где написание программы производится на уровне операторов и функций, включающих десятки и сотни команд МП и значительно более понятных разработчику.

Перечисленные обстоятельства приводят к тому, что студенты впервые приступающие к программированию на Ассемблере встречаются с большими трудностями, которые выражаются в том, что просто не знают с чего начать. Разработка программы по готовому алгоритму протекает значительно легче и сводится к подбору команд

МП реализующих блок алгоритма. Таким образом, разработка алгоритма является основным этапом при программировании микроконтроллеров.

Для того чтобы упростить этот процесс и сделать его более понятным процесс разбивают на ряд последовательных этапов, постепенно детализирующих алгоритм, приводя его к уровню программ. Можно выделить три основных этапа: разработка концептуальной блок – схемы алгоритма, разработка функциональной блок - схемы и разработка структурной схемы машинных команд [6,7].

Первые два этапа пишутся безотносительно, к какому либо микропроцессору, т.е. при их разработке не используются команды микропроцессора, но если эта возможность имеется, то это может ускорить процесс разработки программы. Третий этап реализует детализированный алгоритм в командах выбранного микропроцессора. Рассмотрение процедуры написание алгоритма проведем на примере конкретной задачи.

2.2. Основные этапы разработки алгоритмов

При разработке алгоритма любого уровня можно следовать следующей методике. В любой задаче можно выделить основные действия и вспомогательные или подготовительные. Основными действиями являются действия направленные на выполнение функционально-законченного блока (в концептуальной блок – схеме) или команды преобразования арифметического или логического (в функциональной и структурной схеме машинных команд). Вспомогательными являются действия и команды, обеспечивающие выполнение основных. К ним можно отнести ввод- вывод информации, индикацию и управ-

ление (в концептуальной блок – схеме), команды пересылки, загрузки, сохранения (в функциональной и структурной схеме машинных команд). При переходе от концептуальной схемы к функциональной следует помнить, что вспомогательные блоки концептуальной также будут состоять из основных и вспомогательных команд. Подобный формальный подход к разработке алгоритма позволяет упростить процесс программирования, особенно на этапе начального обучения программированию на ассемблере.

Для примера разработаем микроконтроллер для управления автоматическим термонагревателем, работающем в диапазоне температур от 0 до 100^0C. и поддерживающим температуру с точностью 1^0C в диапазоне задаваемым однобайтовыми числами, вводимыми через внешний порт.

Разработка концептуальной блок – схемы алгоритма. Выделим основные и вспомогательные действия. К основным действиям можно отнести: сравнение текущей температуры с предельными значениями диапазона, управление нагревателем. Вспомогательными действиями на этом этапе будут – измерение текущей температуры, чтение значения нижнего предела, чтение значения верхнего предела, формирование и выдача управляющих слов для управления нагревателем. Концептуальная блок – схема алгоритма приведена на рисунке 2.1. При разработке и оформлении алгоритма необходимо руководствоваться требованиями ЕСКД [9].

Блоки 1 и 2 осуществляют подготовку операндов для их сравнения, операция сравнения, как и большинство операций в микроконтроллерах выполняется с двумя операндами, при большем числе операндов сравнение выполняется последовательно со всеми – операции 4, 5. После выполнения основных операций 3, 5 на основе ана-

лиза результата принимается решение либо перейти к следущей по порядку операции либо выполнить операцию 6 или 7. После выполнения всех действий производится считывание очередного значения текущей температуры $t_{тек.}$- блок 1. Порядок выполнения сравнений в данном случае не имеет принципиального значения.

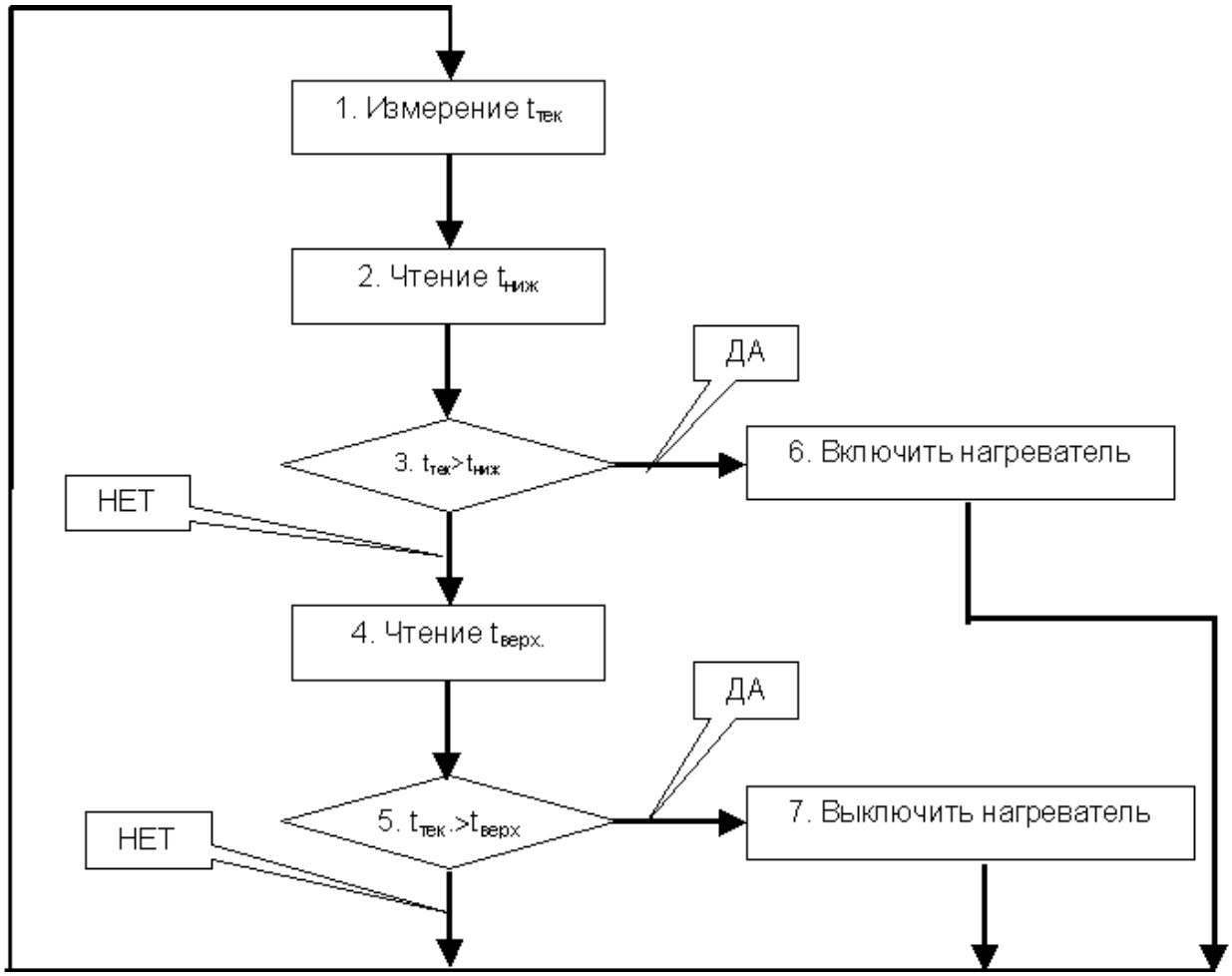

Рисунок 2.1 Концептуальная блок - схема алгоритма.

От порядка будет зависеть длина алгоритма и время работы т.к. при неоптимальной последовательности действий появляются дополнительные действия, позволяющие вывести программу из тупика.

В процессе дальнейшей разработки детализируем концептуальную блок-схему до отдельных шагов.

Блок 1 может быть разбит на следующие шаги: настройка порта, к которому подключен аналогово-цифровой преобразователь с датчиком, запуск АЦП, проверка готовности АЦП, чтение содержимого порта (если разрядность выходного кода АЦП больше разрядности шины данных микроконтроллера, то чтение осуществляется по частям). Детализированная схема рассмотренного фрагмента приведена на рисунке 2.2.

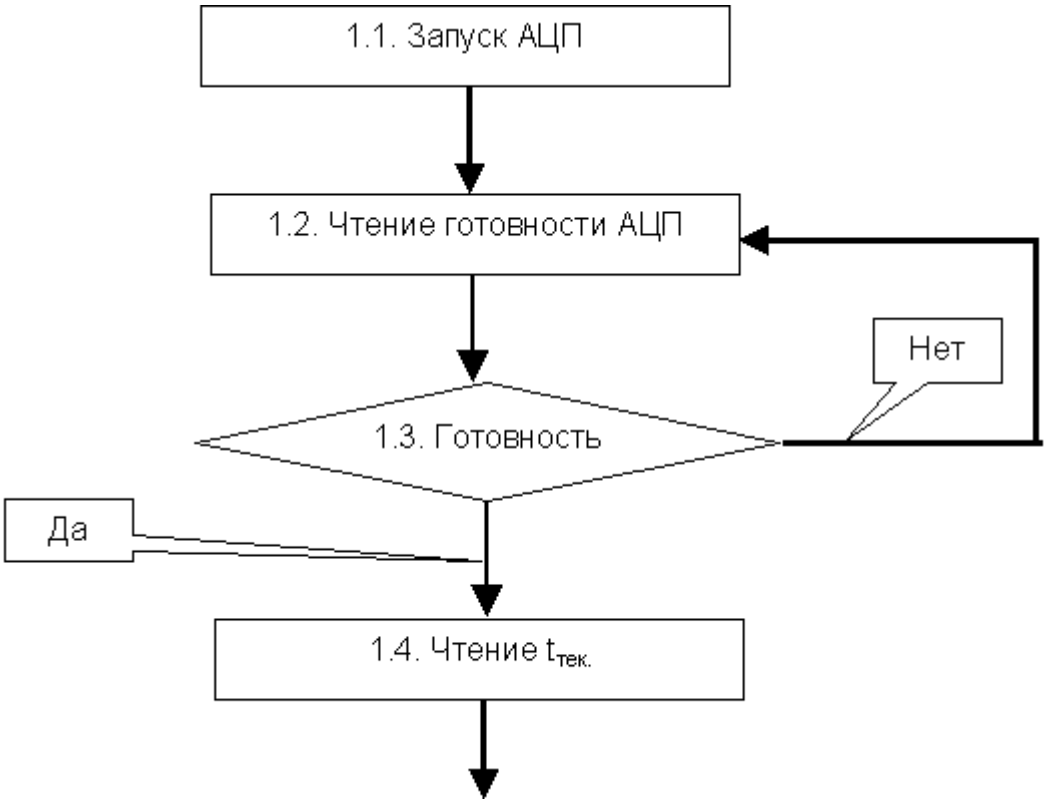

Рисунок 2.2 Детализация концептуальной блок-схемы.

Аналогово-цифровой преобразователь работает в режиме внешнего запуска. После запуска АЦП (1.1), программа ждёт конца преобразования постоянно опрашивая выход готовности (1.2, 1.3). При получении сигнала готовности производится чтение текущего отсчёта АЦП, являющегося результатом измерения $t_{тек.}$. Из приведенной схе-

мы видно, что при детализации концептуальной блок – схемы алгоритма на этом этапе команды МП не используются. Полная функциональная схема алгоритма приведена на рисунке 2.3.

Детализация алгоритма до функционального уровня сводится к поблочному раскрытию концептуальной блок схемы до элементарных шагов. Например, для чтения текущего значения температуры необходимо запустить АЦП (1.1), дождаться конца преобразования (1.2, 1.3) и только после завершения процесса преобразования можно считать выходной код (1.4). Аналогично поступаем со всеми блоками. Для чтения констант необходимо указать их адреса (2.1,2.2, 4.1,4.2), для сравнения значений текущей температуры с заданными необходимо выполнить команду сравнения(3.1,5.1) и проанализировать результат её выполнения (3.2,5.2), после чего принять решение о необходимости перехода.

Для управления нагревателем необходимо сформировать управляющее слово (6.1, 7.1), которое будучи выданным на внешний порт включит (6.2) или выключит (7.2) нагреватель. Как видно из рисунка 2.3 на этом этапе команды конкретного процессора не используются.

Следующим этапом разработки программного обеспечения является написание структурной схемы машинных команд, который начинается с выбора процессора, под который будет писаться программа. Для примера возьмём процессор Кр580ВМ80, который явился прародителем огромный серии различных МП и однокристальных микроЭВМ, сохраняя со многими полную или частичную программную совместимость. Производительности этого МП для нашего примера достаточно. Система команд МП Кр580ВМ80 приведена в приложение.

Разработку структурной схемы машинных команд начнём с того, что определимся, как будет МП обращаться МП к внешним портам. Возможны два варианта: как к ячейке памяти, как к устройству ввода вывода. В первом случае используются команды пересылки, порты будут находится в пределах доступного адресного пространства. Во втором используются специальные команды ввода-вывода IN и OUT. Каждый способ имеет свои преимущества и недостатки. С точки зрения понятия процесса разработки алгоритма используем второй способ. При выполнение операций сравнения будем использовать косвенную адресацию, в этом случае программа будет более универсальной, так как имеется возможность изменять рабочий диапазон температур.

Если диапазон строго задан и непредусматриваеся его изменение, то с точки зрения быстродействия и уменьшения объёма программы целесообразнее применять команды с непосредственной адресацией.

Введём условные обозначения, которые будем использовать вместо констант.

АдМах – адрес ячейки памяти в которой хранится верхнее значение рабочего диапазона ($t_{верх.}$).

АдMin - адрес ячейки памяти в которой хранится нижнее значение рабочего диапазона ($t_{ниж.}$).

Пуск – слово запуска АЦП.

М – маска для выделения сигнала готовности АЦП.

Вкл – управляющее слово, которое при выдаче его в порт включает нагреватель.

Выкл - управляющее слово, которое при выдаче его в порт выключает нагреватель.

ПортВ – имя порта к которому подключается АЦП.

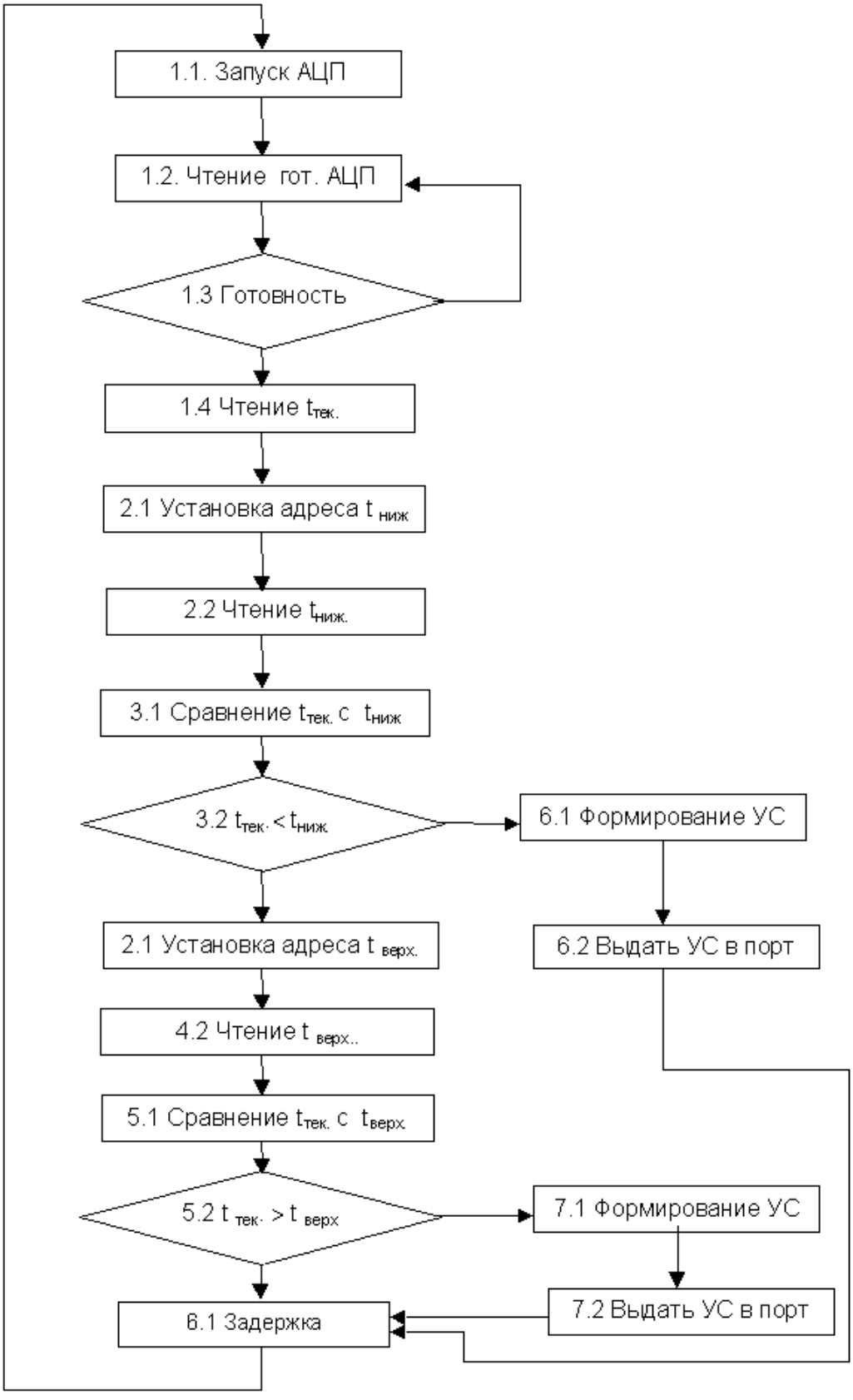

Рисунок 2.3. Функциональная блок – схема алгоритма

ПортГ – имя порта ввода – вывода к которому подключены управлющие входы АЦП и выход готовность.

ПортН – порт управляющий нагревателем.

Time – подпрограмма временной задержки, определяющая период опроса датчика температуры.

С учетом принятых обозначений структурная схема машинных команд примет вид приведенный на рисунке 2.4. Как видно из рисунка по объёму структурная схема машинных команд незначительно превосходит функциональную блок схему, однако реализована она с использованием команд конкретного процессора. Задачей следующего этапа будет присвоение именам конкретных значений, в соответствии с разбивкой адресного пространства МП под реальное устройство и разработка программы на ассемблере МП. Прежде чем приступить к написанию программы необходимо помнить, что полученные алгоритмы представляют собой лишь черновой вариант алгоритма и реальном случае может потребоваться дополнительная оптимизация по быстродействию или объёму необходимой памяти. Если быстродействие разработанного алгоритма не позволяет работать системе в реальном масштабе времени, то может потребоваться внесение коррективов в аппаратную часть МПУ, что в свою очередь повлечет изменения алгоритма.

Другими словами процесс разработки программного обеспечения проводится одновременно с разработкой аппаратной части и носит итеративный характер. Количество итераций зависит от степени подготовки разработчика и его интуиции.

Следующим этапом разработки программного обеспечения является написание рабочей программы на языке ассемблера МП Кр580ВМ80.

2.3. Разработка программы на ассемблере

Разработка программы производится с применением, каких либо средств поддержки или вручную [6]. В настоящее время существует большое количество программных и аппаратно программных средств позволяющих написать и отладить программы для любых процессоров и ОМЭВМ. Чаще всего используются кросс-ассемблеры, эмуляторы МП и ОМЭВМ реализованные на компьютерах IBM PC. Исходные программы пишутся в текстовых редакторах встроенных в средства отладки либо автономных.

В последнем случае нужно иметь в виду, что не все редакторы пригодны для написания программы, а только те которые не содержат невидимые на экране спецкоды форматирования.

Широко распространенные редакторы WORD всех модификаций для этой цели не пригодны. Можно использовать «Блокнот», Лексикон, любые редакторы, работающие под DOS. Программа на Ассемблере представляет собой форму записи машинных кодов в более удобной символической форме. Мнемонические обозначения команд, используемые в ассемблере индивидуальны для каждого микропроцессора. Кроме того, необходимо знать, как работает программируемый процессор и его программную модель, отражающую программные ресурсы доступные программисту. Сведения о некоторых МП и микро-ЭВМ приведены в приложении. Таким образом, стандартизированным в Ассемблере является лишь форма записи командной строки, а её содержание определяется типом применяемого микропроцессора, его архитектурой, системой команд, особенностями функционирования.

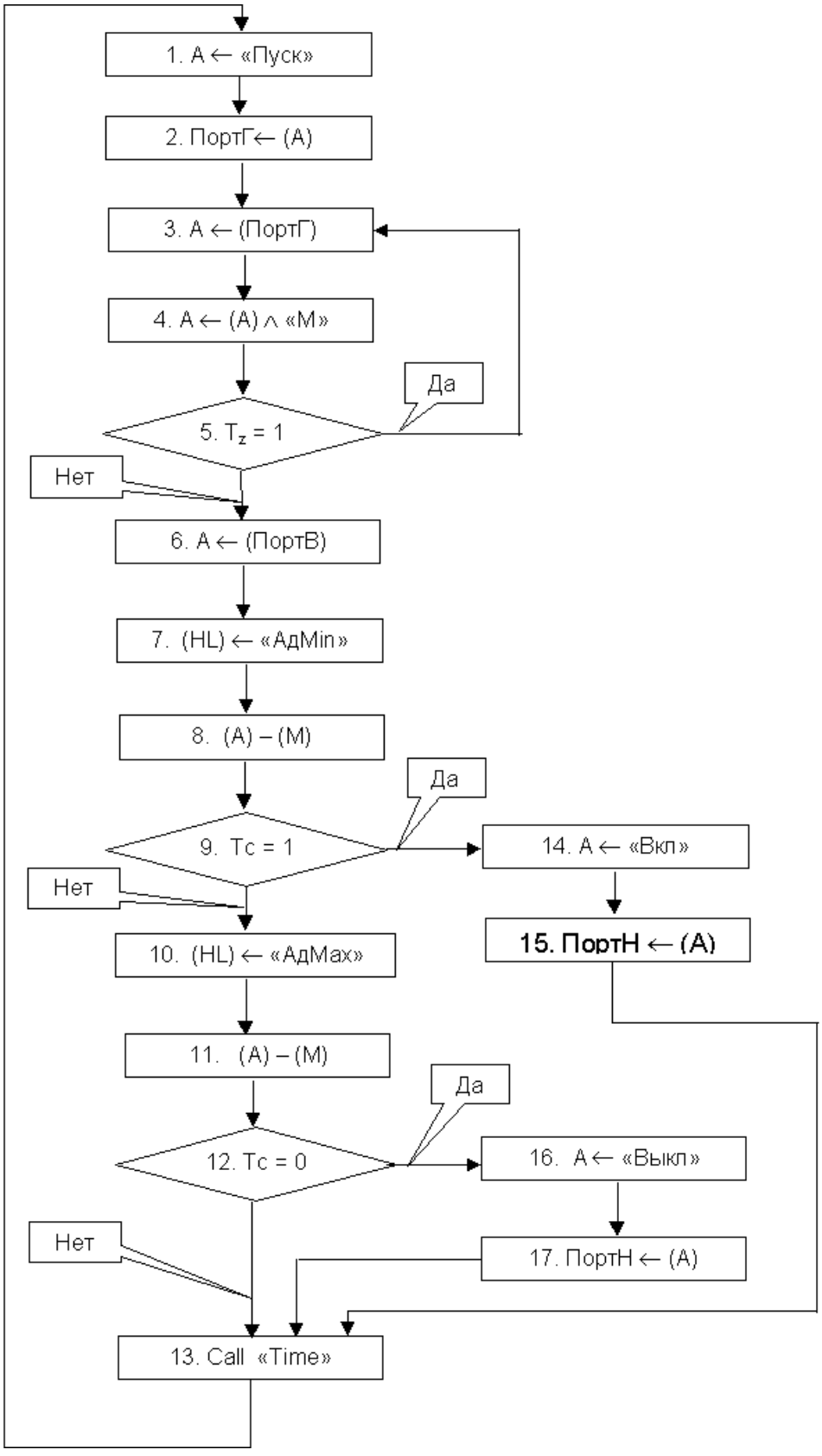

Рисунок 2.4 Структурная схема машинных команд.

Программа на ассемблере состоит из четырёх полей: поля метки, мнемоники, операндов, комментария [6,7].

Поле метки. Необязательное символическое наименование в поле метки ассоциируется с 16-битным адресом той ячейки памяти, в которую будет размещен первый байт отмеченной команды. Метки используются в качестве адресов перехода команд передачи управления и освобождают программиста от необходимости оперировать абсолютными адресами памяти.

Метка имеет длину от одного до пяти или шести символов, первым из которых обязательно должна быть буква. Метка не должна содержать знаков пунктуации и пробелов. За последним символом метки следует двоеточие. В качестве меток не допускается использовать ключевые слова.

Поскольку метки ассоциируются с адресами памяти, их нельзя дублировать. Другими словами, каждое символическое наименование может появиться в поле метки только один раз. В противном случае ассемблер выдает диагностическое сообщение о многократно определенной метке.

Поле мнемоники (кода). В поле мнемоники содержится символическое описание выполняемой команды, которое заменяет числовое значение кода операции. Большинство мнемоник представляют собой аббревиатуры предложений, характеризующих основные функции команд, например:

MOV (MOVe) — передать, переслать;

ACI (Add with Carry Immediate) —сложение с переносом непосредственное;

JNZ (Jump if Non Zero) —перейти, если не нуль;

XCHG (eXCHanGe)—обменять.

Некоторые мнемоники явно определяют функцию команды, например команда PCHL означает передачу содержимого регистров (H, L) в программный счетчик PC, а команда PUSH выполняет загрузку в стек.

Обычно длина поля мнемоники не превышает четырех позиций, а между ним и соседним справа полем операнда необходим минимум один пробел. Мнемоники кодов операций являются ключевыми словами ассемблера и, если содержимое этого поля не входит в множество допустимых мнемоник, ассемблер выдает сообщение о недействительной команде. Мнемоники некоторых типов МП и микро-ЭВМ приведены в приложении.

Поле операнда. В поле операнда определяются данные, являющиеся операндом (операндами) команды. Следовательно, содержимое поля операнда должно интерпретироваться в соответствии с функцией команды. В качестве операндов могут фигурировать адреса памяти, внутренние регистры микропроцессора, адреса портов ввода и вывода, числовые и символьные константы. Некоторые команды, оперирующие определенными внутренними регистрами, имеют пустое поле операнда.

Например, поле операнда стандартного языка Ассемблера микропроцессора К580 может содержать информацию следующих типов: числовые и символьные непосредственные данные, внутренние регистры и регистровые пары, адреса памяти. Рассмотрим способы определения информации указанных типов.

Шестнадцатеричные данные. Содержащееся в поле операнда шестнадцатеричное число должно начинаться с цифр 0 - 9 и завершаться буквой H (Hex). Число, начинающееся с букв A - F, дополняется слева незначащим нулем.

Десятичные данные. Десятичное число в поле операнда заканчивается необязательно буквой D (Decimal).

Восьмеричные данные. Каждое восьмеричное число должно заканчиваться буквой O (Octal). Чаще восьмеричную систему идентифицируют буквой Q, чтобы отличить букву O от цифры 0.

Двоичные данные. Двоичное число в поле операнда должно заканчиваться буквой B (Binary).

Символьные константы. В поле операнда допускается использовать символы внешнего алфавита, заключая их в апострофы. Программа-ассемблер подставляет вместо такого операнда соответствующий двоичный код символа.

Идентификаторы внутренних регистров. В поле операнда разрешается указывать символические наименования, которые определены в самом ассемблере и связаны с внутренней архитектурой микропроцессора.

Метки. В командах передачи управления можно указывать метки, введенные программистом в поле метки других команд. Метки в поле операнда заменяют абсолютные значения адресов перехода. Любой символический адрес, фигурирующий в поле операнда команд передачи управления, должен один раз появиться в поле метки некоторого оператора. Если он не появляется ни разу, программа-ассемблер выдает сообщение о неопределенной метке.

Текущее содержимое программного счетчика. В командах передачи управления допускается относительная адресация, т. е. адрес перехода определяется суммой (разностью) текущего содержимого программного счетчика.

Выражения. Наиболее сложными конструкциями поля операнда являются выражения. Они содержат в качестве аргументов все рас-

смотренные выше типы данных, которые связываются арифметическими и логическими операторами.

Поле комментария. Поле комментария, начинающееся с некоторого разделителя (точка с запятой или наклонная черта), полностью игнорируется программой-ассемблером, поэтому в нем можно помещать любой текст. Содержание этого поля должно пояснять те действия, которые выполняет команда в контексте конкретной прикладной программы. Комментарием может служить целая строка, начинающая с разделителя.

В отношении содержания комментариев следует придерживаться двух рекомендаций. Во-первых, в комментарии акцентируется не общая функция команды, которая определяется мнемоникой, а действие команды в данной программе. Во-вторых, не следует сокращать объем комментария в ущерб его смыслу. Четкие и содержательные комментарии облегчают понимание и использование программ.

