Федеральное агентство по образованию
ГОУ ВПО «Уральский государственный технический университет – УПИ»

В.П. Мокрецов

МИКРОПРОЦЕССОРЫ И МПС

Часть 2

Микроконтроллеры

Учебное электронное текстовое издание
Научный редактор: доц., канд. техн. наук В.И. Паутов

Пособие предназначено для студентов специальностей 220101– Управление и информатика в технических системах, 230101 – Вычислительные машины, комплексы, системы и сети.

Описаны архитектура простейшего микропроцессора и семейство интегральных программируемых микросхем; организация различных подсистем микропроцессорных устройств и МП-систем. Рассмотрено программирование различных операций в системе команд процессора, приведены упражнения, контрольные вопросы и задания.

Екатеринбург
2007

ПРЕДИСЛОВИЕ

Предлагаемое учебное пособие является второй частью курса лекций по микропроцессорным системам, читаемых в ГОУ ВПО «Уральский государственный технический университет – УПИ» для студентов специальностей 220201 - Управление и информатика в технических системах и 230101 – Вычислительные машины, комплексы, системы и сети, и посвящено организации 8 – разрядных микроконтроллеров.

В пособии приведен сравнительный анализ наиболее популярных семейств микроконтроллеров. Для удовлетворения запросов потребителей выпускается большая номенклатура 8–, 16–, и 32–разрядных микроконтроллеров. Восьми разрядные микроконтроллеры представляют наиболее многочисленную группу, которая вполне достаточна для решения широкого круга задач управления различными объектами.

Детальное описание структурной организации, функционирования и системы команд выполнено для семейства микроконтроллеров фирмы Intel 8051, а именно К1816ВЕ51. Это обусловлено наличием соответствующих средств для постановки лабораторного практикума. Пособие предназначено для совершенствования самостоятельной подготовки студентов, в том числе для заочной и дистанционной форм обучения.

Все лекции объединяются на электронных носителях с методическими указаниями к выполнению лабораторных и практических работ, контрольными заданиями, заданиями на курсовое проектирование, рабочими программами и распространяются на Flash-ROM или по сети. На этих же носителях размещаются инструментальные средства проектирования и отладки программного обеспечения микроконтроллерных устройств. Перечисленные материалы дополняются информационно – справочной базой данных, необходимой для успешного изучения дисциплины «Микропроцессорные системы» и выполнения всех учебных мероприятий.

Микропроцессорная техника динамично развивается. Известны и более совершенные микроконтроллерные семейства и средства проектирования систем на их основе. Методические разработки по изучению таких средств подготавливаются для расширения данного пособия.

1. МИКРОКОНТРОЛЛЕРЫ 8-РАЗРЯДНЫЕ

1.1. Структура 8-разрядных микроконтроллеров

Микроконтроллеры (МК) имеют модульный принцип организации, представляют собой законченную микропроцессорную систему обработки информации, созданную в виде большой интегральной микросхемы. МК в пределах одного кристалла объединяет основные функциональные блоки микропроцессорной управляющей системы: центральный процессор (ЦПУ), постоянное запоминающее устройство (ПЗУ), оперативное запоминающее устройство (ОЗУ), периферийные устройства ввода и вывода информации (УВВ). Применяется модульный принцип построения МК. Все МК одного семейства содержат базовый функциональный блок и изменяемый функциональный блок, который различается для разных моделей в пределах одного семейства (рис. 1.1).

Базовый функциональный блок включает:

- центральный процессор;
- внутренние магистрали адреса, данных и управления;
- схему формирования многофазной импульсной последовательности для тактирования ЦПУ и межмодульных магистралей;
- устройство управления режимами работы МК, такими как активный режим, режим пониженного энергопотребления, состояния начального запуска и прерывания.

Базовый функциональный блок называют процессорным ядром МК. Тип процессорного ядра определяет имя семейства МК, основой которого оно явля-

ется. Например, ядро MCS-51 – ядро семейства МК Intel 8xC51, ядро PIC18 – процессорное ядро Microchip PIC18.

Изменяемый функциональный блок включает модули разных типов памяти, модули периферийных устройств, модули генераторов синхронизации и некоторые дополнительные модули специальных режимов работы МК. Каждый модуль подключается к магистралям процессорного ядра. Это позволяет на уровне функционального проектирования новой модели МК подсоединять разнообразные модули к магистралям процессорного ядра, создавая, таким образом разнообразные по структуре МК в пределах одного семейства. Совокупность модулей, разработанных для определенного процессорного ядра, принято называть библиотекой периферийных модулей. Библиотека современного семейства МК включает модули пяти функциональных групп:

1) модули памяти;

2) модули периферийных устройств;

3) модули встроенных генераторов синхронизации;

4) модули контроля напряжения питания и хода выполнения программы;

5) модули внутрисхемной отладки и программирования.

Массив ячеек памяти, доступных для чтения, стирания и записи информации, дополненных аналоговыми и цифровыми схемами управления, а также регистры специальных функций для задания режимов работы объединяются в функциональный блок, который и называют модулем памяти ОЗУ и ПЗУ.

В направлениях развития 8-разрядной элементной базы МК прослеживается тенденция к закрытой архитектуре, при которой линии внутренних магистралей адреса и данных отсутствуют на выводах корпуса МК, что не позволяет подключать внешние микросхемы памяти. Разработчики обычно предлагают ряд модификаций МК с одним и тем же набором периферийных модулей, различающихся объемом резидентной памяти программ и данных.