Для управления процессом трансляции используют директивы, которые не являются командами процессора и не порождают кода при трансляции, а только являются указаниями транслятору как осуществлять трансляцию. Директивы определяют порядок ассемблирования, размещают в памяти информацию, присваивают численные значения символическим наименованиям, резервируют память и выполняют другие функции. Количество директив зависит от сложности транслятора, стандартизировано по умолчанию лишь небольшое их количество.

Директивы подчиняются стандартному формату операторов ассемблера, но содержимое их полей имеет некоторые особенности, например в поле метки директив MACRO, EQU и SET должно обязательно находиться символическое наименование, которое не имеет

заключительного двоеточия. В остальных директивах в поле метки может быть необязательная метка, аналогичная меткам машинных команд. Метка директивы относится к ячейке памяти, которая следует сразу же за последней ячейкой предшествующей машинной команды.

Директива ORG. Значением выражения директивы ORG является допустимый адрес, определяющий ячейку памяти, в которую будет загружаться первый байт следующей команды или байт данных. До новой директивы ORG команды и данные размещаются в смежных ячейках памяти. Если в самом начале программы директива ORG отсутствует, то по умолчанию подразумевается наличие директивы ORG с нулевым операндом. При программировании микроконтроллеров директива ORG предназначена для адаптации разрабатываемой программы к адресному пространству реального устройства. Это связано с тем, что Ассемблеры, реализованные на мощных машинах, могут размещать программу по своим фиксированным адресам. При необходимости в программе может быть несколько директив ORG. Директива ORG может выполнять функцию резервирования памяти.

Директива END. Эта директива информирует программу-ассемблер о достижении физического конца входной программы. В каждой программе может быть только одна директива END, находящаяся в последней строке. После неё не должно быть никаких команд иначе они останутся не оттранслированы.

Директива EQU. При выполнении директивы EQU программа-ассемблер присваивает значение выражения символическому наименованию, находящемуся в поле метки. Когда наименование встречается в поле операнда, программа-ассемблер подставляет вместо него присвоенное значение. Символическое наименование может поя-

виться в поле метки только одной директивы.

Директива DB (определить байт). Относится к группе директив определения, которые применяются для инициализации данных и резервирования памяти. Операнд директивы DB может быть последовательностью выражений, разделенных запятыми и имеющих 8-битные значения, либо цепочкой символов, заключенных в апострофы. При выполнении директивы DB значения выражений или коды символов запоминаются в смежных ячейках (байтах) памяти, начинающихся после ячейки предыдущей команды. В качестве выражения в поле операндов может быть символьное выражение, заключённое в апострофы.

Директива DW. Директива DW (определить слово — 2 байта) также относится к директивам определения и имеет такой же формат, как и директива DB. Однако здесь списком является последовательность выражений, имеющих 16-битные значения. При выполнении директивы DW вычисляется значение первого выражения и его младшие 8 бит запоминаются по текущему адресу, а старшие 8 бит запоминаются по адресу на единицу больше предыдущего. Затем вычисляется значение второго выражения, процедура запоминания повторяется для следующих ячеек памяти и т. д.

Директива DS (определить память). Вычисленное значение выражения из поля операнда определяет число ячеек (байт) памяти, резервируемых для запоминания данных. Никакие значения в этих ячейках не запоминаются, в частности, нельзя считать, что эти ячейки содержат нули. Адрес следующего оператора равен сумме адреса оператора, находящегося перед директивой DS и значения выражения директивы DS.

При программировании возникают ситуации, когда одно и то же

действие, возможно, с небольшими модификациями, описываемое группой команд, необходимо выполнять в программе несколько раз. Для сокращения длины входной программы и ускорения программирования в ассемблерах такую группу команд допускается определить один раз как большую команду - макрокоманду с уникальной мнемоникой, не входящей в систему команд микропроцессора. После определения макрокоманды в начале программы ее мнемонику можно использовать сколько угодно раз так, как будто она включена в систему команд микропроцессора. Всякий раз программа-ассемблер заменяет эту мнемонику той последовательностью команд, которая фигурирует в определении макрокоманды. Таким образом, введение макрокоманд придает ассемблеру некоторые черты языков высокого уровня. Макрокоманды пишутся следующим образом.

В поле метки находится имя (символическое наименование или мнемоника), введенное программистом, а в поле операнда—список формальных параметров, называемых также аргументами. Параметры разделяются запятыми и в общем случае могут быть выражениями. Группа операторов между директивами MACRO и ENDM, называемая телом макрокоманды, может содержать любые машинные команды, директивы (кроме MACRO и ENDM), комментарии и обращения к другим макрокомандам, если ассемблер допускает вложенные макрокоманды. Параметры должны быть определены при каждом обращении к макрокоманде. Отметим, что определение макрокоманды не порождает команд объектной программы, а указывает программе-ассемблеру, что имя из поля метки является эквивалентом ассемблерных операторов, находящихся между директивами MACRO и ENDM.

Обращение к макрокоманде. В поле мнемоники содержится имя

макрокоманды, которое фигурировало в поле метки директивы MACRO, а в поле операнда—список фактических параметров. Значения параметров подставляются в тело макрокоманды слева направо в соответствии с полем операнда директивы MACRO. Если в обращении имеется меньше параметров, чем в определении макрокоманды, то остальные параметры считаются пустыми. Если в обращении больше параметров, чем в определении, лишние параметры игнорируются.

Расширение макрокоманды. Операторы, образующиеся в результате подстановки фактических параметров в тело макрокоманды, называются расширением макрокоманды. Расширение должно содержать только допустимые операторы языка ассемблера.

Для избежания конфликтов, связанных с многократным определением меток, программа-ассемблер считает метки в макрокомандах локальными, т. е. действующими только в пределах конкретного расширения. Поэтому в макрокомандах допускается применять метки, фигурирующие в других частях программы. При необходимости создания глобальной метки, область действия которой выходит за пределы расширения макрокоманды, необходимо закончить метку двумя двоеточиями вместо одного. Глобальная метка может встречаться в программе только один раз.

Каждое появление имени макрокоманды в поле мнемоники ассемблерных операторов вызывает включение команд тела макрокоманды в объектную программу. В этом заключается основное отличие макрокоманд от подпрограмм, тело которых включается в объектную программу только один раз. Подпрограммы сокращают длину программы, но увеличивают время выполнения из-за непроизводительных потерь на вызов и возврат. При использовании макрокоманд

длина объектной программы увеличивается, а время выполнения уменьшается.

Результатом работы транслятора является объектный код который загружается в память контроллера с адреса определяемого директивой ORG. Если разрабатываемая программа должна включать в себя какие либо дополнительные ранее разработанные модули, то для окончательной компоновки программы необходимо провести коррекцию адресов переходов. Для этой цели используются специальные служебные программы - редакторы связей, обычно входящие в состав ассемблеров.

Приведём полный текст программы реализующей рассмотренные выше алгоритмы.

```
;  Программа функционирования нагревателя
;          на ассемблере МП Кр580ВМ80
           ORG 0000H
Пуск:   EQU _____     ; численные значения присваиваемые
ПортГ:  EQU _____     ; символическим именам определяются
ПортВ:  EQU _____     ; в процессе разработки аппаратной
АдMin:  EQU _____     ; части и при распределении адресного
                      ; пространства проектируемого
АдMax:  EQU _____     ; МП устройства
ПортН:  EQU _____     ;
Вкл:    EQU _____     ;
Выкл:   EQU _____     ;
Нач:    MVI  A,Пуск   ; загрузка слова запуска и запуск
        OUT  ПортГ    ; аналого-цифрового преобразователя
Гот:    IN   Порт ; опрос выхода готовность АЦП и
        ANI  M        ; ожидание конца преобразования
```

```
        JZ    Гот       ;
        IN    ПортВ     ; считывание текущего значения tтек.
        LXI   H,АдМin   ; загрузка адреса tmin.
        CPR   M         ; сравнение tтек. с  tmin.
        JC    Вk        ;
        LXI   H,АдМax   ; загрузка адреса tmax.
        CPR   M         ; сравнение tтек. с  tmmax.
        JNC   Вык       ;
пауза:  CALL  Time      ; вызов подпрограммы задержки
        JMP   Нач       ;
Вк:     MVI   A,Вкл     ; загрузка слова включения нагревателя
        OUT   ПортН     ; и включение нагревателя
        JMP   пауза             ;
Вык:    MVI   A,Выкл    ; загрузка слова выключения нагревателя
        OUT   ПортН     ; и выключение нагревателя
        JMP   пауза             ;
        END
```

В поле операндов директив EQU вместо прочерков размещаются численные значения констант, адресов и номеров портов ввода вывода, в соответствии с функциональной схемой устройства.

3. МИКРОПРОЦЕССОРЫ. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

3.1. Микропроцессор Кр580ВМ80

Логическая организация МП Кр580ВМ80. МП с фиксированной разрядностью шин и строго определенной системой, команд представляют собой широкий класс МП, архитектура и логическая организация которых во многом повторяет организацию ЭВМ малой и средней производительности. Разрядность шины данных, состав и формат команд не могут меняться произвольно, программы составляются из небольшого числа команд, что тоже является препятствием при разработке эффективных программ и снижает быстродействие МП системы. Это обуславливается тем, что, отсутствие нужной команды для решения конкретной задачи в списке команд МП приводит к необходимости реализации требуемой функции программным путем. Если, разрядность шины данных меньше разрядности обрабатываемых слов, то обработку приходятся вести по частям, что увеличивает время работы программы.

К достоинствам МП данного типа следует отнести низкую стоимость и простоту программирования. Это связано с тем, что разработчик использует уже готовую систему команд. Несмотря на некоторые серьезные недостатки область применения подобных МП велика. С применением этих МП строятся управляющие и универсальные микро-ЭВМ, микроконтроллеры для управления измерительными приборами и технологическими процессами. Типичным представителем этого класса является широко распространенный микропроцессор Кр580ВМ80 и его более совершенный вариант Кр1821ВМ85. Структурная схема микропроцессора приведена на рисунке 3.1.

МП представляет собой однокристальный 8-разрядный микропроцессор размещенный в 40-выводном корпусе [10,11].

МП включает в себя три основные части: АЛУ, банк регистров, схему управления. Взаимосвязь между различными блоками осуще-

ствляется с помощью внутренней 8-разрядной шины данных. МП имеет внешнюю 8-и разрядную, двунаправленную шину данных с тремя логическими состояниями и 16-разрядную однонаправленную шину адреса с тремя логическими состояниями. Такая 16-разрядная шина адреса позволяет адресовать 2 = 65536 ячеек памяти.

АЛУ МП представляет собой 8-разрядное комбинационное логическое устройство предназначенное для выполнения арифметических и логических операций. Один из операндов поступающих на АЛУ всегда размещается в аккумуляторе, в аккумулятор помещается и результат операции. АЛУ данного МП выполняет только простейшие операции (сложение, вычитание, сдвиг и.т.д.), более сложные операции реализуются программно.

Все регистры МП Кр580ВМ80 можно разделить по назначению на две группы: регистры общего назначения (РОН) и специальные регистры.

Регистры общего назначения. МП имеет 8 регистров общего назначения. Из них регистры W и Z программно недоступны и предназначены для внутренних пересылок информации в МП, в частности для хранения двух и трехбайтовых команд. Регистры B,C,D,E,H,L программно доступны и предназначены для хранения операндов и промежуточных результатов вычислений. Использование этих регистров в программе позволяет сократить время выполнения программы и её объём.

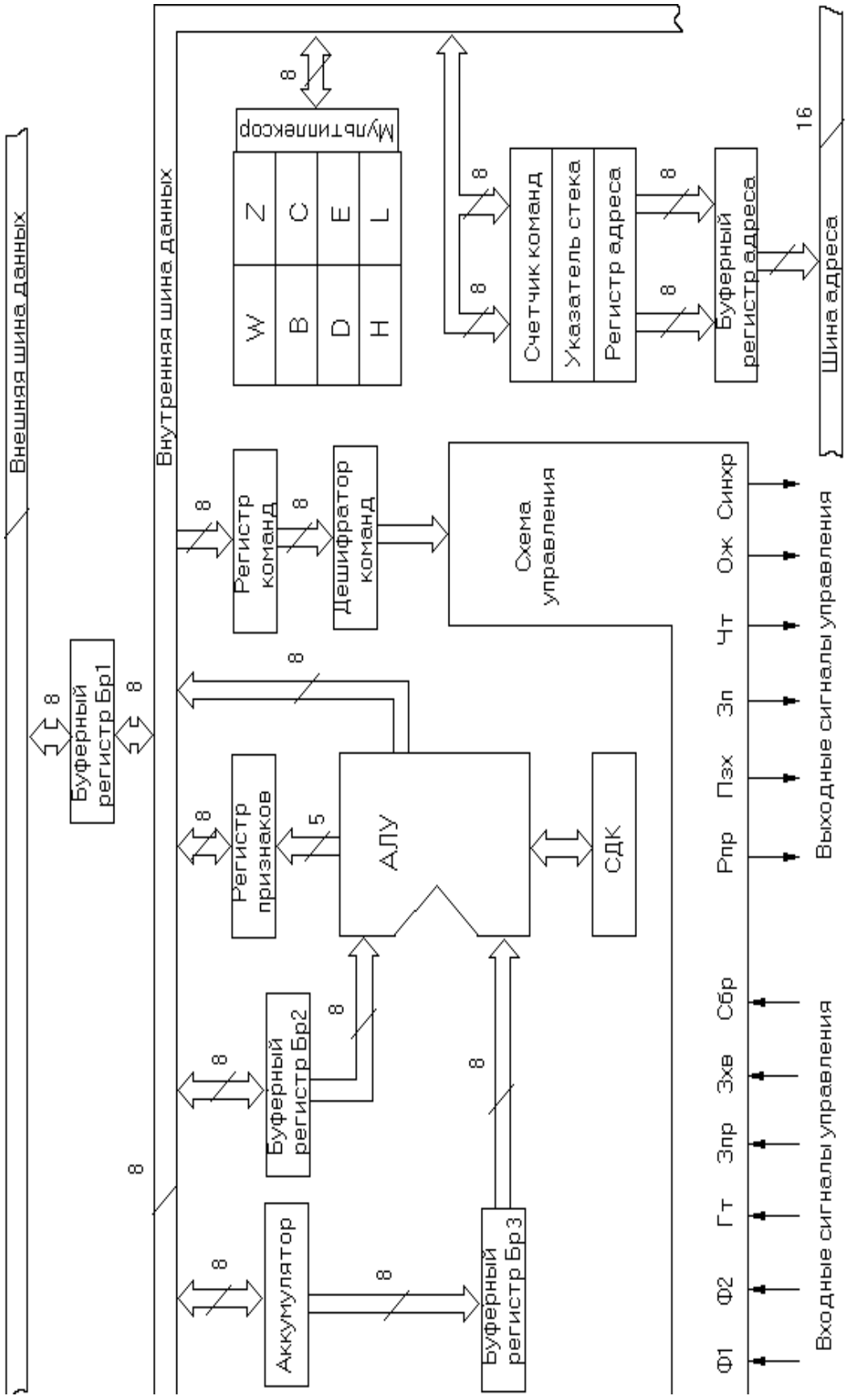

Рисунок 3.1 Структурная схема микропроцессора Кр580ВМ80.

Особенность этих регистров состоит в том, что, они могут использоваться как отдельно (как 8-разрядные регистры), так и в парах HL, BC, DE, при этом. они используются для хранения 16-разрядных операндов и адресов. Пара регистров HL может использоваться для хранения адреса ячейки памяти (M) при обращении к ней как к регистру.

Специальные регистры. К специальным регистрам относятся: счетчик команд, указатель стека, регистр адреса, аккумулятор, регистр признаков.

Счетчик команд Pc представляет собой 16-ти разрядный регистр, содержащий адрес выполняемой команды, при работе МП его значение автоматически увеличиваться при выборке очередной команды на 1,2, или 3 единицы в зависимости от того, сколько байт занимает выбираемая команда. Занесение какого-либо числа N в этот регистр означает, что следующая команда будет выбираться из ячейки памяти с номером N.

Указатель стека Sp – 16-разрядный регистр, предназначенный для хранения адреса вершины стека. При записи информации в стек содержимое указателя стека уменьшается на 2, при чтении увеличивается на 2. Изменение содержимого стека на 2 связано с тем, что загрузка и чтение стека осуществляется по две ячейки.

Буферный регистр адреса предназначен для хранения адреса ячейки памяти, к которой производится обращение. В момент выборки очередной команды из ЗУ содержимое счетчика команд и буферного регистра адреса, одинаково.

Регистр команд предназначен для хранения кода команды на время выполнения команды и имеет 8-разрядную структуру, программно недоступен.

Регистр состояния (регистр признаков) предназначен для хране-

ния информации о результате операции. В нем хранится информация о пяти признаках:

Tc – признак переполнения разрядной сетки,

Tz – признак нулевого результата,

Ts – признак наличия 1 в старшем разряде,

Tv – признак переноса из младшей тетрады в старшую,

Tp – признак четности числа единиц в результате.

Информация хранящаяся в регистре признаков используется для организации ветвящихся и циклических программ, для работы с ними в системе команд имеются команды передачи управления.

Схема десятичной коррекции предназначена для автоматической коррекции результата операции сложения при представлении операндов, в двоично-десятичном коде.

Форматы и система команд. Система команд МП Кр580ВМ80 состоит из 78 команд, которые могут образовывать 244 кодовых комбинаций. В соответствии с функциональным назначением их можно разделить на несколько групп. Полный состав системы команд приведен в приложении 1. В командах используются три вида адресации, и они различаются количеством ячеек памяти требуемых для их размещения.

Команды МП могут быть длиной в одно, два или три машинных слова, т.е. могут быть: однобайтовыми, двухбайтовыми, трехбайтовыми.

Байты команд в программах размещаются в соседних ячейках памяти и последовательно выбираются МП из ОЗУ. Метод указания месторасположения операндов называется адресацией. МП могут иметь несколько видов адресации.

Рисунок 3.2 Форматы команд МП Кр580ВМ80.

Рассматриваемый МП имеет три основных вида адресации: прямую, непосредственную и косвенную. Существует несколько разновидностей указанных методов адресации, но все их можно свести к трем приведенным выше.

Прямая адресация. При прямой адресации каждая команда состоит из кода операции и адреса операнда. Обычно такие команды состоят из трех байтов, при этом второй и третий байты содержат адрес операнда.

Непосредственная адресация. При этом виде адресации подлежащие обработке данные содержатся непосредственно в коде самой команды, во втором и третьем байтах команды.

Косвенная адресация позволяет обращаться ко всем ячейкам памяти как к регистрам общего назначения. В этом случае в поле операнда определяется не регистр с данными, а указатель памяти, содержащий адрес операнда. Для быстрого выполнения операции в качестве указателя используется внутренний регистр, а так же пара регистров. В МП Кр580ВМ80 основным указателем является пара регистров HL.

Адресное пространство МП составляет 64к. Для указания номера устройств ввода вывода используется 8 разрядов шины адреса, причём он дублируется в младшем и старшем байте адреса. Таким образом, МП доступно 256 устройств ввода и 256 устройств вывода.

3.2. Однокристальные микро-ЭВМ семейства 048

Однокристальная 8-разрядная микро-ЭВМ (ОМЭВМ) семейства МК048 представляет собой БИС, имеющую в своем составе арифметическо-логическое устройство, устройство управления, ПЗУ программ, ОЗУ данных и интерфейсные схемы. Набор команд и ограниченные возможности ввода - вывода информации определяют основную форму его использования - в качестве специализированного вычислителя, включаемого в контур управления объектом или процессом [11-15].

В состав ОМЭВМ входят: стираемое перепрограммируемое ПЗУ программ емкостью 1 Кбайт (только для МК048); регистровое ОЗУ данных емкостью 64 байта; 8-разрядное арифметическо-логическое устройство; устройство управления; 8-битный программируемый таймер/счетчик событий; программно-управляемые схемы ввода — вывода (27 линий). Выпускаются модернизации этого семейства, в которых внутреннее ПЗУ может отсутствовать (035), или однократно программируемое ПЗУ (039). Системы команд и архитектура всего семейства идентичны.

Организация МК и его система команд допускают в случае необходимости расширение функционально-логических возможностей контроллера. С использованием внешних дополнительных БИС адресное пространство ППЗУ программ может быть расширено до 4

Кбайт. Архитектура МК обеспечивает возможность прямой адресации внешнего ОЗУ емкостью 256 байт. С использованием более сложных программно-реализуемых способов адресации емкость внешнего ОЗУ может быть увеличена до требуемого объема страницами по 256 байт в каждой. И, наконец, путем подключения интерфейсных БИС КР580ВВ55 число линий связи МК с объектом управления может быть увеличено практически без ограничений.

В МК048 реализована система векторного прерывания от двух источников: внутреннего таймера/счетчика событий и внешнего источника. Внутренний 8-уровневый стек обеспечивает автоматическое сохранение и восстановление основных параметров вычислительного процесса при запросах прерывания и при возврате после обслуживания прерывания.

Три 8-битных порта ввода—вывода информации, два входа тестирующих сигналов и один вход запроса прерывания обеспечивают связь МП с объектом управления по 27 линиям. Тактовая частота ОМЭВМ определяется внешним тактовым генератором и может лежать в пределах от1МГц до 11МГц.

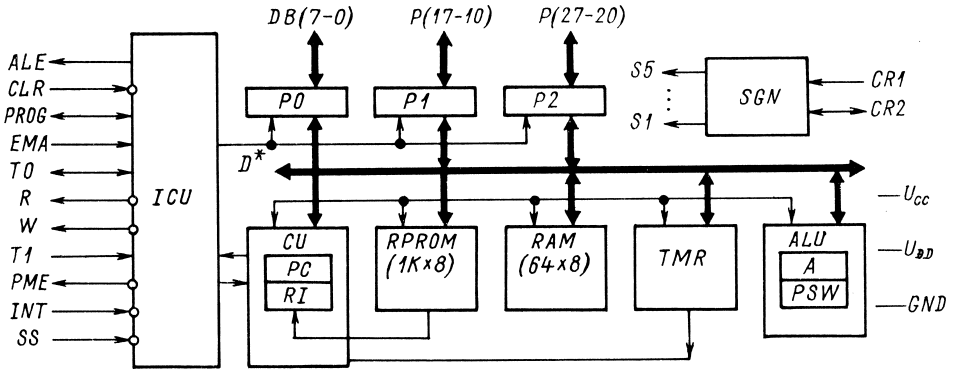

Рисунок 3.3 Структурная схема ОМЭВМ семейства 048.

Основу структуры МК образует внутренняя двунаправленная раз-

деляемая 8-битная шина [14-15], которая связывает между собой все устройства БИС: арифметическо-логическое устройство, устройство управления, память и порты ввода—вывода информации. В состав арифметическо-логического устройства входят следующие блоки: комбинационная схема обработки байт, регистр временного хранения операнда T, регистр-аккумулятор A, схема десятичного корректора и схема формирования признаков, часть из которых фиксируется в регистре слова состояния программы ССП и используется логической схемой управления переходами по программе. Аккумулятор является двухтактным регистром, так как он используется в качестве регистра операнда и регистра результата Динамический регистр T программно-недоступен — служит для временного хранения второго операнда при выполнении двухоперандных команд.

При выполнении операций обработки данных в АЛУ формируются признаки, часть которых не фиксируется на триггерах, а формируется на комбинационной схеме. К таким признакам относятся признак нулевого содержимого аккумулятора, и признаки наличия единицы в селектируемом разряде аккумулятора Логика условных переходов по указанным признакам позволяет без фиксации на триггерах выполнять команды передачи управления (JZ, JNZ, JB0 — JB7).

Признаки переноса (переполнения) и вспомогательного переноса (перенос из младшей тетрады в старшую) фиксируются на триггерах, входящих в состав регистра слова состояния процессора ССП. Формат ССП показан на рисунке 3.4. Функциональное назначение признаков F0 и F1 определяется разработчиком, признаком переполнения таймера FT, признаками наличия сигналов на входах T0 и T1. Программистом могут быть также использованы признаки рабочего банка регистров BS и выбранного блока внешней памяти программ MB.

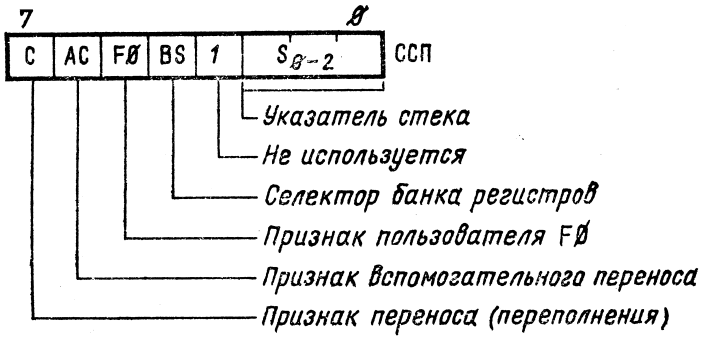

Рисунок 3.4 Формат слова-состояния ОМЭВМ МК048.

Кроме того, после окончания каждого машинного цикла опрашивается еще один признак, а именно признак разрешения/запрета прерываний EI. Полностью система команд семейства 048 приведена в приложении 2.

Память программ и память данных в МП физически и логически разделены. Память программ реализована в резидентном СППЗУ емкостью 1 Кбайт. Максимальное адресное пространство, отводимое для программ, составляет 4 Кбайт. Счетчик команд содержит 12 разрядов, но инкрементируются в процессе счета только младшие 11 разрядов. Поэтому счетчик команд из предельного состояния 7FF (если только по этому адресу не расположена команда передачи управления) перейдет в состояние 000. Состояние старшего разряда счетчика команд может быть изменено специальными командами (SEL MB0, SEL MB1). Подобный режим работы СК позволяет создать два блока памяти емкостью по 2Кбайт каждый. Карта адресов памяти программ показана на рисунке 3.5. В резидентной памяти программ имеются три специализированных адреса адрес 0, к которому передается управление сразу после окончания сигнала системного сброса СБРОС; по этому адресу должна находиться команда безусловного перехода к началу программы; адрес 3, по которому расположен вектор прерывания от внешнего источника; адрес 7, по которому распо-

ложен вектор прерывания от таймера или начальная команда подпрограммы обслуживания прерывания по признаку переполнения таймера/ счетчика событий.

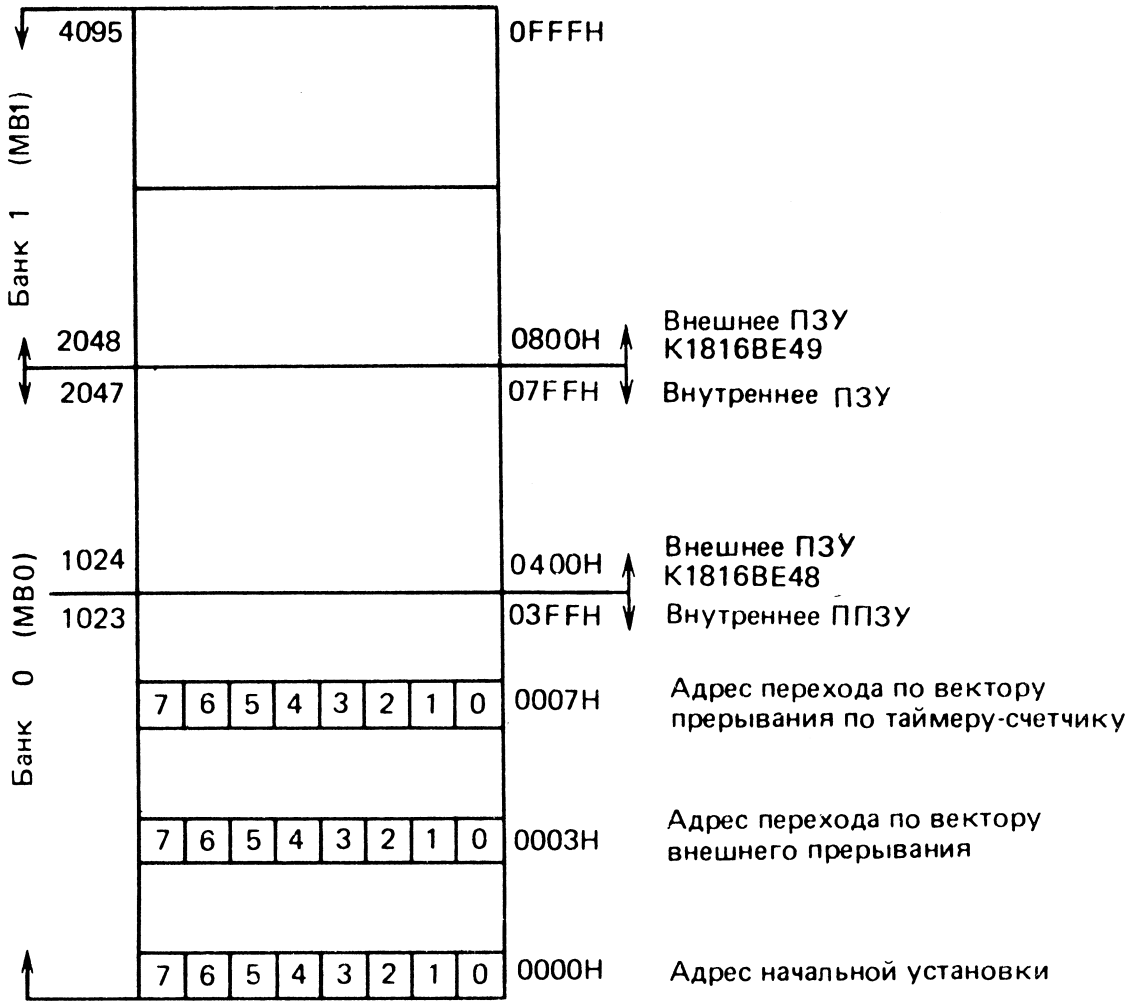

Рисунок 3.5 Распределение адресного пространства МК048.

Память программ разделяется не только на блоки объемом 2 Кбайт, но и на страницы по 256 байт в каждой. В командах условного перехода задается 8-битный адрес передачи управления в пределах текущей страницы. В случае, когда в программе необходимо иметь много переходов по условию, то, если учитывать небольшой размер страницы, возникает проблема размещения подпрограмм в границах страницы и на нескольких страницах. Команда вызова подпрограмм

модифицирует 11 разрядов счетчика команд, обеспечивая тем самым межстраничные переходы в пределах выбранного блока памяти программ.

ПЗУ может использоваться не только для хранения команд, но и констант. Для этого в МК048 применяются два способа адресации постоянной памяти для чтения неизменяемых данных: непосредственная адресация, при которой второй байт двухбайтной команды представляет собой операнд; косвенная адресация, при которой содержимое аккумулятора используется в качестве указателя данных в текущей странице или в странице 3 памяти программ.

При работе с внешней памятью программ, возникает проблема размещения подпрограмм в двух блоках памяти. Так как межблочные переходы выполняются только по командам SEL MB0 и SEL MB1, то необходимо следить за тем, чтобы подпрограммы, взаимно вызывающие друг друга, располагались в пределах одного блока памяти. В противном случае возникает необходимость модификации признака MB в вызываемой подпрограмме и восстановление его при возврате в вызывающую подпрограмму. Но если вызов такой подпрограммы носит условный характер, то проблема восстановления может оказаться трудноразрешимой.

При обработке запросов прерываний в МК 1816 старший разряд $СК_{11}$ принудительно устанавливается в 0. Это приводит к необходимости подпрограмму обслуживания прерывания и все подпрограммы, вызываемые ею, размещать в пределах блока памяти 0.

Память данных. Резидентная память данных емкостью 64 байт имеет в своем составе два банка рабочих регистров 0—7 и 24—31 по восемь регистров в каждом. Выбор одного из двух банков регистров выполняется по команде SEL RB. Рабочие регистры доступны коман-

дам с прямой адресацией, а все ячейки ОЗУ доступны командам с косвенной адресацией. В качестве регистров косвенного адреса используются регистры R0, R1 и R0*, R1*.

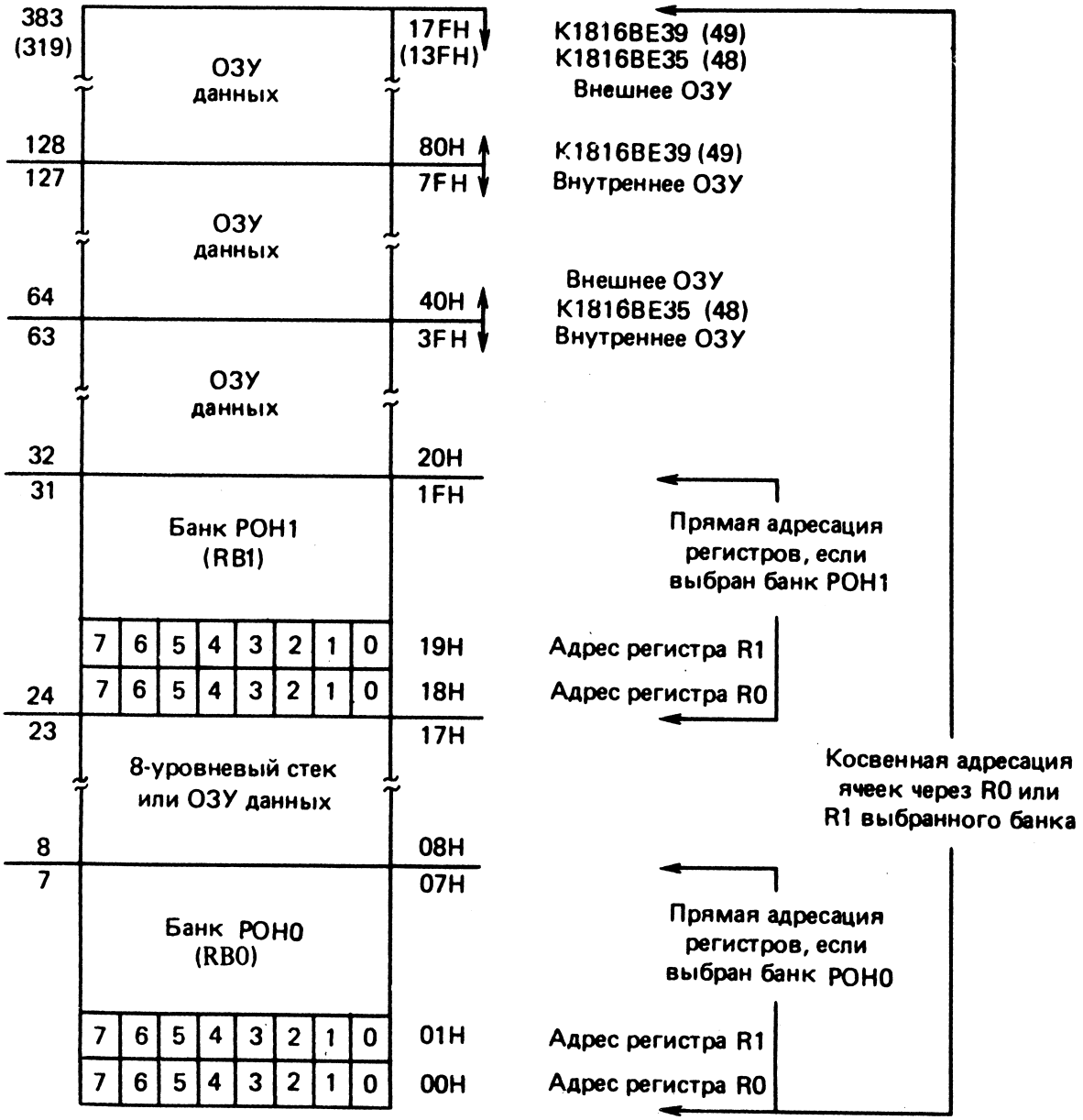

Рисунок 3.6 Распределение памяти данных МК048.

Ячейки ОЗУ с адресами 8 - 23 адресуются указателем стека из ССП и могут быть использованы в качестве 8-уровневого стека. В том случае, если уровень вложенности подпрограмм меньше 8, то незадействованные в стеке регистры могут использоваться как ячейки

ОЗУ. При переполнении стека регистр-указатель стека, построенный на основе 3-разрядного счетчика, переходит из состояния 7 в состояние 0. Программисту необходимо следить за тем, чтобы вложенные подпрограммы не использовали одноименные ячейки ОЗУ в качестве рабочих регистров. Если же такая необходимость возникает, то загрузку в стек и извлечение из него можно выполнить программно, путем передачи ССП, в аккумулятор, выделения по маске указателя стека, передачи его в регистр косвенного адреса R0 или R1 с последующим обращением к ОЗУ по сформированному таким образом адресу вершины стека.

Практически все команды с обращением к ОЗУ оперируют с одним байтом. Однако по командам вызова и возврата осуществляется доступ к двухбайтным словам. В памяти данных слова хранятся так, что старший байт слова располагается в ячейке с большим адресом. Отметим, что в памяти программ порядок расположения байт по старшинству при хранении двухбайтных слов обратный.

Если используется внешнее ОЗУ, через регистры косвенного адреса R0 и R1 возможен доступ к внешней памяти объемом 256 байт. В случае необходимости в МП-системах можно применить внешнее ОЗУ требуемого объема, если, например, использовать 4 бита порта ввода—вывода информации для выбора одной из 16 страниц внешней памяти, каждая из которых имеет объем 256 байт. При этом 4-битный указатель страниц становится дополнением ССП и должен сохраняться в стеке при обработке прерываний.

Для связи с объектом управления, для ввода и вывода информации используются 27 линий. Эти линии сгруппированы в 3 порта по 8 линий в каждом и могут быть использованы для вывода, ввода или для ввода—вывода через двунаправленные линии. Кроме портов

ввода - вывода, имеются 3 входные линии, сигналы на которых могут изменять ход программы по командам условного перехода: линия ЗПР используется для ввода в МК сигнала запроса прерывания от внешнего источника; линия Т0 предназначена для ввода тестирующего сигнала от двоичного датчика объекта управления; кроме того, под управлением программы (ENTO CLC) по этой линии из МК может выдаваться сигнал синхронизации; линия Т1 используется для ввода тестирующего сигнала или в качестве входа счетчика событий (по команде STRT CNT).

Порты ввода—вывода Р1 и Р2. Каждая линия портов Р1 и Р2 может быть программным путем настроена на ввод, вывод или на работу с двунаправленной линией передачи. Квазидвунаправленные порты Р1 и Р2, позволяют выполнять ввод, вывод и двунаправленные передачи.

Для того, чтобы настроить линию на режим ввода в МП, необходимо перед этим в буферный D-триггер этой линии записать 1. Сигнал системного сброса СБРОС автоматически записывает во все линии портов Р1 и Р2 сигнал 1. При программировании портов Р1 и Р2 следует помнить, что в процессе ввода информации выполняется операция логического И над вводимыми данными и текущими (последними выводимыми) данными. Архитектура портов Р1 и Р2 и команды логических операций ANL и ORL представляют программисту эффективное средство маскирования для обработки однобитовых входов и выходов в МК.

Порт Р2 отличается от порта Р1 тем, что его младшие 4 бита могут быть использованы для расширения системы по вводу—выводу. Через младшую тетраду порта Р2 по специальным командам с обращением к портам Р4—Р7 возможен доступ к четырем внешним четы-

рехбитным портам ввода—вывода. Работа этих внешних портов синхронизируется сигналом ПРОГ/СТБВВ.

Порт ввода—вывода BUS. Порт BUS представляет собой двунаправленный буфер с тремя состояниями и предназначен для побайтного ввода, вывода или ввода—вывода информации.

Если порт BUS используется для двунаправленных передач, то обмен информацией через него выполняется по командам MOVX., а выводимый байт фиксируется в буферном регистре. При вводе вводимый байт в буферном регистре не фиксируется. В отсутствие передач порт BUS по своим выходам находится в высокоимпедансном состоянии.

Если порт BUS используется как однонаправленный, то вывод через него выполняется по команде OUTL, а ввод— по команде INS.

Вводимые и выводимые через порт BUS байты можно маскировать с помощью команд AND и OR, что позволяет выделять и обрабатывать в байте отдельный бит или группу бит. Для этого предварительно по команде OUTL BUS, A в порт BUS из аккумулятора должна быть загружена маска.

В МП-системах простой конфигурации, когда порт BUS не используется в качестве порта-расширителя системы, обмен выполняется по командам INS, OUTL и MOVX. Возможно попеременное использование команд OUTL и MOVX. Однако при этом необходимо помнить, что выводимый по команде OUTL байт фиксируется в буферном регистре порта BUS, а команда MOVX уничтожает содержимое буферного регистра порта.

В МП-системах, имеющих внешнюю память программ, порт BUS используется для выдачи адреса внешней памяти и для приема команды из внешней памяти программ. В таких системах использование

команды OUTL BUS недопустимо, так как фиксация в буферном регистре порта BUS выводимого байта явится причиной неправильной выборки следующей команды.

Синхронизация микроконтроллера. Опорную частоту синхронизации определяет или кварцевый осциллятор, подключаемый к входам XI и X2, или LC-цепь; XI является входом, а X2 - выходом генератора, способного работать в диапазоне частот от 1 до 11 МГц. На вход XI может подаваться сигнал от источника внешней синхронизации. Схема синхронизации МК показана на рис. 4.7, *а.* В нее входят два счетчика с модулями пересчета 3 и 5. Первый используется для формирования сигнала системной синхронизации, который может передаваться на вывод T0 после команды ENTO CLK. Этот же сигнал поступает на счетчик машинных циклов, на выходе которого через каждые пять сигналов синхронизации МК формируется сигнал САВП, идентифицирующий каждый машинный цикл и используемый в расширенных МП-системах для стробирования адреса внешней памяти.

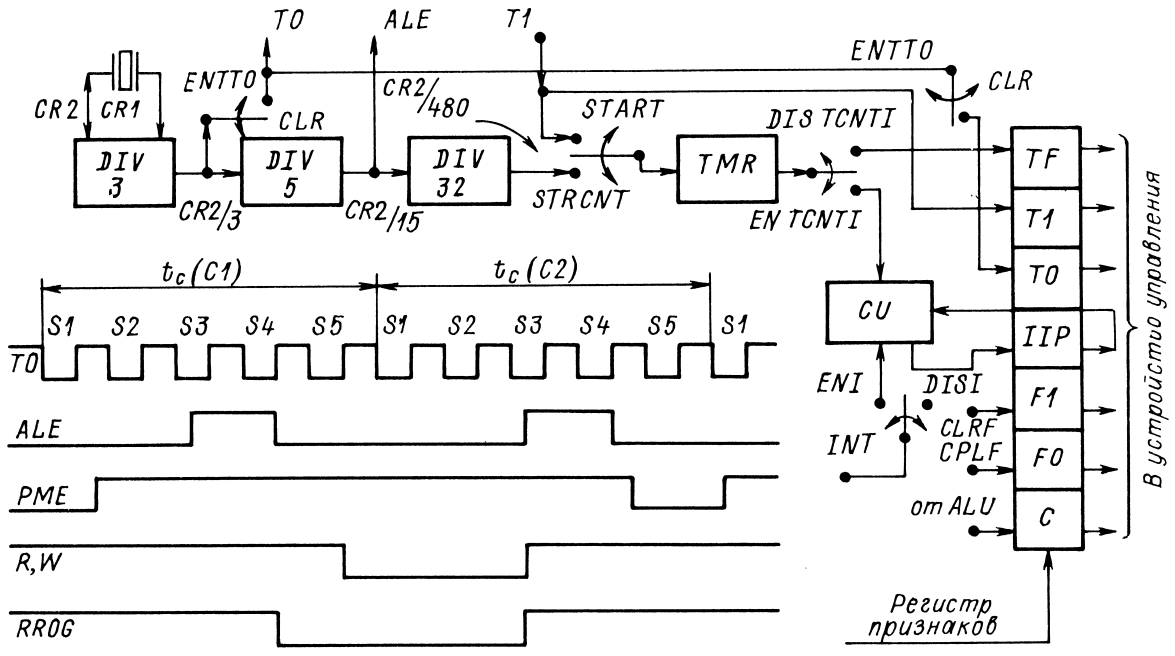

Рисунок 3.7 Временные диаграммы машинного цикла и цикла команды.

Временные диаграммы машинного цикла и цикла команды МК показаны на рисунке 3.7 [17].

Счетчик команд и логика условных переходов. Только 10 младших разрядов счетчика команд используются для адресации 1 Кбайт резидентной памяти программ, а 2 старших разряда—для обращения к внешней памяти программ в расширенных МП-системах.

При прерываниях или вызове подпрограмм все 12 разрядов счетчика команд загружаются в стек.

Кроме того, в стек загружается часть слова состояния процессора, в состав которого входят признаки переноса, вспомогательного переноса, признак пользователя F0 и признак банка регистров BS.

Логика условных переходов МК позволяет прикладной программе проверять не только внутренние признаки, но и условия, внешние по отношению к МК. По командам условного перехода в случае удовлетворения проверяемого условия в счетчик команд (разряды 0—7) из второго байта команды загружается адрес перехода. Логика переходов оперирует со следующим набором условий.

Устройство	Условие перехода	
Аккумулятор	Все нули	Не все нули
Выбранный бит аккумулятора	-	1
Признак переноса C	0	1
Признаки пользователя F0, F1	-	1
Признак переполнения таймера FT	-	1
Тестовые входы (T0, T1)	0	1
Вход запроса прерывания ЗПР	0	-

Система прерывания одноуровневая, так как после распознавания прерывания все последующие запросы прерывания игнорируются до тех пор, пока по команде возврата RETR вновь будет разрешена ра-

бота логики прерываний. При внешнем прерывании осуществляется передача управления ячейке 3. Это справедливо и для внутреннего прерывания, генерируемого признаком переполнения таймера, при котором передача управления осуществляется к ячейке 7.

При необходимости в МК можно создать двухуровневую систему прерываний. Для этого надо разрешить прерывания от таймера, загрузить в него число FF и перевести в режим подсчета внешних событий, фиксируемых на входе T1. Переход сигнала на входе T1 из состояния 1 в состояние 0 приведет к прерыванию по вектору в ячейке 7. В случае одновременного запроса прерываний от внешнего источника и запроса от признака переполнения таймера приоритет остается за источником, воздействующим на вход ЗПР.

При входе в подпрограммы обслуживания прерываний старший разряд счетчика команд $СК_{11}$ принудительно устанавливается в нуль. Следовательно, вся программа обработки прерывания должна быть размещена в блоке памяти 0. Выполнение команд SEL MB в процедуре обработки прерывания нежелательно, так как по этим командам изменяется только содержимое триггера признака блока памяти, но остается неизменным до выхода из прерывания (RETR) содержимое $СК_{11}$=0.

Таймер/счетчик событий. Внутренний 8-разрядный двоичный суммирующий счетчик может быть использован для формирования временных задержек и для подсчета внешних событий. Содержимое счетчика можно прочитать в аккумулятор или модифицировать из аккумулятора по командам MOV. Команды STRT T и STRT CNT настраивают и запускают счетчик в режиме таймера или в режиме счетчика событий соответственно. Остановить работу (но не сбросить со-

держимое) счетчика можно или командой STOP TCNT или сигналом системного сброса СБРОС.

Из состояния FF счетчик переходит в начальное состояние 00. При этом устанавливается в 1 признак переполнения счетчика, который может вызвать прерывание, если оно разрешено командой EN TCNTI. Прерывание от таймера может быть запрещено командой DIS TCNTI, но признак переполнения таймера может быть прочитан по команде условного перехода JTF. Выполнение команды JTF точно так же, как и переход к подпрограмме обработки прерывания по вектору с адресом 7, сбрасывает признак переполнения таймера.

В режиме таймера на вход счетчика через делитель частоты на 32 поступают основные синхросигналы машинного цикла САВП, и счетчик увеличивает свое состояние на 1.

В режиме счетчика событий внутренний счетчик увеличивает свое состояние на 1 каждый раз, когда сигнал на входе Т1 переходит из состояний 1 к состоянию 0.

В тех случаях, когда функционально-логических возможностей однокристального МП оказывается недостаточно, можно относительно простыми средствами расширить МП - систему до следующих размеров: память программ - до 4 Кбайт; память данных - до 320 байт; линии ввода/вывода — практически неограниченно.

МК - система с внешней памятью программ. Шина BUS по своим свойствам подобна двунаправленной шине данных микропроцессора К580, и все расширения МК выполняются с использованием этой шины. При обращении к резидентной памяти программ МК не генерирует внешних управляющих сигналов (за исключением САВП, который всегда идентифицирует каждый машинный цикл). Начиная с адреса 1024,

МК автоматически формирует управляющие сигналы, обеспечивающие выборку команд из внешней памяти.

Ячейки памяти с адресами, лежащими за пределами банка памяти 0, могут быть доступны после выполнения команды переключения банков памяти SELMB1, за которой должна следовать команда перехода (JMP или CALL). На рисунке 3.8 показана структура МК - системы с внешней памятью программ.

МП-система с внешней памятью данных. На рисунке 3.9 показана схема МК-системы, в состав которой включены две дополнительные БИС ОЗУ суммарной емкостью 256 байт. Внешняя оперативная память доступна МК по командам пересылки MOVX A, @ R и MOVX @R, A, которые по косвенному адресу (регистры R0 и R1) выполняют операции передачи байта между ВПД и аккумулятором. Так как косвенный адрес имеет формат байта, то схема на рисунке 3.9 обеспечивает адресацию 256 ячеек ОЗУ в дополнение к 64 ячейкам резидентной памяти данных МК048.

МК - система с расширенным вводом/выводом. Для сопряжения МК с объектом, имеющим большое число входов/ выходов, можно расширить резидентную систему ввода/вывода, подключив к МК необходимое количество внешних портов. Такое расширение может быть выполнено двумя способами: с использованием стандартного расширителя ввода/вывода (РВВ) КР580ВР43 или интерфейсных БИС /КР580ВВ55, КР580ВВ51) [12]. Расширитель подключается к МК048 так, как показано на рисунке 3.10.

Каждый из четырех портов РВВ может использоваться для ввода или вывода информации независимо от остальных и обеспечивает высокую нагрузочную способность.

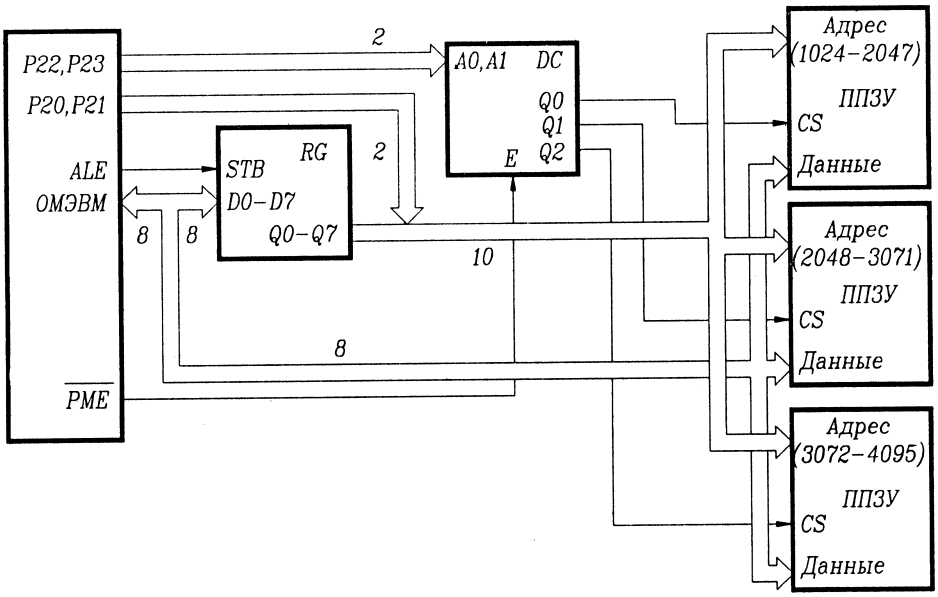

Рисунок 3.8 Структура МК - системы с внешней памятью программ.

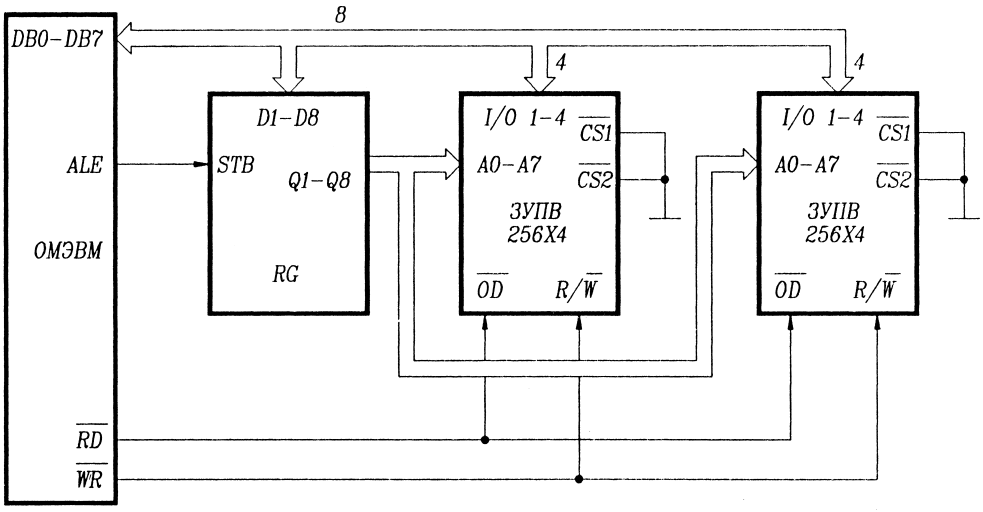

Рисунок 3.9 Структура МК - системы с внешней памятью данных.

Для вывода байта данных в порты P4 и P5 можно воспользоваться следующей последовательностью команд:

```
MOVD    P4, A;
SWAP    A
MOVD    P5, A;
```

На рисунке 3.10 показаны два варианта расширения ввода/вывода

с использованием параллельного периферийного адаптера (ППА) КР580ВВ55. В первом варианте порты адаптера адресуются

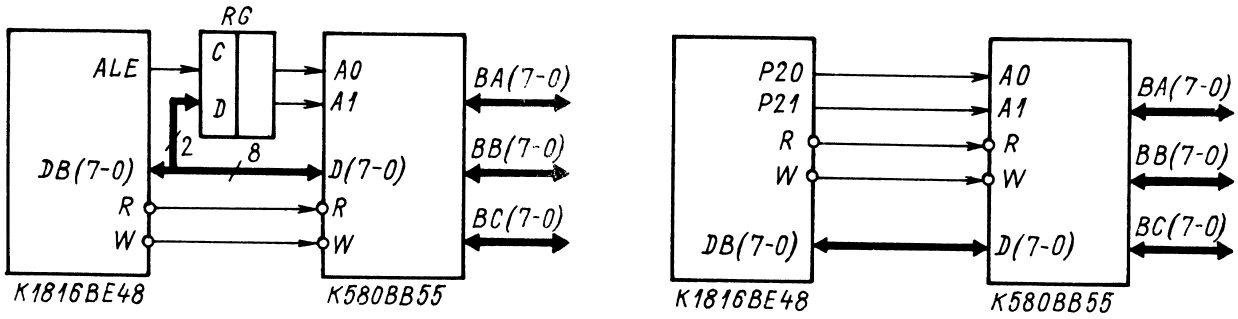

Рисунок 3.10 Расширение ввода/вывода МК048.

Варианты расширения ввода/вывода информации в МК-как ячейки ВПД, доступ к которым осуществляется по командам MOVX. Например, для вывода байта в порт В необходимо выполнить две команды:

```
MOV    R1, #1
MOVX   @R1,A
```

Для второго варианта подключения ППА выбор порта осуществляется через установку/сброс двух разрядов порта Р2. Так, для вывода байта данных в порт А можно воспользоваться следующими командами:

```
ANL    P2,#0FFH
OUTL   BUS,A
```

В МК048 предусмотрено расширение ввода/вывода с использованием специальной БИС расширителя ввода-вывода 8243. Доступ к расширителю ввода-вывода 8243 организуется с помощью команд MOVD, ANLD и ORLD, как и к портам Р3 - Р7. 4-разрядный код выбираемого порта и данные появляются на выводах Р23-Р20, и данные «защелкиваются» в фиксаторе по сигналу PROG.

Полная система команд ОМЭВМ приведена в приложении 2.

3.3. Однокристальные микроЭВМ семейства МК051

Использование ОМЭВМ семейства МК51 по сравнению с МК048 обеспечивает увеличение объема памяти команд и памяти данных. Новые возможности ввода-вывода и периферийных устройств расширяют диапазон применения и снижают общие затраты системы. В зависимости от условий использования, быстродействие системы увеличивается минимум в два с половиной раза и максимум в десять раз.

Семейство МК051 включает пять модификаций ОМЭВМ (имеющих идентичные основные характеристики), основное различие между которыми состоит в реализации памяти программ и мощности потребления [13].

Микросхемы	Аналог	Память программ, кбайт	Тип ПЗУ	Память данных, байт	$F_{такт.}$, МГц	$I_{потр.}$, mA
КР1816ВЕ31	8031АН	-	внешн.	128	12,0	150,0
КР1816ВЕ51	8051АН	4К	ПЗУ	128	12,0	150,0
КМ1816ВЕ751	8751Н	4К	ППЗУ	128	12,0	220,0
КР1830ВЕ31	80С31ВН	-	внешн.	128	12,0	18,0
КР1830ВЕ51	80С51ВН	4К	ПЗУ	128	12,0	18,0

ОМЭВМ содержит:

1) центральный восьмиразрядный процессор;

2) память программ объемом 4 Кбайт (только КМ1816ВЕ751, КР1816ВЕ051 и КР1830ВЕ051);

3) память данных объемом 128 байт;

4) четыре восьмиразрядных программируемых канала ввода-вывода;

5) два 16-битовых многорежимных таймера/счетчика;

6) систему прерываний с пятью векторами и двумя уровнями;

7) последовательный интерфейс;

8) тактовый генератор.

Система команд ОМЭВМ содержит 111 базовых команд с форматом 1, 2, или 3 байта [16-17].

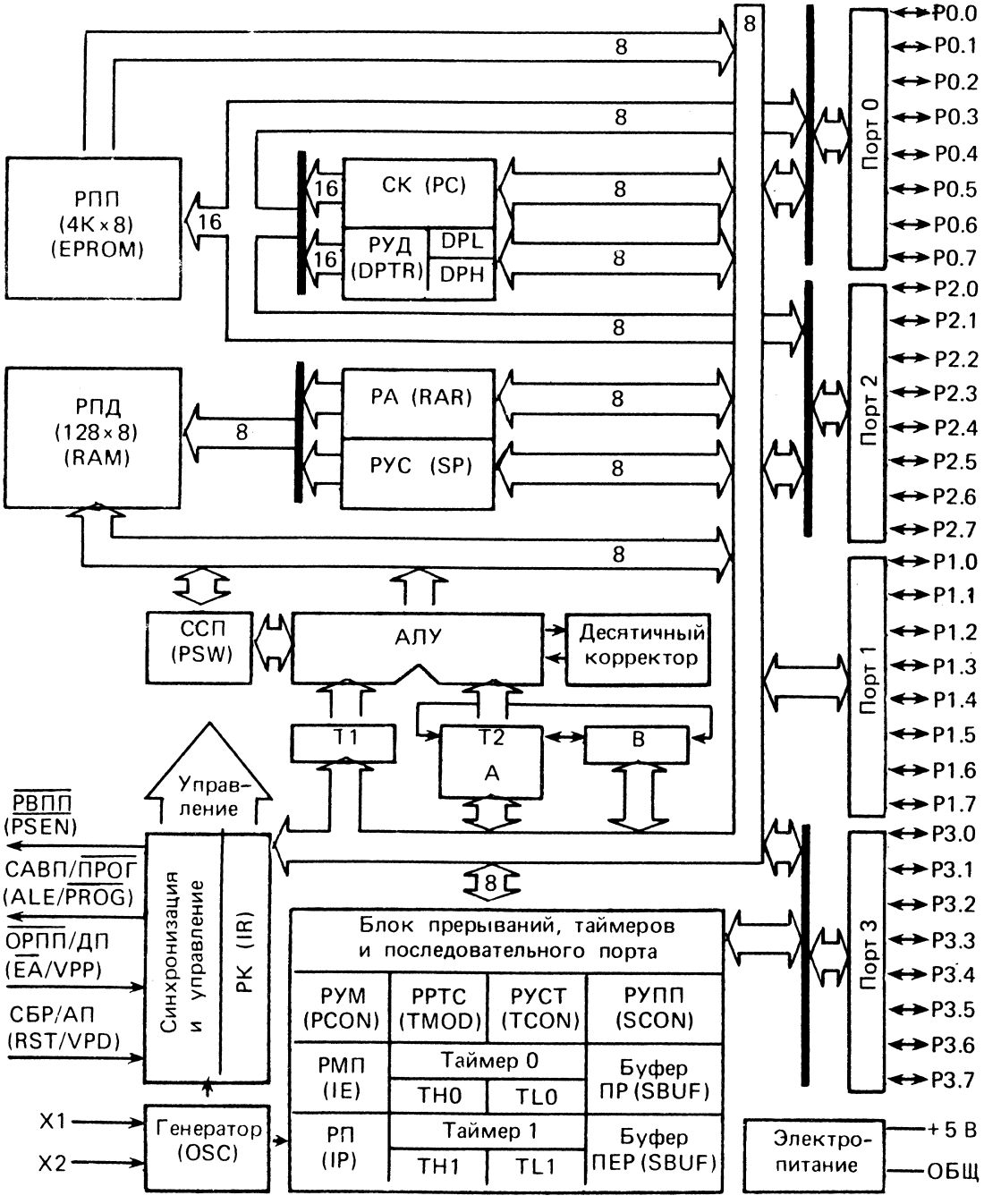

Рисунок 3.11. Структурная схема ОМЭВМ семейства МК051.

Основу структурной схемы МК051 (рисунок 3.11) образует внутренняя двунаправленная 8-битная шина, которая связывает между

собой все основные узлы и устройства: резидентную память, АЛУ, блок регистров специальных функций, устройство управления и порты ввода/вывода.

Арифметическо-логическое устройство. 8-битное АЛУ может выполнять арифметические операции сложения, вычитания, умножения и деления; логические операции И, ИЛИ, исключающее ИЛИ, а также операции циклического сдвига, сброса, инвертирования и т. п. В АЛУ имеются программно недоступные регистры Т1 и Т2, предназначенные для временного хранения операндов, схема десятичной коррекции и схема формирования признаков.

Важной особенностью АЛУ является его способность оперировать не только байтами, но и битами. Отдельные программно-доступные биты могут быть установлены, сброшены, инвертированы, переданы, проверены и использованы в логических операциях. Для управления объектами часто применяются алгоритмы, содержащие операции над входными и выходными булевскими переменными (истина/ложь), реализация которых средствами обычных микропроцессоров сопряжена с определенными трудностями.

Организация памяти. Все ОМЭВМ семейства МК051 имеют несколько адресных пространств, функционально и логически разделенных за счет разницы в механизмах адресации и сигналах управления записью и чтением:

— память программ;

— внутренняя память данных;

— внешняя память данных.

Структура адресного пространства ОМЭВМ показана на рисунке 3.12. Слева приводятся адреса соответствующих областей памяти.

Память программ. Память программ доступна только по чтению. ОМЭВМ не имеют команд и управляющих сигналов, предназначенных для записи в память программ. Память программ имеет байтовую организацию и общий объем до 64 Кбайт.

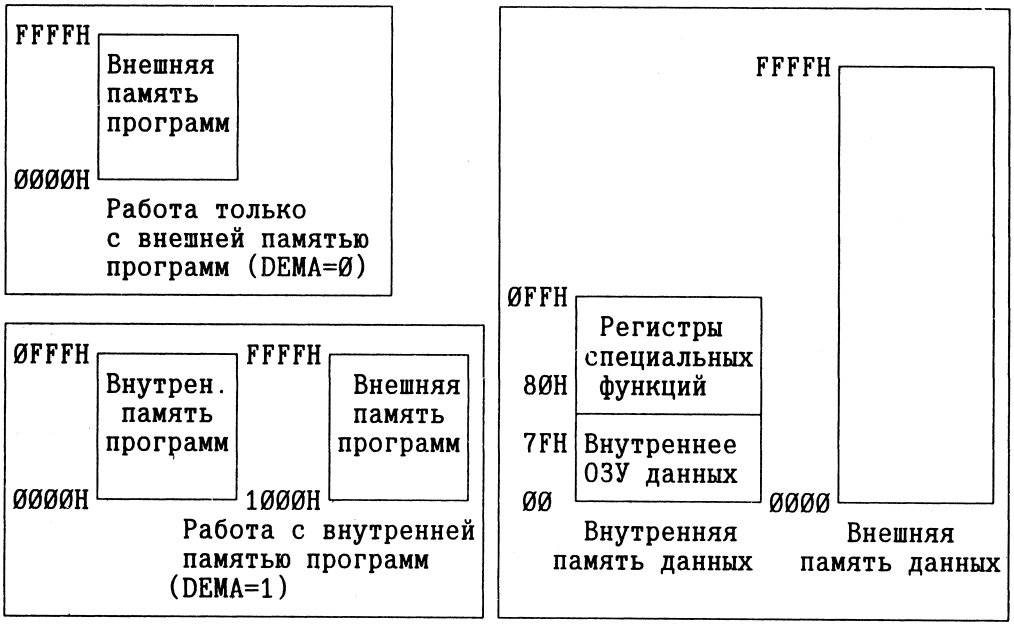

Рисунок 3.12 Пространство памяти ОМЭВМ МК051.

С точки зрения программиста имеется только один вид памяти программ объемом 64 К. Тот факт, что в ряде ОМЭВМ он образуется комбинацией массивов, находящихся на кристалле и вне его, в соотношении 4 К/60 К для программиста неощутим, так как АЛУ автоматически выбирает байт из соответствующего массива в соответствии с его адресом. Младшие адреса памяти программ, как правило, отводятся под обработку прерываний и начало работы ОМЭВМ после сброса.

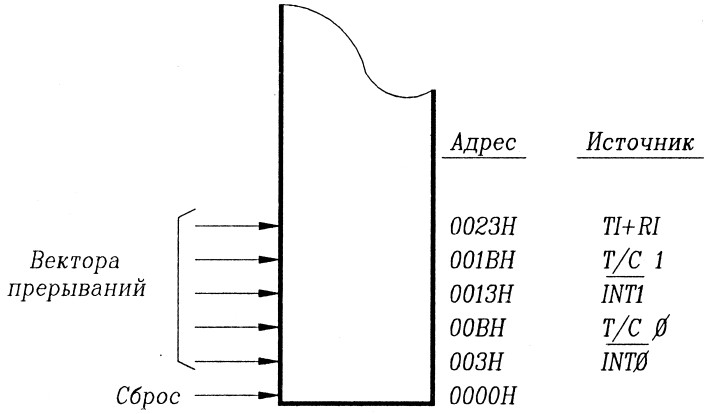

Рисунок 3.13 Распределение младших адресов памяти.

Внутренняя память данных ОМЭВМ состоит из двух областей: 128 байт оперативной памяти (ОЗУ) с адресами 0—7FH и области регистров специальных функций, занимающей адреса 80H—FFH. Распределение пространства внутренней памяти данных показано на рисунке 3.14.

Младшие 32 байта внутреннего ОЗУ данных сгруппированы в 4 банка по 8 регистров в каждом (БАНК0 - БАНК3). Команды программы могут обращаться к регистрам, используя их имена R0 - R7. Два бита PSW (указатели банка рабочих регистров RS0 и RS1) определяют, с регистрами какого банка производятся манипуляции. Следующие после банков регистров внутреннего ОЗУ данных 16 байт (адреса 20H - 2FH) образуют область ячеек, к которым возможна побитовая адресация. Набор команд ОМЭВМ семейства МК051 содержит значительное количество инструкций, позволяющих работать с отдельными битами, используя при этом прямую адресацию. 128 бит, составляющих рассматриваемую область внутреннего ОЗУ данных, имеют адреса 00H - 7FH и предназначены для работы с такими инструкциями. Битовая адресация ОЗУ показана на рисунке 3.15, где в квадратах, символизирующих биты, указаны их адреса.

Обращение к внутреннему ОЗУ данных всегда осуществляется с использованием 8-разрядного адреса.

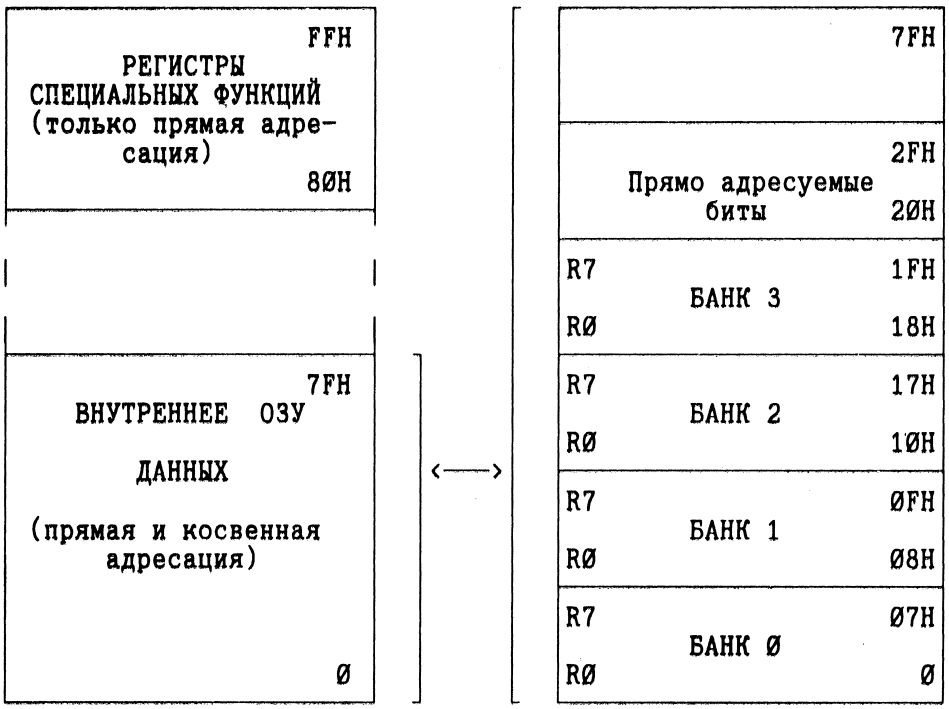

Рисунок 3.14 Адресное пространство внутренней памяти данных.

Область регистров специальных функций содержит защелки портов, регистры таймеров/счетчиков, регистры управления и т.п. Полный список регистров специальных функций с их адресами приведен в таблице 3.1.

Таблица3.1

Блок регистров специальных функций

Символ		Наименование	Адрес
*	ACC	Аккумулятор	0E0H
*	B	Регистр-расширитель аккумулятора	0F0H
*	PSW	Слово состояния программы	0D0H
	SP	Регистр-указатель стека	81H
	DPTR	Регистр-указатель данных (DPH)	83H
		(DPL)	82H
*	P0	Порт 0	80H

*	P1	Порт 1	90H
*	P2	Порт 2	0A0H
*	P3	Порт 3	0B0H
*	IP	Регистр приоритетов	0B8H
*	IE	Регистр маски прерываний	0A8H
	TMOD	Регистр режима таймера/счетчика	89H
*	TCON	Регистр управления/статуса таймера	88H
	TH0	Таймер 0 (старший байт)	8CH
	TL0	Таймер 0 (младший байт)	8AH
	TH1	Таймер 1 (старший байт)	8DH
	TL1	Таймер 1 (младший байт)	8B H
*	SCON	Регистр управления приемопередатчиком	98H
	SBUF	Буфер приемопередатчика	99 H
	PCON	Регистр управления мощностью	87H
Примечание. Регистры, имена которых отмечены знаком (*), допускают адресацию отдельных бит.			

Эти регистры допускают только прямую адресацию. Двенадцать байт в области регистров специальных функций допускают как байтовую, так и побитовую адресацию. Список регистров с побитовой адресацией приведён в таблице 3.2. Побитовую адресацию допускают те регистры специальных функций, чей адрес заканчивается 000B. Биты в рассматриваемой области регистров специальных функций имеют адреса 80H—F7H.

Таблица 3.2

Битовая адресация ОЗУ

Адрес байта	Адрес бита							
	D7	D6	D5	D4	D3	D2	**D1**	D0
2FH	7F	7E	7D	7C	7B	7A	79	78
2EH	77	76	75	74	73	72	71	70
2DH	6F	6E	6D	6C	6B	6A	69	68

2CH	67	66	65	64	63	62	61	60
2BH	5F	5E	5D	5C	5B	5A	59	58
2AH	57	56	55	54	53	52	51	50
29H	4F	4E	4D	4C	4B	4A	49	48
28H	47	46	45	44	43	42	41	40
27H	3F	3E	3D	3C	3B	3A	39	38
26H	37	36	35	34	33	32	31	30
25H	2F	2E	2D	2C	2B	2A	29	28
24H	27	26	25	24	23	22	21	20
23H	IF	1E	ID	1C	1B	1A	19	18
22H	17	16	15	14	13	12	11	10
21H	0F	0E	0D	0C	0B	0A	09	08
20H	07	06	05	04	03	02	01	00

1FH	БАНК 3	R7
18H		R0
17H	БАНК 2	R7
10H		R0
0FH	БАНК 1	R7
08H		R0
07H	БАНК 0	R7
00		R0

Таблица 3.3

Адреса битов регистров специальных функций

Прямые адреса байтов	**Биты**								**Идентифика-торы регист-ров**
	D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**	
0F0H	F7	F6	F5	F4	F3	F2	Fl	F0	B
0E0H	E7	E6	E5	E4	E3	E2	El	E0	**ACC**
	CV	AC	F0	RS1	RS0	0V		P	
0D0H	D7	D6	D5	D4	D3	D2	Dl	D0	PSW
			PT2	PS	PT1	PX1	PT0	PX0	
0B8H	-	-	BD	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	Bl	B0	P3
	EA		ET2	ES	ET1	EX1	ET0	EX0	
0A8H	AF	-	AD	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	Al	A0	P2
	SM0	**SM1**	**SM2**	**REN**	**TB8**	**RB8**	**TI**	**RI**	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	PI

	TF1	TR1	TE0	TR0	IE1	IT1	IE0	IT0	
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
80H	87	86	85	84	83	82	81	80	P0

При выполнении команд в АЛУ формируется ряд признаков операции (флагов), которые фиксируются в регистре ССП, формат которого представлен в таблице 3.4.

Таблица 3.4

Формат слова состояния программы (ССП)

Символ	**Позиция**	Имя и назначение
C	PSW. 7	Флаг переноса. Устанавливается и сбрасывается аппаратурными средствами или программой при выполнении арифметических и логических операций
AC	PSW. 6	Флаг вспомогательного переноса. Устанавливается и сбрасывается только аппаратурными средствами при выполнении команд сложения и вычитания и сигнализирует о переносе или заеме в бите 3
F0	PSW. 5	Флаг 0. Может быть установлен, сброшен или проверен программой как флаг, специфицируемый пользователем
RS1	PSW. 4	Выбор банка регистров. Устанавливается и сбрасывается программой для выбора рабочего банка регистров (см. примечание)
RS0	PSW. 3	
OV	PSW. 2	Флаг переполнения. Устанавливается и сбрасывается аппаратурно при выполнении арифметических операций
-	PSW. 1	**Не используется**

P	PSW. 0	Флаг паритета. Устанавливается и сбрасывается аппаратурно в каждом цикле команды и фиксирует нечетное/четное число единичных бит в аккумуляторе, т. е. выполняет контроль по четности

Таблица 3.5

Выбор рабочего банка регистров

RS1	RS0	Банк	Границы адресов
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H 1FH

Флаг переноса (C) выполняет функции "булевого аккумулятора" в битовых командах. Флаг переполнения (OV) фиксирует арифметическое переполнение при операциях над целыми числами со знаком и делает возможным использование арифметики в дополнительных кодах. Значение флагов полностью RS0, RS1 определяется прикладной программой, и используется для выбора одного из четырех регистровых банков.

В МК051 имеется возможность выполнять команды без участия аккумулятора. Например, данные могут быть переданы из любой ячейки РПД в любой регистр, любой регистр может быть загружен непосредственным операндом и т.д. Многие логические операции могут быть выполнены без участия аккумулятора. Переменные могут быть инкрементированы, декрементированы и проверены (test) без использования аккумулятора.

Регистры-указатели, 8-битный указатель стека (РУС) может адресовать любую область РПД. Его содержимое инкрементируется прежде, чем данные будут запомнены в стеке в ходе выполнения команд

PUSH и CALL. В процессе инициализации МК051 после сигнала СБР в РУС автоматически загружается код 07Н. Это значит, что если прикладная программа не переопределяет стек, то первый элемент данных в стеке будет располагаться в ячейке РПД с адресом 08Н.

Двухбайтный регистр-указатель данных (РУД) обычно используется для фиксации 16-битного адреса в операциях с обращением к внешней памяти. Командами МК051 регистр-указатель данных может быть использован или как 16-битный регистр, или как два независимых 8-битных регистра (DPH и DPL).

Таймер/счетчик. В составе средств МК051 имеются регистровые пары с символическими именами TH0, TL0 и TH1, TL1, на основе которых функционируют два независимых программно-управляемых 16-битных, таймера/счетчика событий.

Буфер последовательного порта. Регистр с символическим именем SBUF представляет собой два независимых регистра - буфер приемника и буфер передатчика. Загрузка байта в SBUF немедленно вызывает начало процесса передачи через последовательный порт. Когда байт считывается из SBUF, это значит, что его источником является приемник последовательного порта.

Регистры специальных функций. Регистры с символическими именами IP, IE, TMOD, TCON, SCON и PCON используются для фиксации и программного изменения управляющих бит и бит состояния схемы прерывания, таймера/счетчика, приемопередатчика последовательного порта и для управления мощностью электропитания МК051.

Порты ввода/вывода информации. Все четыре порта МК051 предназначены для ввода или вывода информации побайтно. Каждый порт содержит управляемые регистр-защелку, входной буфер и выходной драйвер.

Выходные драйверы портов 0 и 2, а также входной буфер порта 0 используются при обращении к внешней памяти (ВП). При этом через порт 0 в режиме временного мультиплексирования сначала выводится младший байт адреса ВП, а затем выдается или принимается байт данных. Через порт 2 выводится старший байт адреса в тех случаях, когда разрядность адреса равна 16 бит.

Все выводы порта 3 могут быть использованы для реализации альтернативных функций, перечисленных в таблице 3.6. Альтернативные функции могут быть задействованы путем записи 1 в соответствующие биты регистра-защелки (Р3.0—Р3.7) порта 3.

Порт 0 является двунаправленным, а порты 1, 2 и 3 - квазидвунаправленными. Каждая линия портов может быть использована независимо для ввода или вывода информации. Для того чтобы некоторая линия порта использовалась для ввода, в D-триггер регистра-защелки порта должна быть записана 1. По сигналу СБР в регистры-защелки всех портов автоматически записываются единицы, настраивающие их тем самым на режим ввода.

Все порты могут быть использованы для организации ввода/вывода информации по двунаправленным линиям передачи. Однако порты 0 и 2 не могут быть использованы для этой цели в случае, если МП-система имеет внешнюю память, связь с которой организуется через общую разделяемую шину адреса/данных, работающую в режиме временного мультиплексирования.

Таблица 3.6

Альтернативные функции порта 3

Символ	**Позиция**	Имя и назначение
RD	Р3.7	Чтение. Активный сигнал низкого уровня формируется аппаратурно при обращении к ВПД

WR	P3.6	Запись. Активный сигнал низкого уровня формируется аппаратурно при обращении к ВПД
T1	P3.5	Вход таймера/счетчика 1 или тест-вход
T0	P3.4	Вход таймера/счетчика 0 или тест-вход
INT1	P3.3	Вход запроса прерывания 1. Воспринимается сигнал низкого уровня или срез
INTO	P3.2	Вход запроса прерывания 0. Воспринимается сигнал низкого уровня или срез
TXD	P3.1	Выход передатчика последовательного порта в режиме УАПП. Выход синхронизации в режиме сдвигающего регистра
RXD	P3.0	Вход приемника последовательного порта в режиме УАПП. Ввод/вывод данных в режиме сдвигающего регистра

Особенности работы портов. Обращение к портам ввода/вывода возможно с использованием команд, оперирующих с байтом, отдельным битом и произвольной комбинацией бит. При этом в тех случаях, когда порт является одновременно операндом и местом назначения результата, устройство управления автоматически реализует специальный режим, который называется "чтение-модификация-запись". Этот режим обращения предполагает ввод сигналов не с внешних выводов порта, а из его регистра-защелки, что позволяет исключить неправильное считывание ранее выведенной информации.

Таймер/счетчик. Два программируемых 16-битных таймера/счетчика. (T/C0 и T/C1) могут быть использованы в качестве таймеров или счетчиков внешних событий. При работе в качестве таймера содержимое T/C инкрементируется в каждом машинном цикле, т.е. через каждые 12 периодов резонатора. При работе в качестве счетчи-

ка содержимое Т/С инкрементируется под воздействием перехода из 1 в 0 внешнего входного сигнала, подаваемого на соответствующий (Т0, Т1) вывод МК051. Для управления режимами работы Т/С и для организации взаимодействия таймеров с системой прерывания используются два регистра специальных функций (РРТС и РУСТ), описание которых приводится в таблицах 3.7 и 3.8 соответственно. Как следует из описания управляющих бит РРТС, для обоих Т/С режимы работы 0,1 и 2 одинаковы. Режимы 3 для Т/СО и Т/С1 различны.

Режим 0. Перевод любого Т/С в режим 0 делает его похожим на таймер МК48 (8-битный счетчик), на вход которого подключен 5-битный предделитель частоты на 32.. В этом режиме таймерный регистр имеет разрядность 13 бит. При переходе из состояния "все единицы" в состояние "все нули" устанавливается флаг прерывания от таймера TF1.

Таблица 3.7

Регистр режима работы таймера/счетчика

Символ	Позиция	Имя и назначение
GATE	TMOD.7 для Т/С1 и TMOD.3 для Т/С0	Управление блокировкой. Если бит установлен, то таймер/счет чик *"x"* разрешен до тех пор, пока на входе "INT x" высокий уровень и бит управления "TRx" установлен. Если бит сброшен, то Т/С разрешается, как только бит управления "TRx" устанавливается
C/T	TMOD.6 для Т/С1 и TMOD.2 для Т/С0	Бит выбора режима таймер а или счетчика событий. Если бит сброшен, то работает таймер от внутреннего источника сигналов синхронизации. Если бит установлен, то работает счетчик от внешних сигналов на входе "Тх"
Ml	TMOD.5 для Т/С1 и TMOD.1	Режим работы (см. примечание)

	для T/C0	
M0	TMOD.4 для T/C1 и TMOD 0 для T/C0	

Примечание

MI	M0	**Режим работы**
o	o	Таймер МК48. "TLx" работает как 5-битный предделитель
o	1	16-битный таймер/счетчик. "THx" и "TLx" включены последовательно
1	o	8-битный автоперезагружаемый таймер/счетчик. "THx" хранит значение, которое должно быть перезагружено в "TLx" каждый раз по переполнению
1	1	Таймер/счетчик 1 останавливается. Таймер/счетчик 0:TL0 работает как 8-битный таймер/счетчик, и его режим определяется управляющими битами таймера 0. TH0 работает только как 8-битный таймер, и его режим определяется управляющими битами таймера 1

Режим 1. Работа любого T/C в режиме 1 такая же, как и в режиме 0, за исключением того, что таймерный регистр имеет разрядность 16 бит.

Режим 2. В режиме 2 работа организована таким образом, что переполнение (переход из состояния "все единицы" в состояние "все нули") 8-битного счетчика TL1 приводит не только к установке флага TF1 (рис. 3.10, б), но и автоматически перезагружает в TL1 содержимое старшего байта (TH1) таймерного регистра, которое предвари-

тельно было задано программным путем. Перезагрузка оставляет содержимое TH1 неизменным.

Режим 3. В режиме 3 T/C0 и T/C1 работают по-разному. T/C1 сохраняет неизменным свое текущее содержимое. Иными словами, эффект такой же, как и при сбросе управляющего бита TR1 в нуль.

В режиме 3 TL0 и TH0 функционируют как два независимых счетчика. Работу TL0 определяют управляющие биты T/C0 (C/T, GATE, TR0), входной сигнал INT0 и флаг переполнения TF0. Работу TH0, который может выполнять только функции таймера (подсчет машинных циклов МК), определяет управляющий бит TR1. При этом TH0 использует флаг переполнения TF 1.

Режим 3 используется в тех случаях применения МК51, когда требуется наличие дополнительного 8-битного таймера или счетчика событий.

Таблица 3.8

Регистр управления/статуса таймера

TF1	TCON.7	Флаг переполнения таймера 1. Устанавливается аппаратурно при переполнении таймера/счетчика. Сбрасывается при обслуживании прерывания аппаратурно.
TR1	TCON.6	Бит управления таймера 1. Устанавливается/сбрасывается программой для пуска/останова
TF0	TCON.5	Флаг переполнения таймера 0. Устанавливается аппаратурно. Сбрасывается при обслуживании прерывания
TR0	TCON.4	Бит управления таймера 0. Устанавливается/сбрасывается программой для пуска/останова таймера/счетчика
IE1	TCON.3	Флаг фронта прерывания 1. Устанавливается

		аппаратурно, когда детектируется срез внешнего сигнала ЗПР1 (INT1). Сбрасывается при обслуживании прерывания
IT1	TCON.2	Бит управления типом прерывания 1. Устанавливается/ сбрасывается программно для спецификации запроса ЗПР1 (срез/низкий уровень)
IE0	TCON.1	Флаг фронта прерывания 0. Устанавливается по срезу сигнала ЗПР0. Сбрасывается при обслуживании прерывания
IT0	TCON.0	Бит управления типом прерывания 0. Устанавливается/ сбрасывается программно для спецификации запроса ЗПР0 (срез/низкий уровень)

Последовательный интерфейс. Через универсальный асинхронный приемопередатчик (УАПП) осуществляется прием и передача информации, представленной последовательным кодом (младшими битами вперед), в полном дуплексном режиме обмена. В состав УАПП, называемого часто последовательным портом, входят принимающий и передающий сдвигающие регистры, а также специальный буферный регистр (SBUF) приемопередатчика. Запись байта в буфер приводит к автоматической переписи байта в сдвигающий регистр передатчика и инициирует начало передачи байта. Наличие буферного регистра приемника позволяет совмещать операцию чтения ранее принятого байта с приемом очередного байта. Если к моменту окончания приема байта предыдущий байт не был считан из SBUF, то он будет потерян.

Последовательный порт МК051 может работать в четырех различных режимах.

Режим 0. В этом режиме информация и передается и принимается через внешний вывод входа приемника (RXD). Принимаются или передаются 8 бит данных. Через внешний вывод выхода передатчика (TXD) выдаются импульсы сдвига, которые сопровождают каждый бит. Частота передачи бита информации равна 1/12 частоты резонатора.

Режим 1. В этом режиме передаются через TXD или принимаются из RXD 10 бит информации: старт-бит (0), 8 бит данных и стоп-бит (1). Скорость приема/передачи - величина переменная и задается таймером.

Режим 2. В этом режиме через TXD передаются или из RXD принимаются 11 бит информации: старт-бит, 8 бит данных, программируемый девятый бит и стоп-бит. При передаче девятый бит данных может принимать значение 0 или 1, или, например, для повышения достоверности передачи путем контроля по четности в него может быть помещено значение признака паритета из слова состояния программы (PSW.0). Частота приема/передачи выбирается программой и может быть равна либо 1/32, либо 1/64 частоты резонатора в зависимости от управляющего бита SMOD.

Режим 3. Режим 3 совпадает с режимом 2 во всех деталях, за исключением частоты приема/передачи, которая является величиной переменной и задается таймером.

Регистр управления/статуса УАПП. Управление режимом работы УАПП осуществляется через специальный регистр с символическим именем SCON. Этот регистр содержит не только управляющие биты, определяющие режим работы последовательного порта, но и девятый бит принимаемых или передаваемых данных (RB8 и TB8) и биты прерывания приемопередатчика (RI и TI).

Таблица 3.9

Регистр управления/статуса УАПП

Символ	Позиция	Имя и назначение
SM0 SM1	SCON.7 SCON.6	Биты управления режимом работы УАПП. Устанавливаются/ сбрасываются программно (см. примечание)
SM2	SCON.5	Бит управления режимом УАПП. Устанавливается программно для запрета приема сообщения, в котором девятый бит имеет значение 0
REN	SCON.4	Бит разрешения приема. Устанавливается/сбрасывается программно для разрешения/запрета приема последовательных данных
TB8	SCON.3	Передача бита 8. Устанавливается/сбрасывается программно для задания девятого передаваемого бита в режиме УАПП-9 бит
RB8	SCON .2	Прием бита 8. Устанавливается/сбрасывается аппаратурно для фиксации девятого принимаемого бита в режиме УАПП-9 бит
TI	SCON.1	Флаг прерывания передатчика. Устанавливается аппаратурно при окончании передачи байта. Сбрасывается программно после обслуживания прерывания
RI	SCON.0	Флаг прерывания приемника. Устанавливается аппаратурно при приеме байта. Сбрасывается программно после обслуживания прерывания

Примечание

SM0	SM1	Режим работы УАПП
0	0	Сдвигающий регистр расширения ввода/вывода
0	1	УАПП-8бит. Изменяемая скорость передачи
1	0	УАПП-9 бит. фиксированная скорость передачи

1	1	УАПП-9 бит. Изменяемая скорость передачи

Прикладная программа путем загрузки в старшие биты регистра SCON 2-битного кода определяет режим работы УАПП. Во всех четырех режимах работы передача из УАПП инициируется любой командой, в которой буферный регистр SBUF указан как получатель байта. Прием в УАПП в режиме 0 осуществляется при условии, что RI = 0 и REN = 1 В режимах 1, 2, 3 прием начинается с приходом старт-бита, если REN = 1.

В бите TB8 программно устанавливается значение девятого бита данных, который будет передан в режиме 2 или 3. В бите RB8 фиксируется в режимах 2 и 3 девятый принимаемый бит данных; В режиме 1, если SM2 = 0, в бит RB8 заносится стоп-бит. В режиме 0 бит RB8 не используется.

Флаг прерывания передатчика TI устанавливается аппаратурно в конце периода передачи восьмого бита данных в режиме 0 и в начале периода передачи стоп-бита в режимах 1,2 и 3. Соответствующая подпрограмма обслуживания прерывания должна сбрасывать бит TI.

Флаг прерывания приемника RI устанавливается аппаратурно в конце периода приема восьмого бита данных в режиме 0 и в середине периода приема стоп-бита в режимах 1, 2 и 3. Подпрограмма обслуживания прерывания должна сбрасывать бит RI.

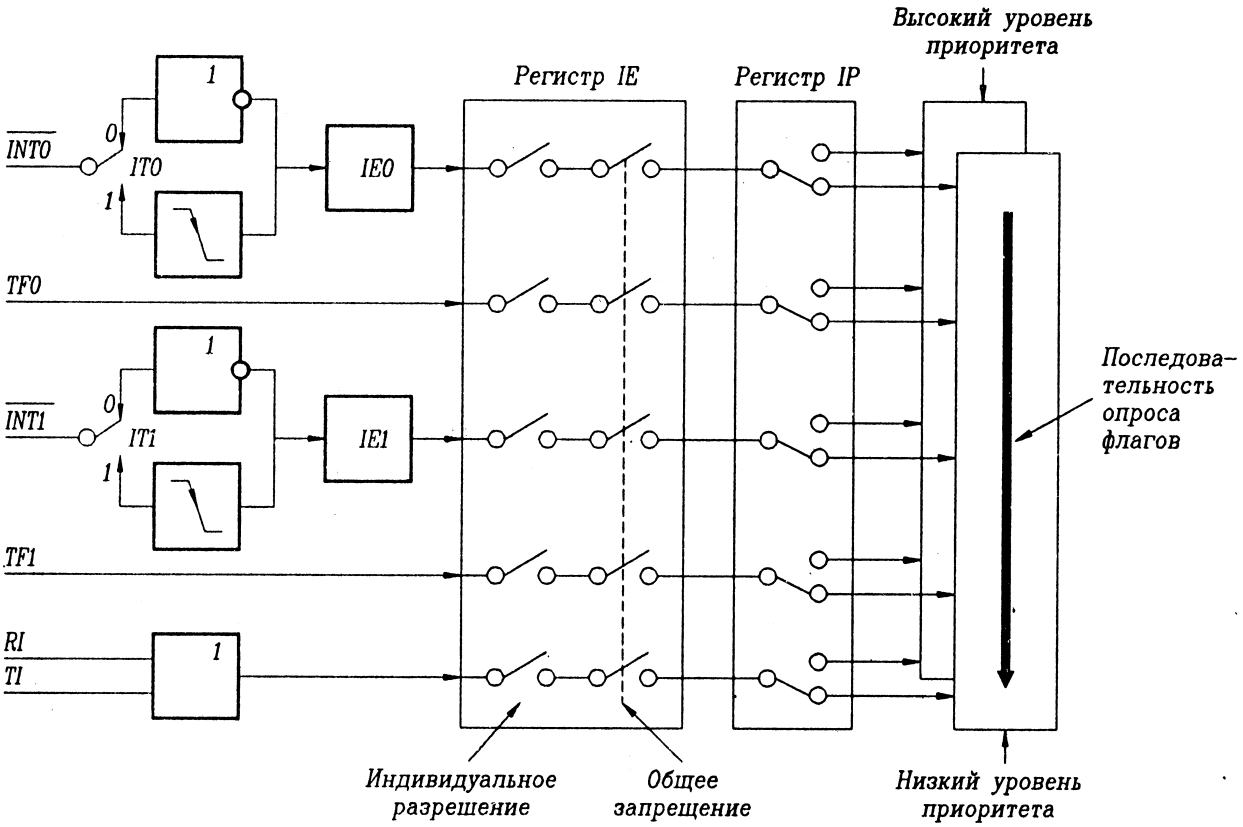

Рисунок 3.16. Схема прерываний ОМЭВМ семейства 051.

Адреса прерывающих подпрограмм

Источник прерывания	Вектор прерывания
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
TI + RI	0023H

Система прерываний. Внешние прерывания INT0 и INT1 могут быть вызваны либо уровнем, либо переходом сигнала из 1 в 0 на входах МК051 в зависимости от значений управляющих бит IT0 и IT1 в регистре TCON. От внешних прерываний устанавливаются флаги IE0 и IE1 в регистре TCON, которые инициируют вызов соответствующей подпрограммы обслуживания прерывания.

Сброс этих флагов выполняется аппаратурно только в том случае, если прерывание было вызвано по переходу (срезу) сигнала. Если же прерывание вызвано уровнем входного сигнала, то сбросом флага IE управляет соответствующая подпрограмма обслуживания прерывания путем воздействия на источник прерывания с целью снятия им запроса.

Флаги запросов прерывания от таймеров TF0 и TF1 сбрасываются автоматически при передаче управления подпрограмме обслуживания. флаги запросов прерывания RI и TI устанавливаются блоком управления УАПП аппаратурно, но сбрасываться должны программой.

В блоке регистров специальных функций есть два регистра, предназначенных для управления режимом прерываний и уровнями приоритета IE и IP.

Таблица 3.10

Регистр масок прерывания (РМП)

Символ	Позиция	Имя и назначение
EA	IE.7	Снятие блокировки прерываний. Сбрасывается программно для запрета всех прерываний независимо от состояний IE4 – IE0
-	IE.6	Не используются
-	IE.5	Не используются
ES	IE.4	Бит разрешения прерывания от УАПП. Установка/сброс программой для разрешения/запрета прерываний от флагов TI или RI
ET1	IE.3	Бит разрешения прерывания от таймера 1. Установка/сброс программой для разрешения/запрета прерываний от таймера 1
EX1	IE.2	Бит разрешения внешнего прерывания 1. Установ-

		ка/сброс программой для разрешения/запрета прерываний
ET0	IE.1	Бит разрешения прерывания от таймера 0. Работает аналогично IE.3
EXD	IE.0	Бит разрешения внешнего прерывания 0. Работает аналогично IE.2

Таблица 3.11

Регистр приоритетов прерываний

Символ	Позиция	**Имя и назначение**
-	IP.7-IP.5	Не используются
PS	IP.4	Бит приоритета УАПП. Установка/сброс программой для присваивания прерыванию от УАПП высшего/низшего приоритета
PT1	IP.3	Бит приоритета таймера 1. Установка/сброс программой для присваивания прерыванию от таймера 1 высшего/низшего приоритета
PX1	IP. 2	Бит приоритета внешнего прерывания 1. Установка/сброс программой для присваивания высшего/низшего приоритета внешнему прерыванию INT1
PT0	IP.1	Бит приоритета таймера 0. Работает аналогично IP.3
PX0	IP.0	Бит приоритета внешнего прерывания 0. Работает аналогично IP. 2

Полная система команд ОМЭВМ семейства МК051 приведена в приложении 3.

3.4. PIC – контроллеры семейства 16C84

PIC контроллеры представляют собой новый класс быстродействующих микроконтроллеров выполненных на основе RISC архитектуры и предназначенные для построения интерфейсных устройств различного назначения.

PIC16C84 имеет внутреннее 1K x 14 бит EEPROM для программ, 8-битовые данные и 64байт EEPROM памяти данных. При этом отличаются низкой стоимостью и высокой производительностью Все команды состоят из одного слова (14 бит шириной) и исполняются за один цикл (400 нс при 10 МГц), кроме команд перехода, которые выполняются за два цикла (800 нс). PIC16C84 имеет прерывание, срабатывающее от четырех источников, и восьмиуровневый аппаратный стек. Периферия включает в себя 8-битный таймер/счетчик с 8-битным программируемым предварительным делителем (фактически 16 - битный таймер) и 13 линий двунаправленного ввода/вывода. Высокая нагрузочная способность (25 мА макс. втекающий ток, 20 мА макс. вытекающий ток) линий ввода/вывода упрощают внешние драйверы и, тем самым, уменьшается общая стоимость системы.

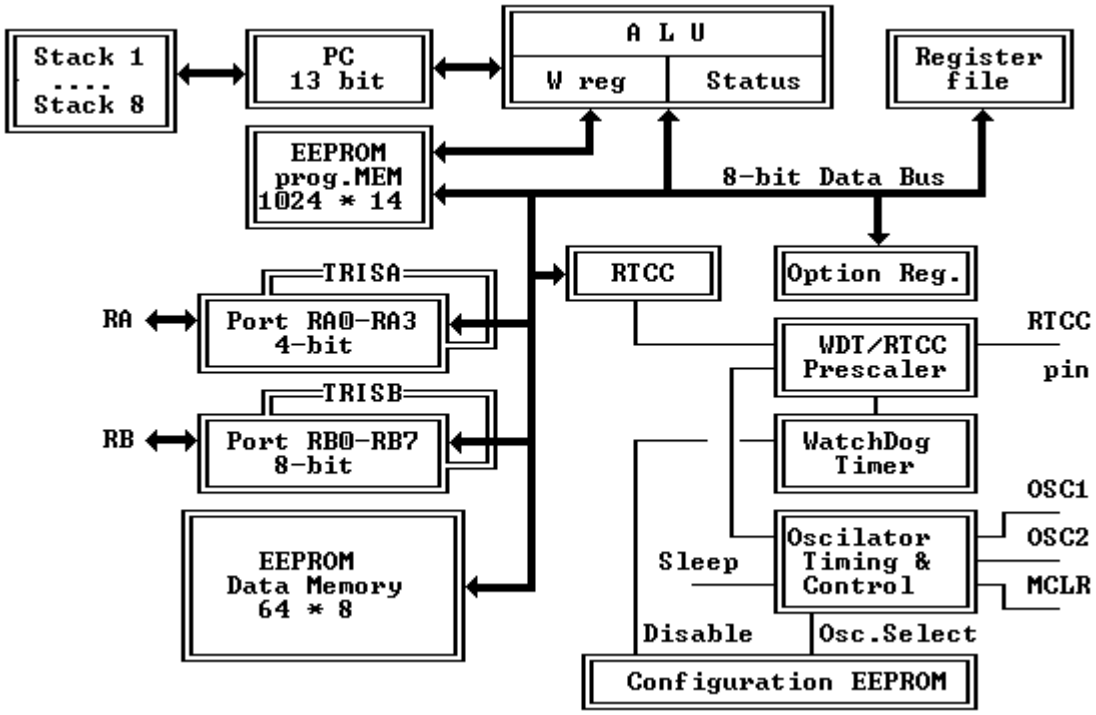

Рисунок 3.17 Структурная схема PIC16C84

Шина данных и память данных (ОЗУ) - имеют ширину 8 бит, а программная шина и программная память (ПЗУ) имеют ширину 14 бит. Все команды выполняются за один цикл, исключая команды переходов. Исполняемая программа может находиться только во встроенном ПЗУ.

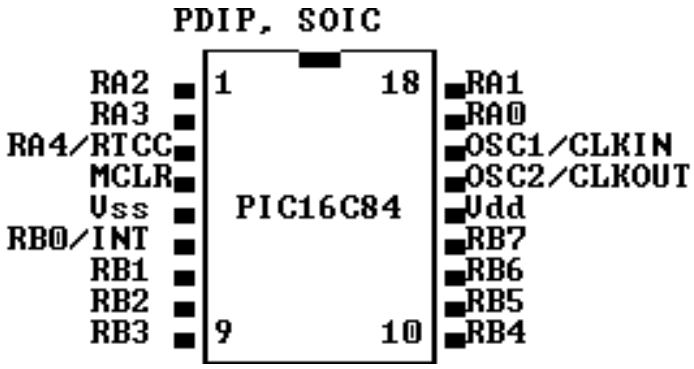

Рисунок 3.18. Обозначения ножек PIC16C84

Контроллер имеет только 35 простых команд с прямой, косвенной и относительной адресацией данных и команд, все команды выпол-

няются за один цикл(400ns), кроме команд перехода - 2 цикла. Рабочая частота 0 Гц- 10 МГц (min 400 нс цикл команды), PIC имеет:

- 36 х 8 регистров общего использования;
- 15 специальных аппаратных регистров SFR;
- 64 х 8 электрически перепрограммируемой EEPROM памяти для данных;
- восьмиуровневый аппаратный стек;
- четыре источника прерывания:
- внешний вход INT
- переполнение таймера RTCC
- прерывание при изменении сигналов на линиях порта B
- по завершению записи данных в память EEPROM
- 13 линий ввода-вывода с индивидуальной настройкой;
- 8 - битный таймер/счетчик RTCC с 8-битным программируемым предварительным делителем;
- встроенное устройство программирования EEPROM памяти программ и данных; используются только две ножки.

Таблица 3.12

Функциональное назначение ножек

Обозначение	Нормальный режим
RA0 - RA3	Двунаправленные линии ввода/ ввда. Входные уровни ТТЛ.
RA4/RTCC	Вход через триггер Шмитта. Ножка порта ввода/вывода с открытым стоком или вход частоты для таймера/счетчика RTCC.
RB0/INT	Двунаправленная линия порта ввода/ вывода или внешний вход прерывания. Уровни ТТЛ.
RB1 - RB5	Двунаправленные линии ввода/ вывода. Уровни ТТЛ.
RB6	Двунаправленные линии ввода/ вывода. Уровни ТТЛ.

RB7	Двунаправленные линии ввода/ вывода. Уровни ТТЛ.
MCLR/Vpp	Низкий уровень на этом входе генерирует сигнал сброса для контроллера. Активный низкий.
OSC1/CLKIN	Для подключения кварца, RC или вход внешней тактовой частоты.
OSC2/CLKOUT	Генератор, выход тактовой частоты в режиме RC генератора, в остальных случаях - для подключения кварца
Vdd	Напряжение питания
Vss	Общий (земля)

Область ОЗУ организована как 128 х 8. К ячейкам ОЗУ можно адресоваться прямо или косвенно, через регистр указатель FSR (04h). Это также относится и к EEPROM памяти данных-констант.

Таблица 3.13

Адреса специальных регистров

Страница 0		Страница 1	
00	Косвенный адрес	Косвенный адрес	80
01	RTCC	OPTION	81
02	PCL	PCL	82
03	STATUS	STATUS	83
04	FSR	FSR	84
05	PORT A	TRISA	85
06	POTR B	TRISB	86
07			87
08	EEDATA	EECON1	88
09	EEADR	EECON2	89
0A	PCLATH	PCLATH	8A
0B	INTCON	INTCON	8B
0C 2F	36 регистров общего пользования	36 регистров общего пользования	8C AF

30 7F	Не существует	Не существует	B0 FF

В регистре статуса (03h) есть биты выбора страниц, которые позволяют обращаться к четырем страницам будущих модификаций этого кристалла. Однако для PIC16C84 память данных существует только до адреса 02Fh. Первые 12 адресов используются для размещения регистров специального назначения. Регистры с адресами 0Ch-2Fh могут быть использованы, как регистры общего назначения, которые представляют собой статическое ОЗУ. Некоторые регистры специального назначения продублированы на обеих страницах, а некоторые расположены на странице 1 отдельно. Когда установлена страница 1, то обращение к адресам 8Ch-AFh фактически адресует страницу 0. К регистрам можно адресоваться прямо или косвенно. В обоих случаях можно адресовать до 512 регистров.

Прямая адресация. Когда производится прямая 9-битная адресация, младшие 7 бит берутся как прямой адрес из кода операции, а два бита указателя страниц (RP1,RP0) из регистра статуса (03h).

Косвенная адресация. f4 - Указатель косвенной адресации. Любая команда, которая использует f0 (адрес 00) в качестве регистра фактически обращается к указателю, который хранится в FSR (04h). Чтение косвенным образом самого регистра f0 даст результат 00h. Запись в регистр f0 косвенным образом будет выглядеть как Nop, но биты статуса могут быть изменены. Необходимый 9 - битный адрес формируется объединением содержимого 8 - битного FSR регистра и бита IRP из регистра статуса.

RTCC таймер/счетчик. Режим таймера выбирается путем сбрасывания в ноль бита RTS, который находится в регистре OPTION. В ре-

жиме таймера RTCC будет инкрементироваться каждый командный цикл (без предделителя).

Режим счетчика выбирается путем установки в единицу бита RTS, который находится в регистре OPTION. В этом режиме RTCC будет инкрементироваться либо положительным, либо отрицательным фронтом на ножке RA4/RTCC от ВНЕШНИХ событий. Пределитель может быть использован или совместно с RTCC, или с Watchdog таймером. Вариант подключения делителя контролируется битом PSA в регистре OPTION. При PSA=0 делитель будет подсоединен к RTCC. Содержимое делителя программе недоступно. Коэффициент деления - программируется.

Прерывание по RTCC вырабатывается тогда, когда происходит переполнение RTCC таймера/счетчика при переходе от FFH к 00h.

При переполнения счетчика устанавливается бит запроса RTIF в регистре INTCON<2>. Прерывание можно замаскировать битом RTIE в регистре INTCON<5>. Бит запроса RTIF должен быть сброшен программно при обработке прерывания. Прерывание по RTCC не может вывести процессор из SLEEP потому, что таймер не функционирует в этом режиме.

Регистр статуса. Регистр (f3) содержит арифметические флаги АЛУ, состояние контроллера при сбросе и биты выбора страниц для памяти данных.

(f3) доступен для любой команды так же, как любой другой регистр. Однако, биты T0 и PD устанавливаются аппаратно и не могут быть записаны в статус программно.

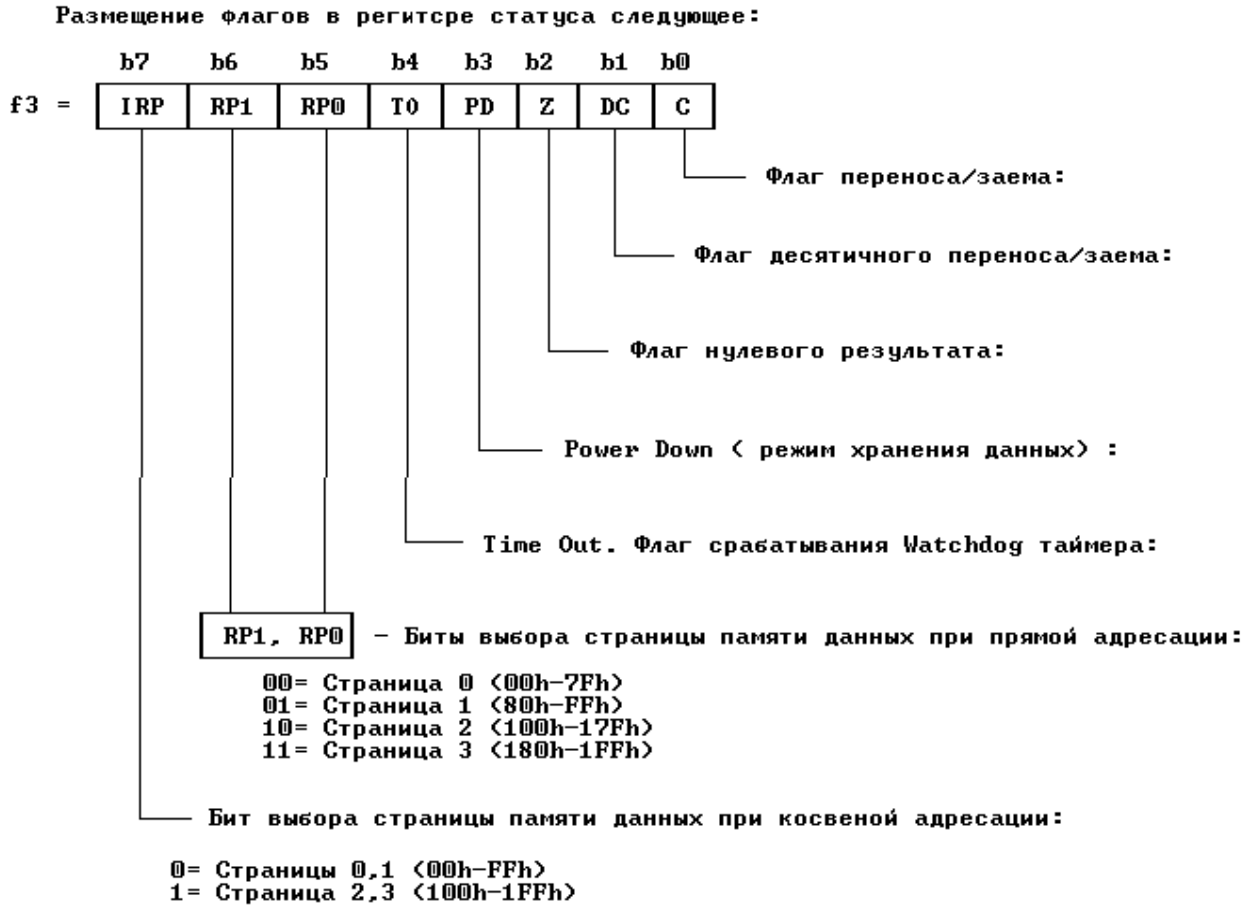

Рисунок 3.19 Формат слова-состояния.

Аппаратные Биты статуса TO (Time Out) и PD (Power Down). По состоянию битов регистра статуса "TO" и "PD" можно определить, чем был вызван "Сброс": просто включением питания, срабатыванием таймера watchdog, - выходом из режим а пониженного энергопотребления, (Sleep) в результате срабатывания watchdog таймера, по внешнему сигналу /MCLR.

Стек и возвраты из подпрограмм. PIC16C84 имеет восьмиуровневый аппаратный стек шириной 13 бит. Область стека не принадлежит ни к программной области ни к области данных, а указатель стека пользователю недоступен.

Долговременная память данных EEPROM. Память данных EEPROM позволяет прочитать и записать байт информации. При за-

писи байта автоматически стирается предыдущее значение и записываются новые данные (стирание перед записью). Все эти операции производит встроенный автомат записи EEPROM. Содержимое ячеек этой памяти сохраняется при выключении питания.

Кристалл PIC16C84 имеет память данных 64x8 EEPROM бит, которая позволяет запись и чтение во время нормальной работы (во всем диапазоне питающих напряжений). Эта память не принадлежит области регистров ОЗУ. Доступ к ней осуществляется через два регистра: EEDATA <08h>, который содержит в себе восьмибитовые данные для чтения/записи и EEADR <09h>, который содержит в себе адрес ячейки к которой идет обращение. Дополнительно имеется два управляющих регистра: EECON1 <88h> и EECON2 <89h>.

При считывании данных из памяти EEPROM необходимо записать требуемый адрес в EEADR регистр и затем установить бит RD EECON1<0> в еденицу. Данные появятся в следующем командном цикле в регистре EEDATA и могут быть прочитаны. Данные в регистре EEDATA защелкиваются.

При записи в память EEPROM, необходимо сначала записать требуемый адрес в EEADR регистр и данные в EEDATA регистр. Затем выполнить специальную последовательность команд, производящую непосредственную запись:

```
movlv   55h
movwf   EECON2
movlv   AAh
movwf   EECON2
bsf     EECON1,WR ;установить WR бит, начать запись
```

Во время выполнения этого участка программы, все прерывания должны быть запрещены для точного выполнения временной диаграммы. Время записи - примерно 10мс.

Управление EEPROM.

Таблица 3.14

Управляющие регистры для EEPROM

Название	**Функция**	**Адрес**	**Значение после включения**
EEDATA	EEPROM регистр данных	08h	XXXX XXXX
EEADR	EEPROM регистр адреса	09h	XXXX XXXX
EECON1	EEPROM 1 управляющий регистр	88h	0000 X000
EECON2	EEPROM 2 управляющий регистр	89h	-

Организация прерываний. Прерывания в PIC16C84 могут быть от четырех источников:

- внешнее прерывание с ножки RB0/INT,
- прерывание от переполнения счетчика/таймера RTCC,
- прерывание по окончании записи данных в EEPROM
- прерывание от изменения сигналов на ножках порта RB<7:4>.

Все прерывания имеют один и тот же вектор/адрес - 0004h. Однако, в управляющем регистре прерываний INTCON записывается от какого именно источника поступил запрос прерывания. Записывается соответствующим битом-флагом, имеющим адрес 0Bh. Значение при reset= 0000 000?.

Управляющий регистр прерываний и его биты

GIE	EEIE	RTIE	INTE	RBIE	RTIF	INTF	RBIF

RBIF - Флаг прерывания от изменения на порту RB.

1 - когда сигнал на входе RB<7:4> изменяется, сбрасывается программным способом.

INTF - Флаг прерывания INT.

1 - когда на ножке INT появляется сигнал от внешнего источник прерывания. Сбрасывается программным способом.

RTIF - Флаг прерывания от переполнения RTCC.

1 - когда RTCC переполняется сбрасывается программным способом.

RBIE - Бит разрешения/запрещения RBIF прерывания.

0 - запрещает RBIE прерывание

1 - разрешает RBIE прерывание

INTE - Бит разрешения/запрещения INT прерывания.

0 - запрещает INT прерывание

1 - разрешает INT прерывание

RTIE - Бит разрешения/запрещения RTIF прерывания.

0 - запрещает RTIE прерывание

1 - разрешает RTIE прерывание

EEIE - Бит разрешения/запрещения прерывания EEPROM записи.

0 - запрещает EEIF прерывание

1 - разрешает EEIF прерывание

GIE - Бит разрешения/запрещения всех прерываний.

0 - запрещает прерывания

1 - разрешает прерывания

Внешнее прерывание. Внешнее прерывание на ножке RB0/INT осуществляется по фронту: либо по нарастающему (если бит 6

INTEDG=1 в регистре OPTION), либо по спадающему фронту (если INTEDG=0). Бит запроса INTF должен быть очищен прерывающей программой перед тем, как опять разрешить это прерывание.

Прерывание от RTCC. Переполнение счетчика RTCC (FFh->00h) установит бит запроса RTIF (INTCON<2>). Это прерывание может быть разрешено/запрещено установкой/сбросом бита маски RTIE (INTCON<5>).

Прерывание от порта RB. Любое изменение сигналов на четырех входах порта RB<7:4> установит бит RBIF (INTCON<0>). Это прерывание может быть разрешено/запрещено установкой/сбросом бита маски RBIE (INTCON<3>).

Прерывание от EEPROM. Флаг запроса прерывания по окончании записи в EEPROM, EEIF (EECON1<4>) устанавливается по окончании автоматической записи данных в EEPROM. Это прерывание может быть замаскировано сбросом бита EEIE (INTCON<6>).

Обзор регистров/портов. Кристалл имеет два порта: 5 бит порт RA и 8 бит порт RB с побитовой индивидуальной настройкой на ввод или на вывод.

Порт A - это порт шириной 5 бит, соответствующие ножки кристалла RA<4:0>. Линии RA<3:0> двунаправленные, а линия RA4 -выход с открытым стоком. Адрес регистра порта A - 05h. Относящийся к порту A управляющий регистр TRISA расположен на первой странице регистров по адресу 85h. TRISA<4:0> - это регистр шириной 5 бит. Если бит управляющего TRISA регистра имеет значение единица, то соответствующая линия будет устанавливаться на ввод. Ноль переключает линию на вывод и одновременно выводит на нее содержимое соответствующего регистра защелки.

Порт B - это двунаправленный порт, шириной в восемь бит (адрес регистра 06h). Относящийся к порту B управляющий регистр TRISB расположен на первой странице регистров по адресу 86h. Если бит управляющего TRISB регистра имеет значение единица, то соответствующая линия будет устанавливаться на ввод. Ноль переключает линию на вывод и одновременно выводит на нее содержимое соответствующего регистра защелки. Четыре линии порта B (RB<7:4>)

имеют способность вызвать прерывание при изменении значения сигнала на любой из них.

Таблица 3.20

Назначение выводов порта B

RB0/INT	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	Вход внешнего прерывания
RB1	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	-
RB2	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	-
RB3	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	-
RB4	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	Прерывание при изменении
RB5	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	Прерывание при изменении
RB6	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	Прерывание при изменении
RB7	Порт ввода/вывода. Входные уровни ТТЛ и внутренняя программируемая активная нагрузка.	Прерывание при изменении

Последовательное обращение к портам ввода/вывода. Запись в порт вывода происходит в конце командного цикла. Но при чтении,

данные должны быть стабильны в начале командного цикла. Будьте внимательны в операциях чтения, следующих сразу за записью в тот же порт. Здесь надо учитывать инерционность установления напряжения на выводах. Может потребоваться программная задержка, чтобы напряжение на ножке (зависит от нагрузки) успело стабилизироваться до начала исполнения следующей команды чтения.

Система команд и обозначения. Каждая команда PIC16C84 - это 14-битовое слово, которое разделено по смыслу на следующие части:

- 1. код операции, -2. поле для одного и более операндов, которые могут участвовать или нет в этой команде. Система команд PIC16C84 включает в себя байт-ориентированные команды, бит-ориентированные, операции с константами и команды передачи управления.

Все команды выполняются в течение одного командного цикла. В двух случаях исполнение команды занимает два командных цикла: -1. проверка условия и переход, -2.изменение программного счетчика как результат выполнения команды. Один командный цикл состоит из четырех периодов генератора. Таким образом, для генератора с частотой 4 МГц время исполнения командного цикла будет 1 мкс. Система команд приведена в приложении 4.

Регистр OPTION (адрес 81h) доступен для чтения и записи и содержит различные управляющие биты, которые определяют конфигурацию пределителя, куда он подключен: к RTCC или WDT, знак фронта внешнего прерывания INT и внешнего сигнала для RTCC, подключение активной нагрузки на порту RB.

Регистр OPTION - адрес 81h. При включении питания равен = FFH.

RBPU	INTEDG	RTS	RTE	PSA	PS2	PS1	PS0

PSA - Бит, подключающий делитель к: 0 - RTCC

RTE - Фронт внешнего сигнала RTCC:

0 - инкремент по положительному фронту на ножке RTCC

1 - инкремент по отрицательному фронту на ножке RTCC

RTS| - Источник сигнала для RTCC

0 - сигнал от внутреннего генератора

1 - Внешний сигнал на ножке RTCC

INTEDG - Фронт сигнала INT:

0 - прерывание по отрицательному фронту на ножке INT

1 - прерывание по положительному фронту на ножке INT

RBPU - Инверсный бит подключения активной нагрузки к порту B.

0 - Активные нагрузки будут подключаться по алгоритму работы порта RB

1 - Активные нагрузки порта B отключены всегда

Подключения делителя частоты. Один и тот же восьмибитный счетчик может быть включен либо перед RTCC либо после Watchdog таймера. Отметим, что делитель работает только с одним из этих устройств.

Для настройки предделителя необходимо установить коэффициенты деления в соответствие с таблицей.

Таблица 3.21

Коэффициенты деление предделителя

PS2...PS0	RTCC	WDR
0 0 0	1 : 2	1 : 1
0 0 1	1 : 4	1 : 2
0 1 0	1 : 8	1 : 4
0 1 1	1 : 16	1 : 8
1 0 0	1 : 32	1 : 16

1 0 1	1 : 64	1 : 32
1 1 0	1 : 128	1 : 64
1 1 1	1 : 256	1 : 128

4. ОРГАНИЗАЦИЯ ВВОДА-ВЫВОДА

4.1. Датчики, общие сведения, терминология, параметры

Технические системы, содержащие микропроцессор, практически бесполезны без сигналов об окружающей среде, т. е. без датчиков, которые во многом определяют уровень системы, поэтому при построении микропроцессорных устройств одним из важных вопросов является правильный выбор датчиков и устройств согласования, которые осуществляют преобразование измеряемой величины в электрический сигнал и последующий ввод в МПУ.

Датчики во многом определяют основные параметры проектируемой системы - быстродействие, чувствительность, стоимость. В любом датчике можно выделить две основные составляющие: чувствительный элемент и преобразователь (рисунок 4.1) [18].

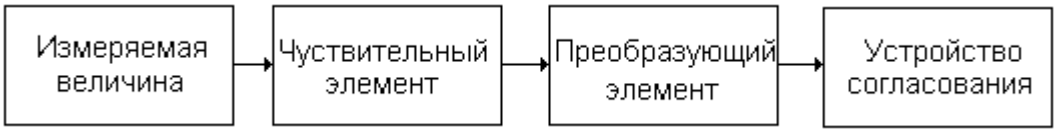

Рисунок 4.1 Структурная схема датчика.

Чувствительный элемент воспринимает измеряемое свойство объекта и преобразует его в другую физическую величину. Затем преобразующий элемент преобразует эту физическую величину в электрический сигнал, значение которого отражает уровень измеряе-

мого свойства объекта. Показанные в пунктирных линиях элементы могут в некоторых преобразователях отсутствовать.

Для характеристики датчиков применяют ряд общих параметров не зависимости от классификации. Наиболее важные из них: погрешность, чувствительность, разрешающая способность, время отклика, полоса частот.

Погрешность. Точность измерений, характеризующая близость измеренного значения физической величины к его действительному значению, обычно оценивается **погрешностью,** т.е. максимально возможной разностью между измеренным и действительным значениями. Погрешность может измеряться в абсолютных значениях измеряемой величины, в процентах к измеряемой величине или к полной шкале, в единицах значащих разрядов измерительных приборов.

Разрешающая способность системы, характеризует наибольшую точность, с которой осуществляются измерения. Важно помнить о том, что хотя разрешающая способность может быть меньше, чем присущая преобразователю погрешность, это вовсе не означает, что отсчет имеет малую погрешность. Общая погрешность будет, безусловно, больше.

Чувствительность (масштабный коэффициент) - есть отношение изменения его выходного сигнала к изменению на входе.

Время отклика - равно времени установления выходного сигнала в ответ на изменение измеряемой величины. Оно показывает, через какой промежуток времени выходной сигнал преобразователя отреагирует на изменение измеряемой величины. Присущая преобразователю инертность означает, что его нельзя использовать для измерения входной величины с быстроизменяющейся флуктуацией.

Полоса частот — это характеристика, связанная с временем от-

клика. Она характеризует диапазон частот изменений измеряемой величины.

По характеру зависимости выходного сигнала от измеряемой величины датчики делятся на линейные и нелинейные.

Линейные – значение выходного сигнала линейно связано с изменением измеряемой величины. Использование преобразователей с линейной характеристикой позволяет существенно упростить аппаратную часть системы и программу, но диапазон измеряемых величин у них уже чем у нелинейных.

Нелинейные - значение выходного сигнала нелинейно связано с изменением измеряемой величины. Нелинейность характеризуется порядком, который определяется типом описывающего её уравнения. При использовании нелинейных преобразователей необходимо выполнять линеаризацию либо аппаратным, либо программным путем.

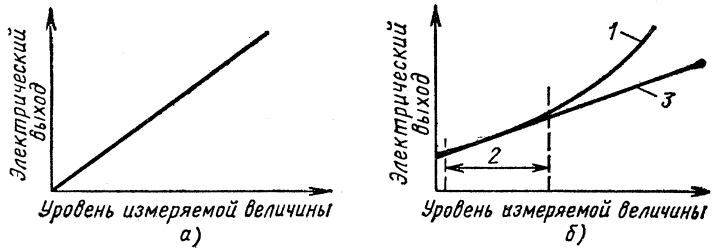

Рисунок 4.2 Линейные (а) и нелинейные (б) характеристики датчиков.

Аппаратные методы предполагают использование функциональных преобразователей, однако, метод применим лишь в несложных случаях, т. к. аппаратная линеаризация сложных функций с достаточной точностью является сложной и дорогостоящей задачей. В программном методе линеаризация снижает быстродействие системы, но не требует дополнительных аппаратных затрат.

Датчики информируют о состоянии внешней среды путем взаимодействия с ней и преобразования реакции на это взаимодействие в

электрические сигналы. Существует множество явлений и эффектов, видов преобразования свойств и энергии, которые можно использовать для создания датчиков (таблица 4.1) [8].

Таблица 4.1

Физические явления и преобразования энергии на их основе

Эффект, явление, свойство	Физическая сущность преобразования
Теплопроводность (тепловая энергия → изменение физических свойств)	Переход теплоты внутри физического объекта из области с более высокой в область с более низкой температурой
Тепловое излучение (тепловая энергия → инфракрасные лучи)	Оптическое излучение при повышении температуры физического объекта
Эффект Зеебека (температура → электричество)	Возникновение ЭДС в цепи с биметаллическими соединениями при разной температуре спаев
Пироэлектрический эффект (температура → электричество)	Возникновение электрических зарядов на гранях некоторых кристаллов при повышении температуры
Термоэлектронный эффект (тепловая энергия → электроны)	Испускание электронов при нагревании металла в вакууме
Электротермический эффект Пельтье (электричество → тепловая энергия)	Поглощение или генерация тепловой энергии при электрическом токе в цепи с биметаллическими соединениями
Электротермический эффект Томсона (температура и электричество → тепловая энергия)	Генерация или поглощение тепловой энергии в электрической цепи из однородного материала при разных температурах участков цепи
Фотогальванический эффект (свет → электричество)	Появление свободных электронов и положительных дырок (возникновение ЭДС) в облучаемом светом *p*-n переходе
Эффект фотопроводимости (свет → электрическое сопротивление)	Изменение электрического сопротивления полупроводника при его облучении светом
Эффект Зеемана (свет, магнетизм → спектр)	Расщепление спектральных линий при прохождении света в магнитном поле

Эффект Рамана или комбинационное рассеяние света (свет → свет)	Возникновение в веществе светового излучения, отличного по спектру от исходного монохроматического
Эффект Поккельса (свет и электричество → свет)	Расщепление светового луча на обыкновенный и необыкновенный при прохождении через пьезокристалл с приложенным к нему электрическим напряжением в перпендикулярном лучу направлении
Эффект Керра (свет и электричество → свет)	Расщепление светового луча на обыкновенный и необыкновенный в изотропном веществе с приложенным к нему электрическим напряжением в перпендикулярном к лучу направлении
Эффект Фарадея (свет и магнетизм → свет)	Поворот плоскости поляризации линейно-поляризованного светового луча, проходящего через парамагнитное вещество
Эффект Холла (магнетизм и электричество → электричество)	Возникновение разности потенциалов на гранях твердого тела при пропускании через него электрического тока и приложении магнитного поля перпендикулярно направлению электрического тока
Магнитосопротивление (магнетизм и электричество → электрическое сопротивление)	Увеличение электрического сопротивления твердого тела в магнитном поле
Магнитострикция (магнетизм → деформация)	Деформация ферромагнитного тела, помещенного в магнитное поле
Пьезоэлектрический эффект (давление → электричество)	Возникновение разности потенциалов на гранях сегнетоэлектрика, находящегося под давлением

Эффект Доплера (звук, свет → частота)	Изменение частоты при взаимном перемещении объектов по сравнению с частотой, когда эти объекты неподвижны

По типу реакции на воздействие датчики можно разделить на датчики бинарного типа и аналоговые.

Датчики бинарного типа обнаруживают только переход контролируемой физической величиной определенного порогового значения. Типичными представителями подобных датчиков являются биметаллические переключатели и датчики положения на основе концевых выключателей. Такие датчики называются бинарными, так как их выходной сигнал имеет только одно из двух состояний: «включено» или «выключено».

Бинарные датчики по принципу действия делятся на контактные и бесконтактные. Первые содержат электромеханический контакт. Выходные сигналы контактных бинарных датчиков легко вводятся в микро-ЭВМ. Общая схема включения контактных датчиков приведена на рисунке 4.3.

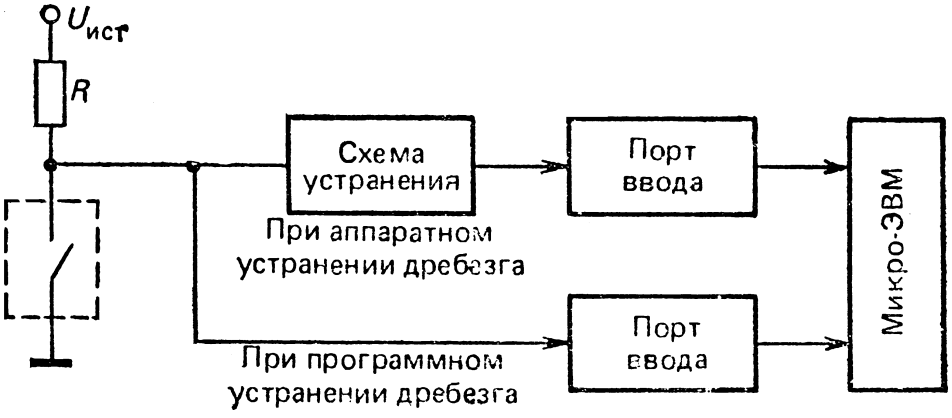

Рисунок 4.3. Схема включения контактных датчиков.

При замыкании контакта в таких датчиках возникает «дребезг» (многократность контактирования), для устранения нежелательных последствий этого явления необходимо предусмотреть соответствующие аппаратные или программные средства [19].

В качестве аппаратных средств устранения дребезга можно применить триггер к одному из входов, которого подключён контактный датчик. Программное устранение дребезга заключается в формировании задержки считывания датчика относительно первого срабатывания контакта. Время задержки определяется конструктивными особенностями датчика.

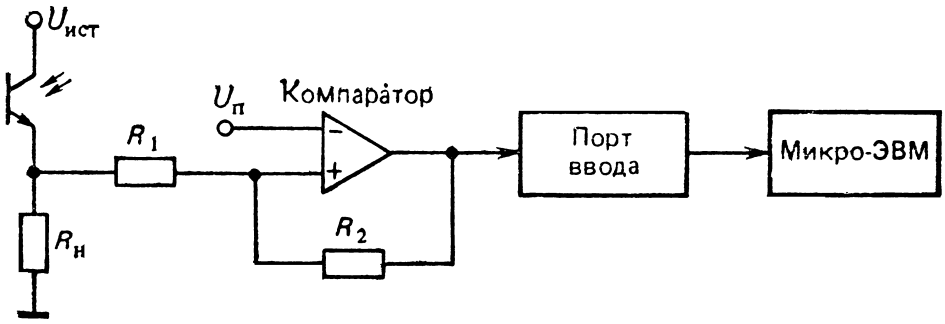

Рисунок 4.4 Схема включения бесконтактного датчика.

К бесконтактному типу бинарных датчиков относятся, например, датчики, выполненные на основе оптического прерывателя или элемента Холла, различного типа магнитные, емкостные и т.д. В датчиках этого типа состояние «включено» или «выключено» на выходе отражается в виде изменения электрического сигнала, имеющего скорее аналоговый, чем чисто цифровой характер. Обычно для улучшения качества выходного сигнала бесконтактного датчика используется компаратор, как это показано на рисунке 4.4. Компаратор сравнивает выходной сигнал датчика с некоторым пороговым уровнем $U_п$ и на основании этого оценивает состояние датчика—«включено» или «выключено». Когда выходной сигнал датчика близок к уровню сравне-

ния, то под влиянием шумовых составляющих может начаться многократное срабатывание компаратора. Для устранения этого недостатка схема компаратора должна обладать некоторым гистерезисом, обеспечивающим необходимую зону нечувствительности. При этом следует учитывать, что с увеличением гистерезиса стабильность срабатывания улучшается, но зато снижается точность обнаружения (Ширина петли гистерезиса определяется соотношением R_1 и R_2)

Аналоговые датчики получают информацию в некотором непрерывном интервале значений физической величины.

По виду изменяемого выходного электрического параметра аналоговые датчики делятся на три группы: с изменяемым выходным напряжением, током или сопротивлением рисунке 4.5.

Самый распространенный способ — с использованием АЦП. Выходной сигнал датчика после первичной обработки превращается в аналоговое напряжение оптимального уровня, а затем с помощью АЦП преобразуется в цифровой сигнал.

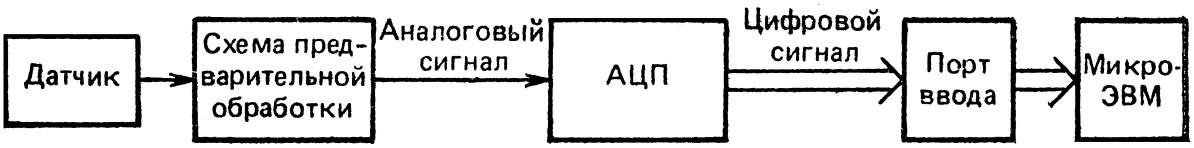

Рисунок 4.5 Схема включения аналогового датчика.

Схема предварительной обработки преобразует выходной сигнал датчика в напряжение значение, которого лежит в рабочем диапазоне входных напряжений используемого АЦП.

В тех случаях, когда датчики и микро-ЭВМ расположены на значительном расстоянии друг от друга, удобен способ соединения, при котором аналоговое напряжение преобразуется с помощью преобразователя «напряжение → частота» в соответствующее изменение частоты несущей или импульсов.

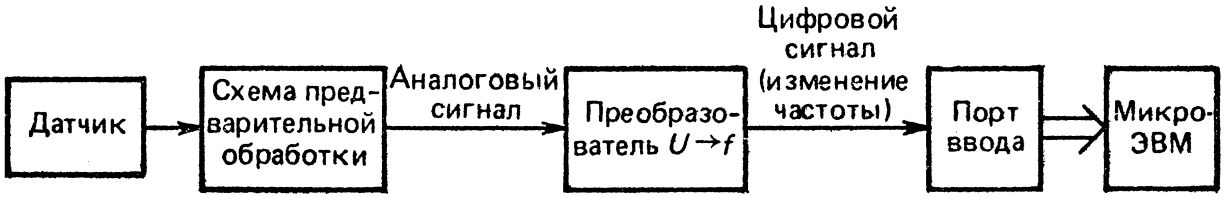

Рисунок 4.6 Схема включения аналогового датчика с преобразователем.

Схема позволяет уменьшить число сигнальных жил в соединительном кабеле между датчиками и микро-ЭВМ, повысить помехоустойчивость системы.

Аналого-цифровые преобразователи. Описание принципов работы различных АЦП приведены в [20]. Правильный выбор АЦП очень важен и отражается на характеристиках измерительной системы и её стоимости. При выборе того или иного типа преобразователя необходимо исходить из цели применения с учетом следующих наиболее важных характеристик этих устройств.

Разрешающая способность. Под разрешающей способностью обычно понимается минимальное значение аналогового сигнала, которое еще может различаться преобразователем. Разрешающая способность n -разрядного АЦП равна частному от деления на 2^n диапазона входного аналогового напряжения. В некоторых случаях разрешающая способность определяется в процентах диапазона входного напряжения.

С достаточной для практики точностью разрядность АЦП (n) можно определить из приближённой формулы

$$D[дБ]=6[дБ]*n$$

где - D динамический диапазон входных сигналов в децибелах.

Точность. В процессе квантования входного сигнала по уровню происходит округление его до ближайшего цифрового значения в пределах самого младшего разряда цифрового кода, т. е. возникает погрешность квантования, которая находится в пределах ±0, 5 значения самого младшего разряда. В реальных АЦП кроме этой погреш-

ности существуют и другие, которые необходимо учитывать при построении высокоточных измерительных систем [20,21].

Время преобразования – это время необходимое для преобразования одного отсчёта входного сигнала в цифровой код. Временной интервал, необходимый для осуществления правильного преобразования (временная апертура), зависит от скорости изменения входного сигнала и заданной разрешающей способности преобразования. Если изменение входного сигнала во время квантования превысит значение одного бита, то выходной код АЦП уже не будет соответствовать истинному значению входного сигнала, что приводит к дополнительной погрешности. Для устранения её необходимо применить устройство выборки – хранения (УВХ) рисунке 4.7.

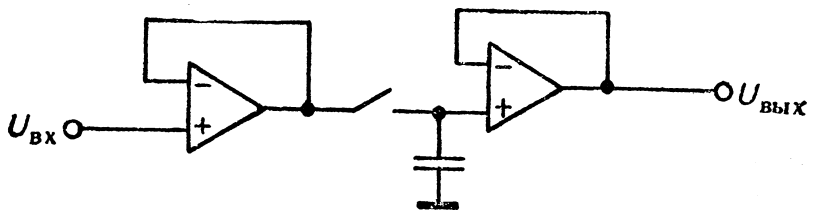

Рисунок 4.7 Схема устройства выборки-хранения.

УВХ устанавливается на входе АЦП, при кратковременном замыкании ключа конденсатор заряжается до значения входного напряжения. После размыкания ключа конденсатор оказывается подключен к высокоомному входу ОУ и таким образом $U_{вых}$ сохраняется неизменным в течении всего преобразования отсчёта.

Чтобы сохранить информацию об исходном сигнале, желательно выборку производить как можно чаще. Однако необоснованное увеличение частоты выборки приводит к большим затратам памяти для хранения отсчётов и повышению требований к быстродействию МП, что сказывается на стоимости измерительной системы. Для того, чтобы оцифрованный сигнал можно было восстановить без искажений

минимальная частота выборки должна не менее чем в два раза превышать максимальную частоту в спектре входного сигнала.

Для снижения шумовых высокочастотных составляющих, перед схемой выборки-запоминания необходимо установить фильтр нижних частот, подавляющий шумы.

Аналоговый мультиплексор. Стоимость АЦП с хорошими параметрами обычно довольно велика, поэтому для преобразования в цифровые коды сразу нескольких аналоговых сигналов удобнее использовать единственный преобразователь, но в совокупности с аналоговым мультиплексором (рисунок 4.8). Понятно, что благодаря применению мультиплексора одновременно можно ограничиться и меньшим числом портов ввода микроЭВМ.

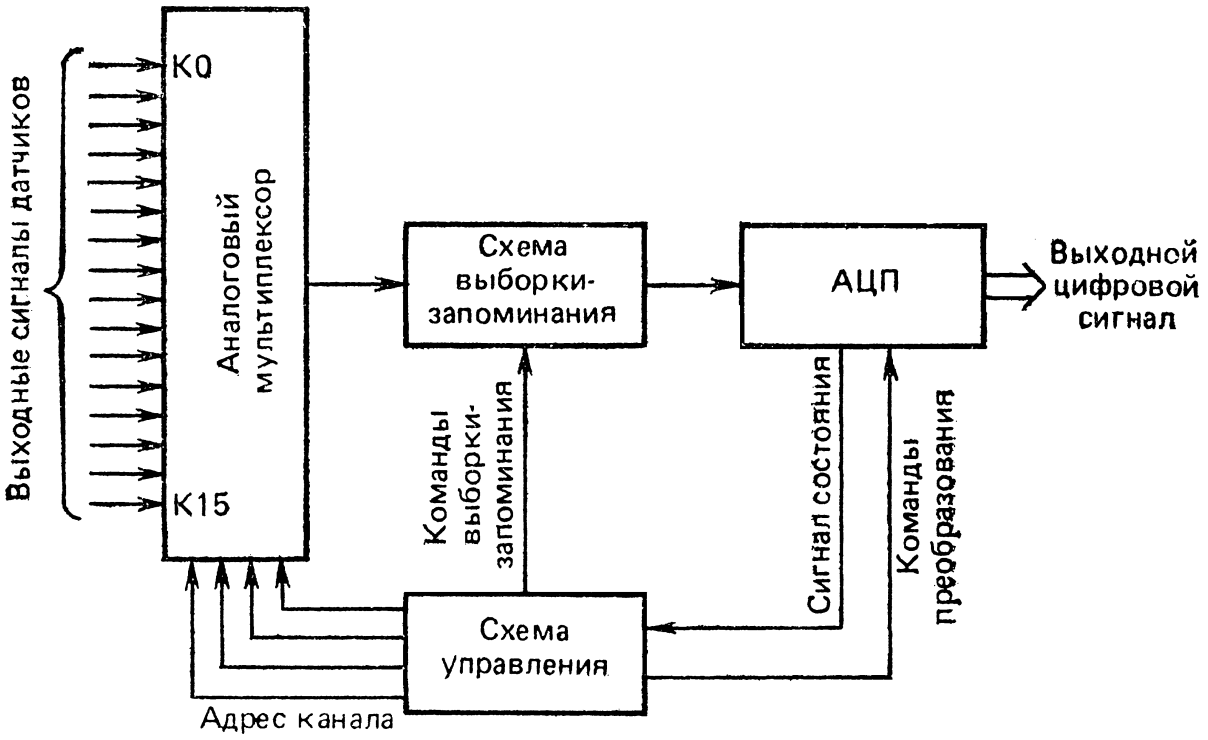

Рисунок 4.8. Включение аналоговых датчиков через мультиплексор.

Мультиплексор состоит из аналоговых переключателей, каждый из которых может присоединять свой вход к общему для всех переключателей

выходу. Выбор того или иного входного канала для подсоединения к выходу производится включением соответствующего аналогового переключателя путем выдачи бинарного кода с его адресом. В качестве аналоговых переключателей чаще всего используются полевые МОП- транзисторы, и если последующие каскады схемы переключателя имеют высокое входное сопротивление, то эти транзисторы присоединяются к ним непосредственно. Управление мультиплексором и УВХ легко осуществить программным путём, выделив для этого один портов МПУ.

4.2. Устройства сопряжения датчиков с МП

Данные, полученные в результате преобразования в АЦП, необходимо ввести в микро-ЭВМ. Алгорим управления ввода данных заключается в следующем: выдаётся имульс записи на УВХ, формируется импульс запуска для начала преобразования, после его окончания полученные данные считываются в память микро-ЭВМ.

Выходы АЦП подключается к МПУ непосредственно, если имеются свободные входы, либо через специально выделенный порт, например с помощью БИС Кр580ВВ55 [17].

Существуют различные способы получения результатов преобразования после того, как подана от микро-ЭВМ в АЦП стартовая команда и преобразование завершено.

Метод голосования. Микро-ЭВМ после выдачи в АЦП стартовой команды работает по специальной программе, в соответствии с которой производится опрос выхода готовности преобразователя. Сразу по завершении преобразования результирующие данные считываются и микро-ЭВМ может приступать к их обработке. Обычно для опроса входа готовности требуется отдельный входной порт.

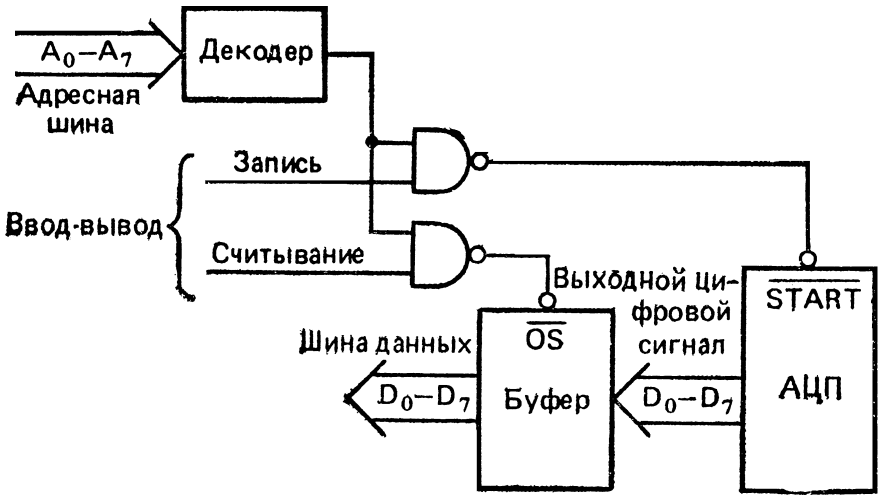

Рисунок 4.9 Схема сопряжения АЦП с микро-ЭВМ.

Метод прерываний. После запуска АЦП микро-ЭВМ продолжает работать по своей программе. Сигнал готовности АЦП подаётся на вход запроса прерывания МП. Как только преобразование заканчивается, микро-ЭВМ в соответствии с сигналом от АЦП временно прерывает выполнение текущей программы и производит считывание данных преобразования. Этот метод считывания особенно приемлем при использовании АЦП с интегрированием, в которых продолжительность преобразования сравнительно велика и зависит от значения входного сигнала.

Метод прямого доступа к памяти. Данные из АЦП с помощью контроллера прямого доступа к памяти, минуя центральный процессор, переносятся непосредственно в память ЭВМ. Этот способ удобно использовать при большом объеме данных, получаемых от быстродействующих АЦП.

5. ПРИЛОЖЕНИЯ

5.1. Приложение 1. Система команд Кр580ВМ80

Таблица 5.1

Условные обозначения

Обозначение	Содержание	
< Bi >	Содержание i-го байта команды	
Программно доступные регистры МП		
r	регист	код регистра
	B	000
	C	001
	D	010
	E	011
	H	100
	L	101
	M	110 (ячейка памяти)
	A	111 (аккумулятор)
Регистр признаков		
F	**Tc**	Признак переноса из старшего разряда
	Tz	Признак нулевого результата
	Ts	Признак равенства старшего байта = 1
	Tp	Признак четности числа единиц в результате
	Tv	Признак переноса из младшей тетрады в старшую
(r)	Операция выполняется над содержимым регистра r	
[(r)]	Содержимое ячейки памяти с адресом в паре регистров (r, r)	
$\wedge$	Логическое И	
$\oplus$	Исключающее ИЛИ	
$\vee$	Логическое ИЛИ	
(rm)	Операция выполняется над содержимым байта m регистра r	

Sp	Указатель вершины стека
Pc	Счетчик команд
←	Направление перемещения информации

Таблица 5.2

Система команд МП Кр580ВМ80

Код команды	Б	Ц	Т	Выполняемая операция		Мнемоника	Признаки Z	S	C	V	P
1. Команды пересылки и загрузки											
01 ri rj	1	1/3	5/7	Регистр - регистр ri ← (rj)		MOV ri,rj	-	-	-	-	-
00ri1100	2	2/3	7/10	Непосредственная загрузка ri ← < B2 >		MVI ri,B2	-	-	-	-	-
00ri001	3	3	10	Загрузка пары регистров ri ← <B3>, ri+1 ← <B2> при ri = 100 загрузка Sp		LXI r,	-	-	-	-	-
00 К1 010	1	2	7	Запоминание / Загрузка							
				К1	Операция		-	-	-	-	-
				000	[(BC)] ← (A)	STAX B	-	-	-	-	-
				010	[(DE)] ← (A)	STAX D	-	-	-	-	-
				001	A ← [(BC)]	LDAX B	-	-	-	-	-
				011	A ← [(DE)]	LDAX D	-	-	-	-	-
				110	[<B3> <B2>] ← (A)	STA	-	-	-	-	-
				111	A ← [<B3> <B2>]	LDA	-	-	-	-	-
				100	[<B3> <B2>] ← (L) [<B3> <B2>] ← (H)	SHLD	-	-	-	-	-
				101	L ← [<B3> <B2>] H ← [<B3> <B2>]	LHLD	-	-	-	-	-
11 ri 101	1	3	11	Загрузка стека. Sp ← (Sp) - 2 [Sp - 1] ← (ri), [Sp - 2] ← (ri + 1)		PUSH ri	-	-	-	-	-
11110101	1	3	11	Загрузка в стек AF Sp←(Sp) - 2 [Sp - 1] ← (A), [Sp - 2] ← (F)		PUSH PSW					
11 ri 001	1	3	10	Чтение стека Sp ← (Sp) + 2 ri + 1← [Sp - 1] ,ri ← [Sp - 1]		POP ri	-	-	-	-	-
11110001	1	3	10	Чтение из стека AF Sp←(Sp)+2		POP PSW	+	+	+	+	+

				F← [Sp], A← [Sp +1]						
11101011	1	1	4	Обмен между парами HL и DE (H) ↔ (D), (L) ↔ (E)	XCHG	-	-	-	-	-
11100011	1	5	13	Обмен вершины стека с HL [Sp] ↔ (L), [Sp + 1] ↔ (H)	XTHL	-	-	-	-	-
11111001	1	1	5	Пересылка HL в указатель Sp Sp ← (HL)	SPHL	-	-	-	-	-
11101001	1	1	5	Пересылка HL в счетчик Pc Pc ← (HL)	PCHL	-	-	-	-	-
2. Команды инкремента и декремента										
00 ri 100	1	1/3	5/6	Приращение одного регистра ri ← (ri) + 1	INR ri	+	+	-	+	+
00 ri 101	1	1/3	5/10	Уменьшение одного регистра ri ← (ri) - 1	DCR ri	+	+	-	+	+
00 ri 011	1	1	5	Приращение пары регистров ri, ri+1 ← (ri, ri+1) + 1 при ri = 110 приращение Sp	INX ri	-	-	-	-	-
01 ri 011	1	1	5	Уменьшение пары регистров ri, ri+1 ← (ri, ri+1) - 1 при ri = 111 уменьшение Sp	DCX ri	-	-	-	-	-
3. Арифметические и логические операции										
10 K2 ri	1	1/2	4/7	Сложение содержимого (A),(ri)						
				K2	Операция					
				000	A ← (A) + (ri)	ADD ri	+	+	+	+	+
				001	A ← (A) + (ri) + (Tc)	ADC ri	+	+	+	+	+
				010	A ← (A) - (ri)	SUB ri	+	+	+	+	+
				011	A ← (A) - (ri) - (Tc)	SBB ri	+	+	+	+	+
				100	A ← (A) ∧ (ri)	ANA ri	+	+	0	0	+
				101	A ← (A) ⊗ (ri)	XRA ri	+	+	0	0	+
				110	A ← (A) ∨ (ri)	ORA ri	+	+	0	0	+
				111	(A) - (ri), сравнение	CMP ri	+	+	+	+	+

00 ri 001	1	3	10	Сложение пар регистров HL ← (HL) + (ri-1, ri) при ri = 111 HL ← (HL) + (Sp)		DAD ri-1	-	-	+	-	-
11 K2 110	2	2	7	Сложение (A) с константой							
				K2	Операция						
				000	A ← (A) + (B2)	ADI ri	+	+	+	+	+
				001	A ← (A) + (B2) + (Tc)	ACI ri	+	+	+	+	+
				010	A ← (A) - (B2)	SUI ri	+	+	+	+	+
				011	A ← (A) - (B2) - (Tc)	SBI ri	+	+	+	+	+
				100	A ← (A) ∧ (B2)	ANI ri	+	+	0	0	+
				101	A ← (A) ⊗ (B2)	XRI ri	+	+	0	0	+
				110	A ← (A) ∨ (B2)	ORI ri	+	+	0	0	+
				111	(A) - (B2), сравнение	CMI ri	+	+	+	+	+
4. Операции циклического сдвига											
00 K3 111	1	1	4	K3	Операция						
				000	Сдвиг влево без переноса	RLC	-	-	+	-	-
				001	Сдвиг вправо без переноса	RRC	-	-	+	-	-
				010	Сдвиг влево с переносом	RAL	-	-	+	-	-
				011	Сдвиг вправо с переносом	RAR	-	-	+	-	-
5. Операции передачи управления											
11000011	3	3	10	Безусловный переход Pc ← <B3> <B2>		JMP	-	-	-	-	-
				Условные переходы							
11 K4 010	3	3	10	K4	Вид перехода						
				000	Если Tz=0, то Pc←<B3,B2> иначе Pc ← (Pc) + 3	JNZ	-	-	-	-	-
				001	Если Tz = 1,	JZ	-	-	-	-	-
				010	Если Tc = 0,	JNC	-	-	-	-	-
				011	Если Tc = 1,	JC	-	-	-	-	-
				100	Если Tp = 0,	JPO	-	-	-	-	-
				101	Если Tp = 1,	JPE	-	-	-	-	-
				110	Если Ts = 0,	JP	-	-	-	-	-

				111	Если Tz = 1,	JM	-	-	-	-	-
11001101	3	5	17	Вызов п/п безусловный [Sp-1], [Sp-2] ← (Pc), Sp ← (Sp)-2, Pc ← <B3> <B2>		CALL	-	-	-	-	-
				Вызов п/п условный							
11 K5 100	3	3/5	11/17	K5	Вид перехода						
				000	Если Tz=0, [Sp - 1][Sp - 2] ← (Pc), Sp←(Sp) - 2, Pc ← <B3,B2> иначе Pc ← (Pc) + 3	CNZ	-	-	-	-	-
				001	Если Tz = 1,	CZ	-	-	-	-	-
				010	Если Tc = 0,	CNC	-	-	-	-	-
				011	Если Tc = 1,	CC	-	-	-	-	-
				100	Если Tp = 0,	CPO	-	-	-	-	-
				101	Если Tp = 1,	CPE	-	-	-	-	-
				110	Если Ts = 0,	CP	-	-	-	-	-
				111	Если Tz = 1,	CM	-	-	-	-	-
11001001	1	3	10	Возврат из п/п безуслов-ный Pc ← [Sp][Sp+1], Sp ← (Sp) + 2		RET	-	-	-	-	-
				Возврат из п/п условный							
11 K6 000	1	1/3	5/11	K6	Вид перехода						
				000	Если Tz = 0 , Pc←[Sp][Sp+2],Pc ← (Pc)+1 иначе Pc ← (Pc) + 1	RNZ	-	-	-	-	-
				001	Если Tz = 1,	RZ	-	-	-	-	-
				010	Если Tc = 0,	RNC	-	-	-	-	-
				011	Если Tc = 1,	RC	-	-	-	-	-
				100	Если Tp = 0,	RPO	-	-	-	-	-
				101	Если Tp = 1,	RPE	-	-	-	-	-
				110	Если Ts = 0,	RP	-	-	-	-	-
				111	Если Tz = 1,	RM	-	-	-	-	-
6. Операции ввода и вывода											

11011011	2	3	10	6.1 Ввод данных из УВВ № A ← (ШД), ША ← № УВВ	IN №	-	-	-	-	-
11010011	2	3	10	6.2 Вывод данных в УВВ № ШД ← (A), ША ← № УВВ	OUT №	-	-	-	-	-
7. Прочие операции										
11AAA111	2	3	11	7.1 Рестарт. [Sp-1][Sp-2] ← (Pc) Sp←(Sp) - 2, Pc ← 0000000.000**AAA**000	RST n	-	-	-	-	-
00101111	1	1	4	7.2 Инверсия A. A ← ($\overline{A}$)	SMA	-	-	-	-	-
00101111	1	1	4	7.3 Установка Tc. Tc ← 1	STC	-	-	1	-	-
00111111	1	1	4	7.4 Инверсия Tc. Tc ← ($\overline{T}$c)	CMC	-	-	+	-	-
11111011	1	1	4	7.5 Разрешение прерываний	EI	-	-	-	-	-
11110011	1	1	4	7.6 Запрещение прерываний	DI	-	-	-	-	-
00100111	1	1	4	7.7 Двоично-десятичная коррекция A	DAA	+	+	+	+	+
00000000	1	1	4	7.8 Отсутствие операции	NOP	-	-	-	-	-
01110110	1	1	7	7.9 Останов	HLT	-	-	-	-	-

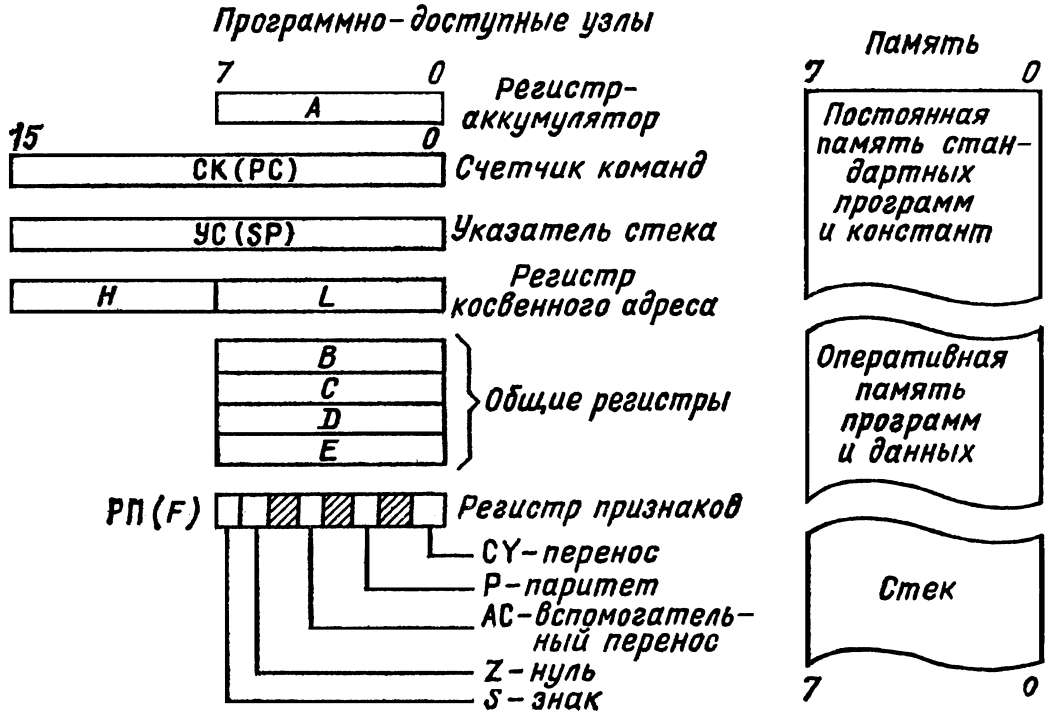

Рисунок 5.1 Программная модель МП Кр580ВМ80

5.2. Приложение 2. Система команд ОМЭВМ семейства МК048

Таблица 5.3

Условные обозначенья

R	Регистр общего назначения в выбранном банке регистров
@R	Регистр общего назначения в выбранном банке регистров, используемый в качестве регистра косвенного адреса
#D	Непосредственные данные
A	Аккумулятор
PSW	Слово-состояние программы
T	Регистр таймера – счётчика
BUS	Порт BUS
Pp (p=1,2)	Порт ввода-вывода
Pd (d=4-7)	Расширение портов ввода-вывода
b	Бит аккумулятора с номером b
КОП	Код операции
T	Тип команды (1-4)
B	Количество байтов в команде (1-2)
C	Число циклов за которе выполняется команда (1-2)

Таблица 5.4

Система команд Кр1816ВЕ048

Мнемоника	Код команды	**Выполняемая функция**	Количество	
	8 7 6 5 4 3 2 1 0		Б	Ц
1. Команды передачи данных				
MOV A,R	1111 1RRR	Пересылка из регистра в А	1	1
MOV A,@R	1111 000R	Пересылка из памяти в А	1	1

MOV A, #DATA	23	Пересылка непосредственных данных в А	2	2
MOV R,A	1010 1RRR	Пересылка из А в регистр	1	1
MOV @R,A	1010 000R	Пересылка из А в память	1	1
MOV R, #DATA	1011 IRRR	Пересылка непосредственных данных в регистр	2	2
MOV @R, #DATA	1011 000R	Пересылка непосредственных данных в память	2	2
MOV A,PSW	C7	Пересылка из регистра ССП в А	1	1
MOV PSW,A	D7	Пересылка из А в регистр ССП	1	1
XCH A,R	0010 1RRR	Обмен между регистром и А	1	1
XCH A,@R	0010 000R	Обмен между памятью и А	1	1
XCHD A,@R	0011 000R	Обмен младшими полубайтами между памятью и А	1	1
MOVX A,@R	1000 000R	Пересылка из внешней памяти в А	1	2
MOVX @R,A	1001 000R	Пересылка из А во внешнюю память	1	2
MOVP A, @A	A3	Пересылка из текущей страницы в А	1	2
MOVP3 A,@A	E3	Пересылка из страницы 3 в А	1	2
2. Команды инкремента				
INC R	0001 1RRR	Инкремент регистра	1	1
INC @R	0001 000R	Инкремент памяти	1	1
DEC R	1100 1RRR	Декремент регистра	1	1
INC A	17	Инкремент А (увеличение на 1)	1	1
DEC A	07	Декремент А (уменьшение на 1)	1	1
3. Арифметические и логические операции				
ADD A,R	0110 1RRR	Сложение регистра и А	1	1
ADD A, @R	0110 000R	Сложение памяти данных и А	1	1
ADD A, #DATA	03	Сложение непосредственных данных (2-го байта команды) и А	2	2

ADDC A, R	0111 1RRR	Сложение регистра и А с переносом	1	1
ADDC A,@R	0111 000R	Сложение памяти и А с переносом	1	1
ADDC A, #DATA	13	Сложение непосредственных данных и А с переносом	2	2
ANL A,R	0101 1RRR	Логическое И регистра и А	1	1
ANL A,@R	0101 000R	Логическое И памяти и А	1	1
ANL A, #DATA	53	Логическое И непосредственных данных и А	2	2
ORL A,R	0100 1RRR	Логическое ИЛИ регистра и А	1	1
ORL A,@R	0100 000R	Логическое ИЛИ памяти и А	1	1
ORL A, #DATA	43	Логическое И ЛИ непосредственных данных и А	2	2
XRL A,R	1100 1RRR	Исключающее ИЛИ регистра и А (сложение по модулю 2)	1	1
XRL A,@R	1100 000R	Исключающее ИЛИ памяти и А	1	1
XRL A, #DATA	D3	Исключающее непосредственных данных и А	2	2
CLR A	27	Обнуление А	1	1
CPL A	37	Инвертирование А 1	1	1
DA A	57	Преобразование А в двоично-десятичный код (десятичная коррекция)	1	1
SWAP A	47	Обмен местами полубайтов А	1	1
4. Команды сдвига				
RL A	E7	Циклический сдвиг А влево	1	1
RLC A	F7	Циклический сдвиг А влево через разряд переноса	1	1
RR A	77	Циклический сдвиг А вправо	1	1
RRC A	67	Циклический сдвиг А вправо через разряд переноса	1	1
5. Команды переходов				
JMP ADDR	AAA0 0100	Безусловный переход	2	2
JMPP @A	B3	Косвенный переход в текущей странице	1	2

DJNZ R, ADDR	1110 1RRR	Декремент регистра и переход, если не 0	2	2
JC ADDR	F6	Переход, если рязряд переноса в 1	2	2
JNC ADDR	Еб	Переход,, если разряд переноса в 0	2	2
JZ ADDR	06	Переход, если А равен 0	2	2
JNZ ADDR	96	Переход, если А не равен 0	2	2
JT0 ADDR	36	Переход, если 1 на выводе Т0	2	2
JNT0 ADDR	26	Переход, если 0 на выводе Т0	2	2
JT1 ADDR	56	Переход, если 1 на выводе Т1	2	2
JNT1 ADDR	46	Переход, если 0 на выводе Т1	2	2
JF0 ADDR	Вб	Переход, если флаг F0 в 1	2	2
JF1 ADDR	76	Переход, если флаг Fl в l	2	2
JTF ADDR	16	Переход, если флаг таймера в 1	2	2
JNI ADDR	86	Переход, если 0 на входе INT	2	2
JBB ADDR	BBB1 0010	Переход, если указанный разряд А в 1	2	2
CALL ADDR	AAA1 0100	Вызов подпрограммы	2	2
RET	83	Возврат из подпрограммы	1	2
RETR	93	Возврат из подпрограммы и восстановление состояния	1	2
6. Команды ввода вывода				
IN A,P	0000 10PP	Ввод из порта (Р1 или Р2) в А	1	2
OUTL P,A	0011 10PP	Вывод из А в порт	1	2
ANL P, # DATA	1001 10PP	Логическое И непосредственных данных и порта	2	2
ORL P, #DATA	1000 10PP	Логическое ИЛИ непосредственных данных и порта	2	2
INS A, BUS	08	Ввод с шины данных (порта Р0) в А	1	2
OUTL BUS, A	02	Вывод из А на шину данных	1	2

ANL BUS,#DATA	98	Логическое И непосредст-венных	2	2
ORL BUS,#DATA	88	Логическое ИЛИ непосред-ственных данных и шины данных	2	2
MOVD A, P	0000 11PP	Пересылка из порта-расширителя в младший полубайт А	1	2
MOVD P,A	0011 11PP	Пересылка младшего полу-байта А в порт-расширитель	1	2
ANLD P,A	1001 11PP	Логическое И младшего по-лубайта А и порта-расширителя	1	2
ORLD P,A	1000 11PP	Логическое ИЛИ младшего полубайта А и порта-расширителя	1	2
7. Команды состояний (флагов)				
CLR C	97	Обнуление разряда перено-са	1	1
CPL C	A7	Инвертирование разряда переноса	1	1
CLR F0	85	Обнуление F0	1	1
CPL F0	95	Инвертирование F0	1	1
CLR FI	A5	Обнуление FI	1	1
CPL FI	B5	Инвертирование FI	1	1
8. Команды таймера-счетчика				
MOV A,T	42	Пересылка из регистра тай-мера счетчика в А	1	1
MOV T,A	62	Пересылка из А в регистр таймера счетчика	1	1
STRT T	55	Пуск таймера	1	1
STRT CNT	45	Пуск счетчика	1	1
STOP TCNT	65	Останов таймера-счетчика	1	1
EN TCNTI	25	Разрешение прерываний от таймера- счетчика	1	1
DIS TCNTI	35	Запрещение прерываний от таймера- счетчика	1	1
9. Команды управления				
EN I	05	Разрешение внешних пре-рываний	1	1

DIS I	15	Запрещение внешних прерываний	1	1
SEL RB0	C5	Выбор нулевого банка памяти данных	1	1
SEL RB1	D5	Выбор первого банка памяти данных	1	1
SEL MB0	E5	Выбор нулевого банка памяти программ	1	1
SEL MB1	F5	Выбор первого банка памяти программ	1	1
ENT0 CLK	75	Разрешение выдачи импульсов синхронизации на вывод T0	1	1
NOP	00	Нет операции	1	1

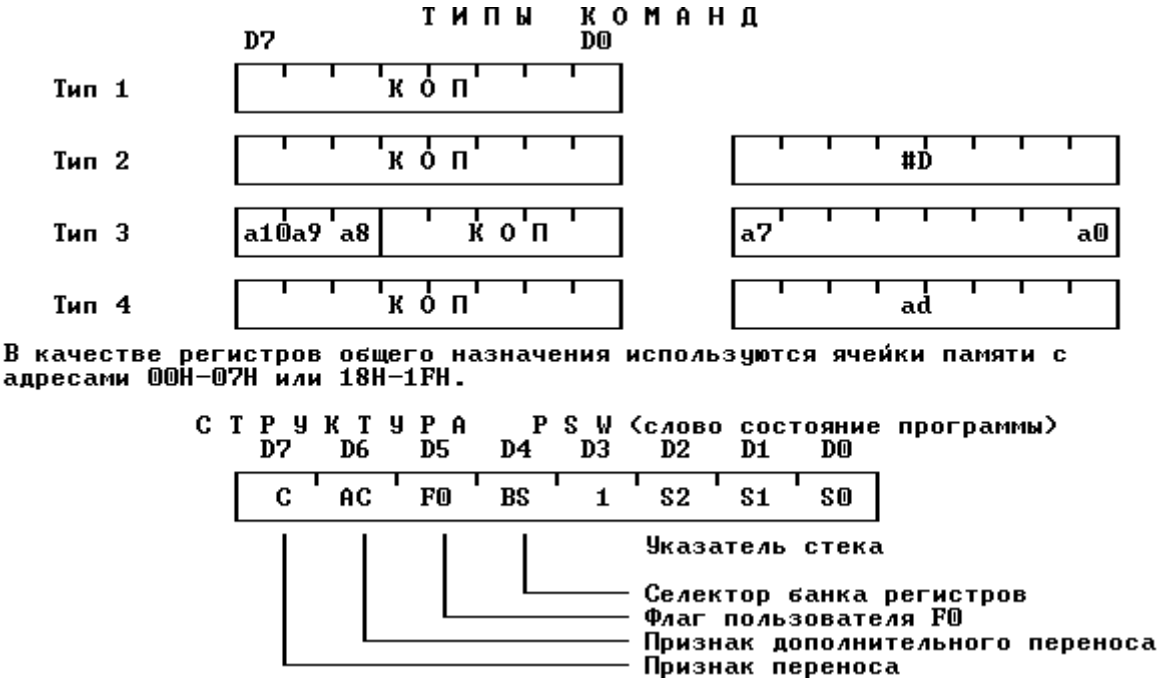

Рисунок 5.2 Форматы команд и слово-состояние ОМЭВМ 048.

5.3. Приложение 3. Система команд ОМЭВМ семейства МК051

Таблица 5.4

Условные обозначенья

Rn (n=0-7)	Регистр общего назначения в выбранном банке регистров
@Ri(i=0, 1)	Регистр общего назначения в выбранном банке регистров, используемый в качестве регистра косвенного адреса
#d	Непосредственные данные (8-разрядные)
#d16	Непосредственные данные (16-разрядные)
A	Аккумулятор
PC	Счетчик команд
DPTR	Регистр указателя данных
ad	Адрес ячейки памяти
ads	Адрес ячейки памяти источника
add	Адрес ячейки памяти приемника
ad11	11-разрядный абсолютный адрес перехода
ad16	16-разрядный абсолютный адрес перехода
rel	Относительный адрес перехода
bit	Адрес прямоадресуемого бита
/bit	Инверсия прямоадресуемого бита
( )	Содержимое ячейки памяти или регистра
КОП	Код операции
Т	Тип команды (1-13; смотри далее)
В	Количество байтов в команде (1-3)
С	Число циклов за которое выполняется команда (1,2,4)

Таблица 5.5

Система команд Кр1816ВЕ051

Мнемоника	Код	Функция	Т	Б	С
1.Команды пересылки					
MOV A,Rn	11101rrr	Пересылка в аккумулятор из регистра n	1	1	1

		(n=0...7)			
MOV A,ad	11100101	Пересылка в аккумулятор прямоадресуемого байта	3	2	1
MOV A,@Ri	1110011i	Пересылка в аккумулятор байта из РПД (i=0,1)	1	1	1
MOV A,#d	01110100	Загрузка в аккумулятор константы	2	2	1
MOV Rn,A	11111rrr	Пересылка в регистр из аккумулятора	1	1	1
MOV Rn,ad	10101rrr	Пересылка в регистр прямоадресуемого байта	3	2	2
MOV Rn,#d	01111rrr	Загрузка в регистр константы	2	2	1
MOV ad,A	11110101	Пересылка по прямому адресу аккумулятора	3	2	1
MOV ad,Rn	10001rrr	Пересылка по прямому адресу регистра	3	2	2
MOV add,ads	10000101	Пересылка прямоадресуемого байта по прямому адресу	9	3	2
MOV ad,@Ri	1000011i	Пересылка байта из РПД по прямому адресу	3	2	2
MOV ad,#d	01110101	Пересылка по прямому адресу константы	7	3	2
MOV @Ri,A	1111011i	Пересылка в РПД из аккумулятора	1	1	1
MOV @Ri,ad	0110011i	Пересылка в РПД прямоадресуемого байта	3	2	2
MOV @Ri,#d	0111011i	Пересылка в РПД константы	2	2	1
MOV DPTR,#d16	10010000	Загрузка указателя данных	13	3	2
MOVC A,@A+DPTR	10010011	Пересылка в аккумулятор байта из ПП	1	1	2
MOVC A,@A+PC	10000011	Пересылка в аккумулятор байта из ПП	1	1	2
MOVX A,@DPTR	11100000	Пересылка в аккумулятор байта из ВПД	1	1	2
MOVX @Ri,A	1111001i	Пересылка в ВПД из аккумулятора	1	1	2

MOVX @DPTR,A	11110000	Пересылка в расширенную ВПД из аккумулятора	1	1	2
PUSH ad	11000000	Загрузка в стек	3	2	2
POP ad	11010000	Извлечение из стека	3	2	2
XCH A,Rn	11001rrr	Обмен аккумулятора с регистром	1	1	1
XCH A,ad	11000101	Обмен аккумулятора с прямоадресуемым байтом	3	2	1
XCH A,@Ri	1100011i	Обмен аккумулятора с байтом из РПД	1	1	1
XCHD A,@Ri	1101011i	Обмен младшей тетрады аккумулятора с младшей тетрадой байта РПД	1	1	1
2.Арифметические операции					
ADD A,Rn	00101rrr	Сложение аккумулятора с регистром (n=0...7)	1	1	1
ADD A,ad	00100101	Сложение аккумулятора с прямоадресуемым байтом	3	2	1
ADD A,@Ri	0010011i	Сложение аккумулятора с байтом из РПД (i=0,1)	1	1	1
ADD A,#d	00100100	Сложение аккумулятора с константой	2	2	1
ADDC A,Rn	00111rrr	Сложение аккумулятора с регистром и переносом	1	1	1
ADDC A,ad	00110101	Сложение аккумулятора с прямоадресуемым байтом и переносом	3	2	1
ADDC A,@Ri	0011011i	Сложение аккумулятора с байтом из РПД и переносом	1	1	1
ADDC A,#d	00110100	Сложение аккумулятора с константой и переносом	2	2	1
DA A	11010100	Десятичная коррекция аккумулятора	1	1	1
SUBB A,Rn	10011rrr	Вычитание из аккумуля-	1	1	1

		тора регистра и заема			
SUBB A,ad	10010101	Вычитание из аккумулятора прямоадресуемого байта и заема	3	2	1
SUBB A,@Ri	1001011i	Вычитание из аккумулятора байта РПД и заема	1	1	1
SUBB A,#d	10010100	Вычитание из аккумулятора константы и заема	2	2	1
MUL AB	10100100	Умножение аккумулятора на регистр В	1	1	4
DIV AB	10000100	Деление аккумулятора на регистр В	1	1	4
3. Команды инкремента и декремента					
INC A	00000100	Инкремент аккумулятора	1	1	1
INC Rn	00001rrr	Инкремент регистра	1	1	1
INC ad	00000101	Инкремент прямоадресуемого байта	3	2	1
INC @R	0000011i	Инкремент байта в РПД	1	1	1
INC DPTR	10100011	Инкремент указателя данных	1	1	2
DEC A	00010100	Декремент аккумулятора	1	1	1
DEC Rn	00011rrr	Декремент регистра	1	1	1
DEC ad	00010101	Декремент прямоадресуемого байта	3	2	1
DEC @Ri	0001011i	Декремент байта в РПД	1	1	1
4.Команды логических операций					
ANL A,Rn	01011rrr	Логическое И аккумулятора и регистра	1	1	1
ANL A,ad	01010101	Логическое И аккумулятора и прямоадресуемого байта	3	2	1
ANL A,@Ri	0101011i	Логическое И аккумулятора и байта из РПД	1	1	1
ANL A,#d	01010100	Логическое И аккумулятора и константы	2	2	1
ANL ad,A	01010010	Логическое И прямоадресуемого байта	3	2	1

		и аккумулятора			
ANL ad,#d	01010011	Логическое И прямоадресуемого байта и константы	7	3	2
ORL A,Rn	01001rrr	Логическое ИЛИ аккумулятора и регистра	1	1	1
ORL A,ad	01000101	Логическое ИЛИ аккумулятора и прямоадресуемого байта	3	2	1
ORL A,@Ri	0100011i	Логическое ИЛИ аккумулятора и байта из РПД	1	1	1
ORL A,#d	01000100	Логическое ИЛИ аккумулятора и константы	2	2	1
ORL ad,A	01000010	Логическое ИЛИ прямоадресуемого байта и аккумулятора	3	2	1
ORL ad,#d	01000011	Логическое ИЛИ прямоадресуемого байта константы	7	3	2
XRL A,Rn	01101rrr	Исключающее ИЛИ аккумулятора и регистра	1	1	1
XRL A,ad	01100101	Исключающее ИЛИ аккумулятора и прямоадресуемого байта	3	2	1
XRL A,@Ri	0110011i	Исключающее ИЛИ аккумулятора и и байта	1	1	1
XRL A,#d	01100100	Исключающее ИЛИ аккумулятора и константы	2	2	1
XRL ad,A	01100010	Исключающее ИЛИ прямоадресуемого байта и аккумулятора	3	2	1
XRL ad,#d	01100011	Исключающее ИЛИ прямоадресуемого байта и константы	7	3	2
CLR A	11100100	Сброс аккумулятора	1	1	1
CPL A	11110100	Инверсия аккумулятора	1	1	1
5. Команды сдвига					
RL A	00100011	Сдвиг аккумулятора влево циклический	1	1	1
RLC A	0011001	Сдвиг аккумулятора	1	1	1

	1	влево через перенос			
RR A	00000011	Сдвиг аккумулятора вправо циклический	1	1	1
RRC A	00010011	Сдвиг аккумулятора вправо через перенос	1	1	1
SWAP A	11000100	Обмен местами тетрад в аккумуляторе	1	1	1
6. Команды операций с битами					
CLR C	11000011	Сброс переноса	1	1	1
CLR bit	11000010	Сброс бита	4	2	1
SETB C	11010011	Установка переноса	1	1	1
SETB bit	11010010	Установка бита	4	2	1
CPL C	10110011	Инверсия переноса	1	1	1
CPL bit	10110010	Инверсия бита	4	2	1
ANL C,bit	10000010	Логическое И бита и переноса	4	2	2
ANL C,/bit	10110000	Логическое И инверсии бита и переноса	4	2	2
ORL C,bit	01110010	Логическое ИЛИ бита и переноса	4	2	2
ORL C,/bit	10100000	Логическое ИЛИ инверсии бита и переноса	4	2	2
MOV C,bit	10100010	Пересылка бита в перенос	4	2	1
MOV bit,C	10010010	Пересылка переноса в бит	4	2	2
7. Команды передачи управления					
LJMP ad 16	00000010	Длинный переход в полном объеме памяти программ	12	3	2
AJMP ad 11	a10a9a800001	Абсолютный переход внутри страницы в 2 Кбайта	6	2	2
SJMP re1	10000000	Короткий относительный переход внутри страницы в 256 байт	5	2	2

JMP @A+DPTR	01110011	Косвенный относительный переход	1	1	2
JZ re1	01100000	Переход, если аккумулятор равен нулю	5	2	2
JNZ re1	01110000	Переход, если аккумулятор не равен нулю	5	2	2
JC re1	01000000	Переход, если перенос равен единице	5	2	2
JNC re1	01010000	Переход, если перенос равен нулю	5	2	2
JB bit,re1	00100000	Переход, если бит равен единице	11	3	2
JNB bit,re1	00110000	Переход, если бит равен нулю	11	3	2
JBC bit,re1	00010000	Переход, если бит установлен ,с последующим сбросом бита	11	3	2
DJNZ Rn,re1	11011rrr	Декремент регистра и переход, если не нуль	5	2	2
DJNZ ad,re1	11010101	Декремент прямоадресуемого байта и переход, если не нуль	8	3	2
CJNE A,ad,re1	10110101	Сравнение аккумулятора с пряморадресуемым байтом и переход, если не равно	8	3	2
CJNE A,#d,re1	10110100	Сравнение аккумулятора с константой и переход, если не равно	10	3	2
CJNE @Ri,#d,re1	1011011i	Сравнение регистра с РПД с константой и переход, если не равно	10	3	2
LCALL ad16	00010010	Длинный вызов подпрограммы	12	3	2
ACALL ad11	a10a9a810001	Абсолютный вызов подпрограммы в пределах страницы 2 Кбайта	6	2	2
RET	00100010	Возврат из подпрограмм	1	1	2
RETI	00110010	Возврат из подпрограммы обработки прерывания	1	1	2

NOP	00000000	Холостая команда	1	1	1

Команды, модифицирующие флаги результата

Команды	Флаги	Команды	Флаги
ADD	C,OV,AC	CLR C	C=0
ADDC	C,OV,AC	CPL C	C = $\overline{C}$
SUBB	C,OV,AC	ANL C,b	C
MUL	C=0,OV	ANL C,/b	C
DIV	C=0,OV	ORL C,b	C
DA	C	ORL C,/b	C
RRC	C	MOV C,b	C
RLC	C	CJNE	C
SETB C	C=1		

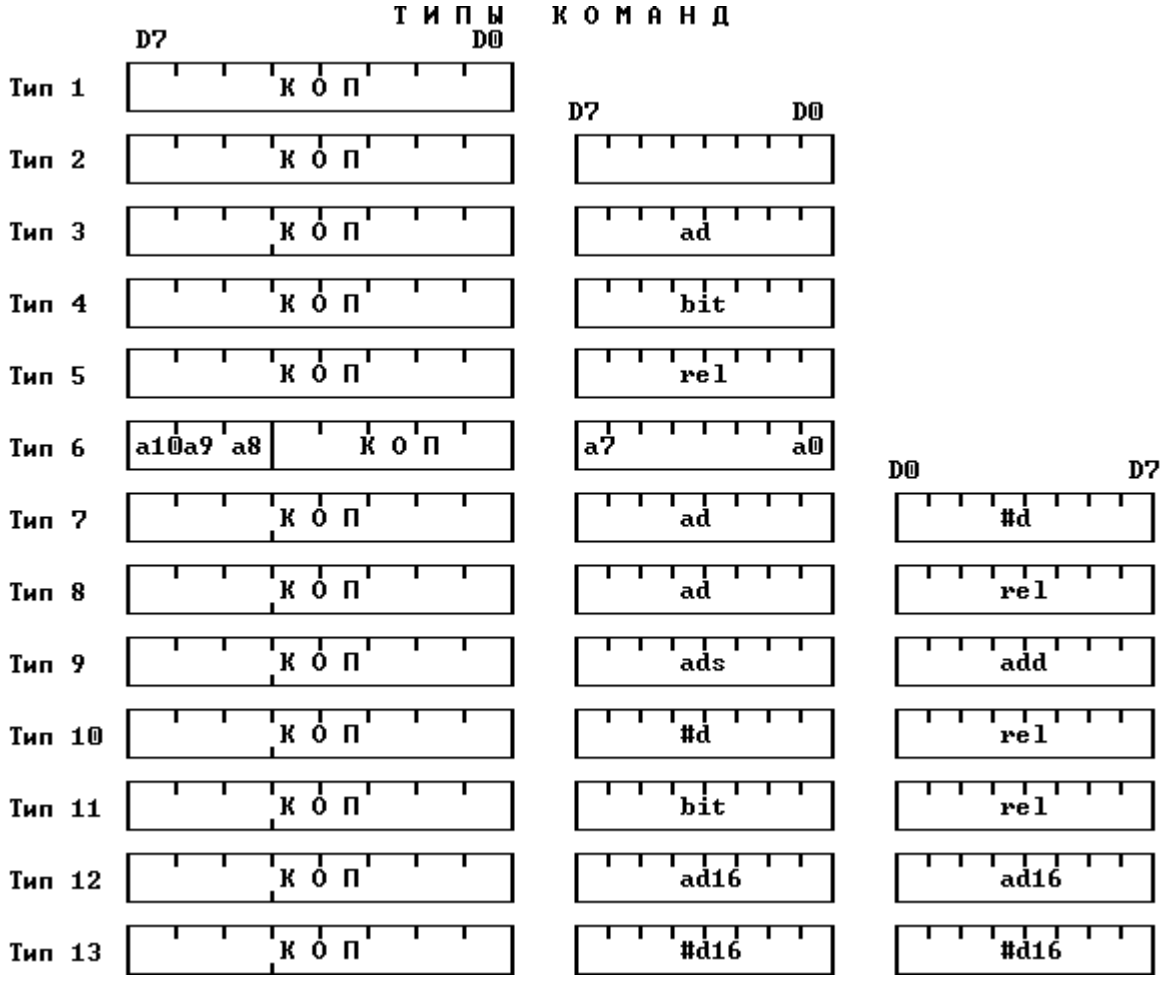

Рисунок 5.4 Форматы команд ОМЭВМ семейства МК051.

5.4. Приложение 4. Система команд PIC процессора 16C84

Каждая команда PIC16C84 - это 14-битовое слово, которое разделено по смыслу на следующие части: 1. код операции, 2. поле для одного и более операндов, которые могут участвовать или нет в этой команде. Система команд PIC16C84 включает в себя байт-ориентированные команды, бит-ориентированные, операции с константами и команды передачи управления.

Для байт-ориентированных команд **"f"** обозначает собой регистр, с которым производится действие; **"d"** - бит определяет, куда положить результат. Если **"d"** =0, то результат будет помещен в **W** регистр, при **"d"**=1 результат будет помещен в **"f",** упомянутом в команде.

Для бит-ориентированных команд **"b"** обозначает номер бита, участвующего в команде, а **"f"** -это регистр, в котором этот бит расположен.

Для команд передачи управления и операций с константами, **"к"** обозначает восьми или одинадцатибитную константу.

Все команды выполняются в течение одного командного цикла. В двух случаях исполнение команды занимает два командных цикла: 1. Проверка условия и переход, 2.изменение программного счетчика как результат выполнения команды. Один командный цикл состоит из четырех периодов генератора. Таким образом, для генератора с частотой 4 МГц время исполнения командного цикла будет 1 мкс.

Таблица 5.6

Система команд PIC контроллера 16C84

Мнемоника	**Функция**	**Примечания**
1. Байт ориентированные команды		
ADDWF f,d	Сложение W с f	2,3
ANDWF f,d	Логическое И W и f	2,3
CLRF f	Сброс регистра f	3
CLRW	Сброс регистра W	
COMF f,d	Инверсия регистра f	2,3
DECF f,d	Декремент регистра f	2,3
DECFSZ f,d	Декремент f, пропустить команду, если 0	2,3
INCF f,d	Инкремент регистра f	2.3
INCFSZ f,d	Инкремент f, пропустить команду, если 0	2,3
IORWF f,d	Логическое ИЛИ W и f	2,3
MOVF f,d	Пересылка регистра f	2,3
MOVWF f	Пересылка W в f	3
NOP	Холостая команда	
RLF f,d	Сдвиг f влево через перенос	2,3
RRF f,d	Сдвиг f вправо через перенос	2,3
SUBWF f,d	Вычитание W из f	2,3
SWAPF f,d	Обмен местами тетрад в f	2,3
XORWF f,d	Исключающее ИЛИ W и f	2,3
ADDLW k	Сложение константы с W	
ANDLW k	Логическое И константы и W	
IORLW k	Логическое ИЛИ константы и W	
SUBLW k	Вычитание W из константы	

MOVLW k	Пересылка константы в W	
XORLW k	Исключающее ИЛИ константы и W	
OPTION	Загрузка W в OPTION регистр	
TRIS f	Загрузка TRIS регистра	
2. Бит ориентированные команды		
BCF f,b	Сброс бита в регистре f	2,3
BSF f,b	Установка бита в регистре f	2,3
BTFSC f,b	Пропустить команду, если бит равен нулю	
BTFSS f,b	Пропустить команду, если бит равен единице	
3. Переходы		
CALL k	Вызов подпрограммы	
CLRWDT	Сброс Watchdog таймера	
GOTO k	Переход по адресу	
RETLW k	Возврат из подпрограммы с загрузкой константы в W	
RETFIE	Возврат из прерывания	
RETURN	Возврат из подпрограммы	
SLEEP	Переход в режим SLEEP	

Примечание 1: Команды TRIS и OPTION помещены в перечень команд для совместимости с семейством PIC16C5X. Их использование не рекомендуется. В PIC16C84 регистры TRIS и OPTION доступны для чтения и записи как обычные регистры с номером. Предупреждаем, что эти команды могут не поддерживаться в дальнейших разработках PIC16CXX.

Примечание 2: Когда модифицируется регистр ввода/вывода, например MOVF 6,1, значение, используемое для модификации считывается непосредственно с ножек кристалла. Если значение защелки вывода для ножки, запрограммированной на вывод равно "1", но внешний сигнал на этом выводе "0" из-за "навала" снаружи, то будет считываться "0".

Примечание 3: Если операндом этой команды является регистр f1 (и, если допустимо, d=1), то делитель, если он подключен к RTCC, будет обнулен.

Обзор регистров и ОЗУ. Область ОЗУ организована как 128 x 8. К ячейкам ОЗУ можно адресоваться прямо или косвенно, через регистр указатель FSR (04h). Это также относится и к EEPROM памяти данных-констант.

В регистре статуса (03h) есть биты выбора страниц, которые позволяют обращаться к четырем страницам будущих модификаций этого кристалла.

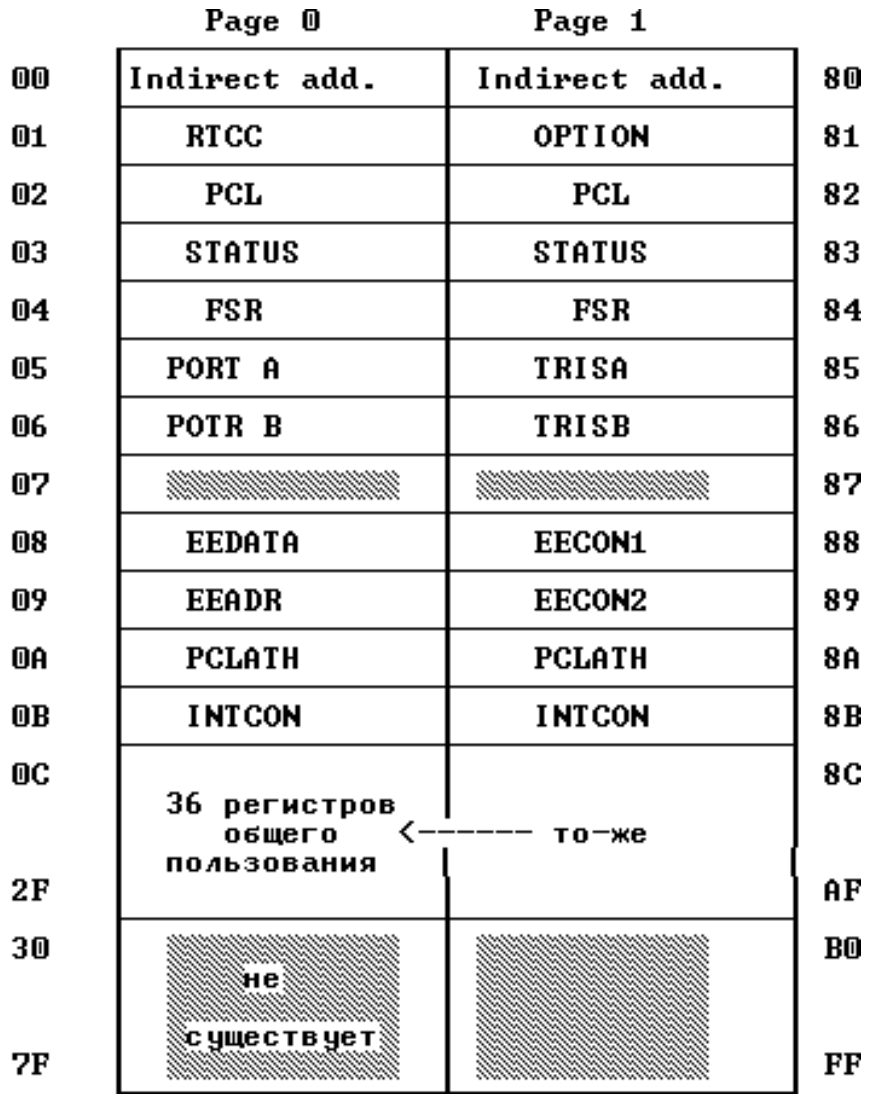

Рисунок 5.5. Распределение памяти данных PIC16C84.

Однако для PIC16C84 память данных существует только до адреса 02Fh. Первые 12 адресов используются для размещения регистров специального назначения. Регистры с адресами 0Ch-2Fh могут быть использованы, как регистры общего назначения, которые представляют собой статическое ОЗУ. Некоторые регистры специального назначения продублированы на обеих страницах, а некоторые расположены на странице 1 отдельно. Когда установлена страница 1, то обращение к адресам 8Ch-AFh фактически адресует

страницу 0. К регистрам можно адресоваться прямо или косвенно. В обоих случаях можно адресовать до 512 регистров.

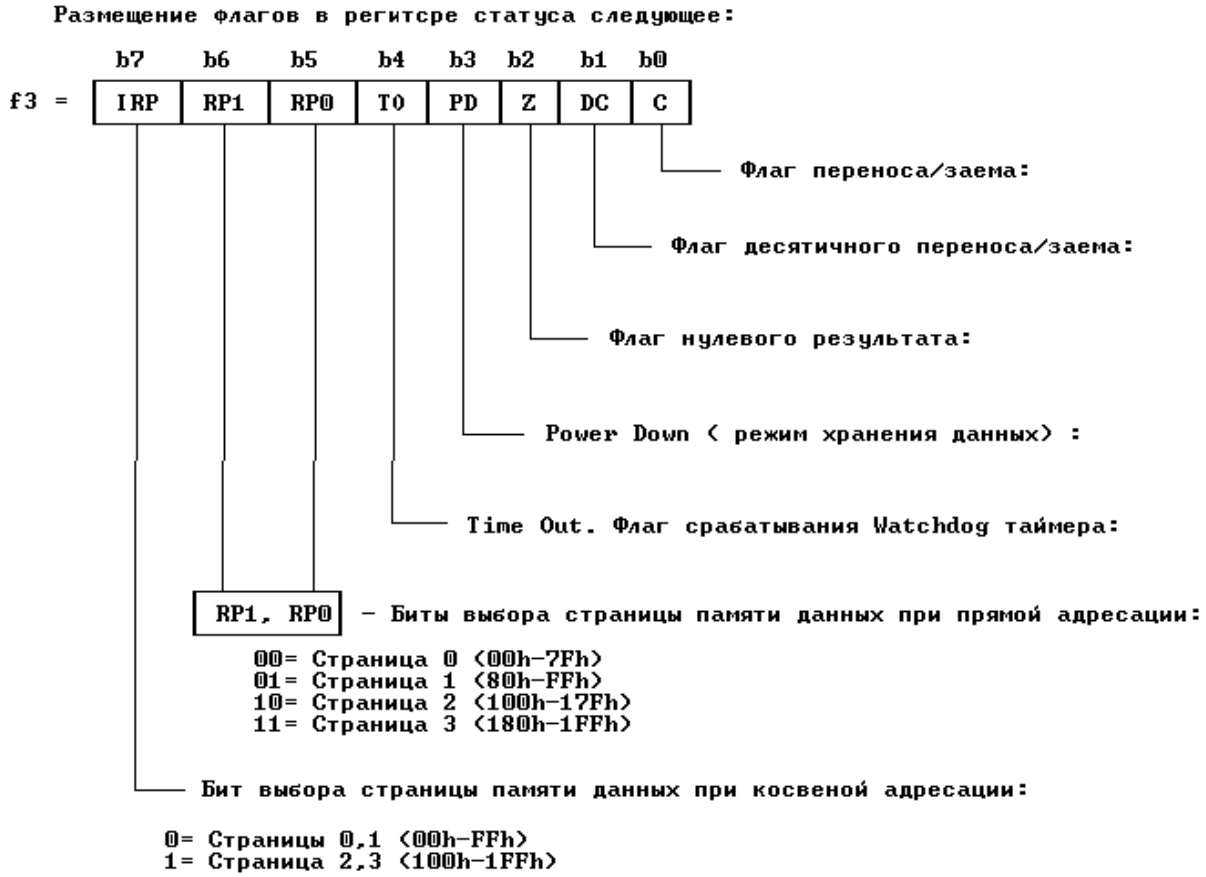

Рисунок 5.6. Формат слова-состояния PIC 16C84.

Рисунок 5.8. Формат слова-состояния ОМЭВМ семейства 051.

Блок регистров специальных функций

Символ	Наименование	Адрес
ACC	Аккумулятор	0E0H
B	Регистр-расширитель аккумулятора	0F0H
PSW	Слово состояния программы	0D0H
SP	Регистр-указатель стека	81H
DPTR	Регистр-указатель данных (DPH)	83H
	(DPL)	82H
P0	Порт 0	80H
P1	Порт 1	90H
P2	Порт 2	0A0H
P3	Порт 3	0B0H
IP	Регистр преоритетов	0B8H
IE	Регистр маски прерываний	0A8H
TMOD	Регистр режима таймера/счетчика	89H
TCON	Регистр управления/статуса таймера	88H
TH0	Таймер 0 (старший байт)	8CH
TL0	Таймер 0 (младший байт)	8AH
TH1	Таймер 1 (старший байт)	8DH
TL1	Таймер 1 (младший байт)	8BH
SCON	Регистр управления приемопередатчиком	98H
SBUF	Буфер приемопередатчика	99H
PCON	Регистр управления мощностью	87H

Примечание.Регистры,имена которых отмечены знаком (*), допускают адресацию отдельных бит.

6. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Алексенко А.А., Галицын А.А., Иванников А.Д. Проектирование радиоэлектронной аппаратуры на микропроцессорах: Программирование, типовые решения, методы отладки. - М.: Радио и связь, 1984. - 272 с.

2. Микропроцессоры. Аппаратурно-программные средства отладки. /Васильев А.П., Горовой В.Р.; Под ред. Л.Н. Преснухина. - М.: Высшая школа, 1984. - 95 с.

3. Зеленко Г.В., Иванников А.Д., Сыпчук П.П. Проектирование и отладка микропроцессорных систем. - М,: Машиностроение, 1982. - 56 с.

4. Левенталь Л. Введение в микропроцессоры. - М.: Энергоатомиздат, 1983. - 463 с.

5. Каган Б.М., Сташин В.В. Основы проектирования микропроцессорных устройств автоматики. - М.: Энергоатомиздат, 1987. – 304 с.

6. Григорьев В.Л. Программное обеспечение микропроцессорных систем. - М.: Энергоатомиздат, 1983. - 207с.

7. Левенталь Л., Сэйвилл У. Программирование на языке ассемблера для микропроцессоров 8080 и 8085. - М.: Радио и связь, 1987. - 448 с.

8. Како Н., Яманэ Я. Датчики и микро-ЭВМ. - Л.: Энергоатомиздат. 1986. - 120 с.

9. ГОСТ 2.743-82. ЕСКД. Обозначения условные графические в схемах. Элементы цифровой техники. - М.: Изд-во стандартов, 1982.

10. Микропроцессоры /Под ред. Л.Н. Преснухина. - М.: Высшая школа, 1986. – 386 с.

11. Микропроцессоры и микроЭВМ в системах автоматического управления: Справочник /С.Т. Хвощ, Н.Н. Варлинский, Е.А. Попов; Под ред. С.Т. Хвоща. - Л.: Машиностроение, 1987. - 640 с.

12. Рафикузаман М. Микропроцессоры и машинное проектирование микропроцессорных систем. - М.: Мир, 1988. - 312 с.

13. Кушнер В.Е., Панфилов Д.И., Шаронин С.Г. Многофункциональный комплекс программно-аппаратных средств для семейства однокристальных ЭВМ серии 1816// Микропроцессорные средства и системы. - 1988. - № 6. - С. 3-4.

14. Боборыкин А.В., Липовецкий Г.П., Литвинский Г.В., и др. Однокристальные микро-ЭВМ. - М.: Бином, 1994.-400 с.

15. MCS 48. Family of single-chip microcomputers. User's manual. Intel Corp. July 1979. - 320 p.

15. Intel 8051. User's manual. Intel Corp., 1980. - 110 p.

17. Алексенко А.Г., Галицин А.А., Иванников А.Д. Проектирование радиоэлектронной аппаратуры на микропроцессорах. - М.: Радио и

связь. 1984. – 287 с.

18. Бриндли К. Измерительные преобразователи. Справочное пособие. - М.: Энергоатомиздат, 1991. - 144 с.

19. Мячев А. А. Организация ввода-вывода. - М.: Энергоатомиздат. 1983. - 210 с.

20. Гнатек Ю.Р. Справочник по цифро-аналоговым и аналого-цифровым преобразователям. - М.: Радио и связь, 1982. - 552 с.

21. Шевкопляс Б.В. Микропроцессорные структуры. Инженерные решения. - М.: Радио и связь, 1986. – 302 с.

ОГЛАВЛЕНИЕ