Группа модулей периферийных устройств имеет следующие основные типы:

- параллельные порты ввода-вывода;
- таймеры-счетчики, таймеры периодических прерываний, процессоры событий;
- контроллеры последовательного интерфейса связи нескольких типов (UART, SCI, SPI, IIC, CAN, USB);
- аналого-цифровые преобразователи (АЦП);
- цифроаналоговые преобразователи (ЦАП);
- контроллеры ЖК индикаторов и светодиодной матрицы.

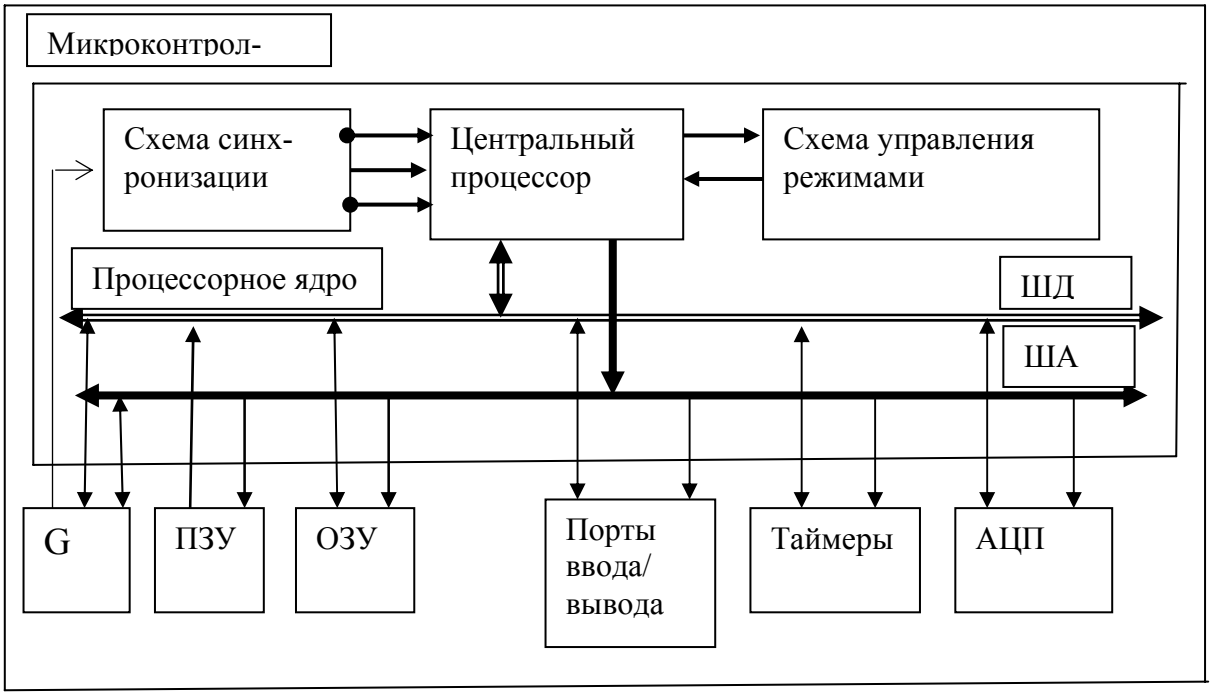

Рис. 1. Модульная структура МК.

2. МИКРОКОНТРОЛЛЕР К1816ВЕ51

Микроконтроллер К1816ВЕ51 является представителем семейства микроконтроллеров фирмы Intel 8051.

Микроконтроллер выполнен на основе высокоуровневой n-МОП технологии. Через четыре программируемых параллельных порта ввода/вывода и один по-

следовательный порт микроконтроллер взаимодействует с внешними устройствами. Основу структурной схемы (рис.2.) образует внутренняя двунаправленная 8-битная шина, которая связывает между собой основные узлы и устройства микроконтроллера: резидентную память программ (RPM), резидентную память данных (RDM), арифметико-логическое устройство (ALU), блок регистров специальных функций, устройство управления (CU) и порты ввода/вывода (P0-P3).

2.1. Арифметико-логическое устройство

8-битное арифметико-логическое устройство (ALU) 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. К входам подключены программно-недоступные регистры T1 и T2, предназначенные для временного хранения операндов, схема десятичной коррекции (DCU) и схема формирования признаков результата операции(PSW).

Простейшая операция сложения используется в ALU для инкрементирования содержимого регистров, продвижения регистра-указателя данных (RAR) и автоматического вычисления следующего адреса резидентной памяти программ.

Простейшая операция вычитания используется в ALU для декрементирования регистров и сравнения переменных.

Простейшие операции автоматически образуют «тандемы» для выполнения таких операций, как, например, инкрементирование 16-битных регистровых пар. В ALU реализуется механизм каскадного выполнения простейших операций для реализации сложных команд. Так, например, при выполнении одной из команд условной передачи управления по результату сравнения в ALU трижды инкрементируется счетчик команд (PC), дважды производится чтение из RDM, выполняется арифметическое сравнение двух переменных, формируется 16-битный адрес перехода и принимается решение о том, делать или не делать переход по программе. Все перечисленные операции выполняются всего лишь за 2 мкс.

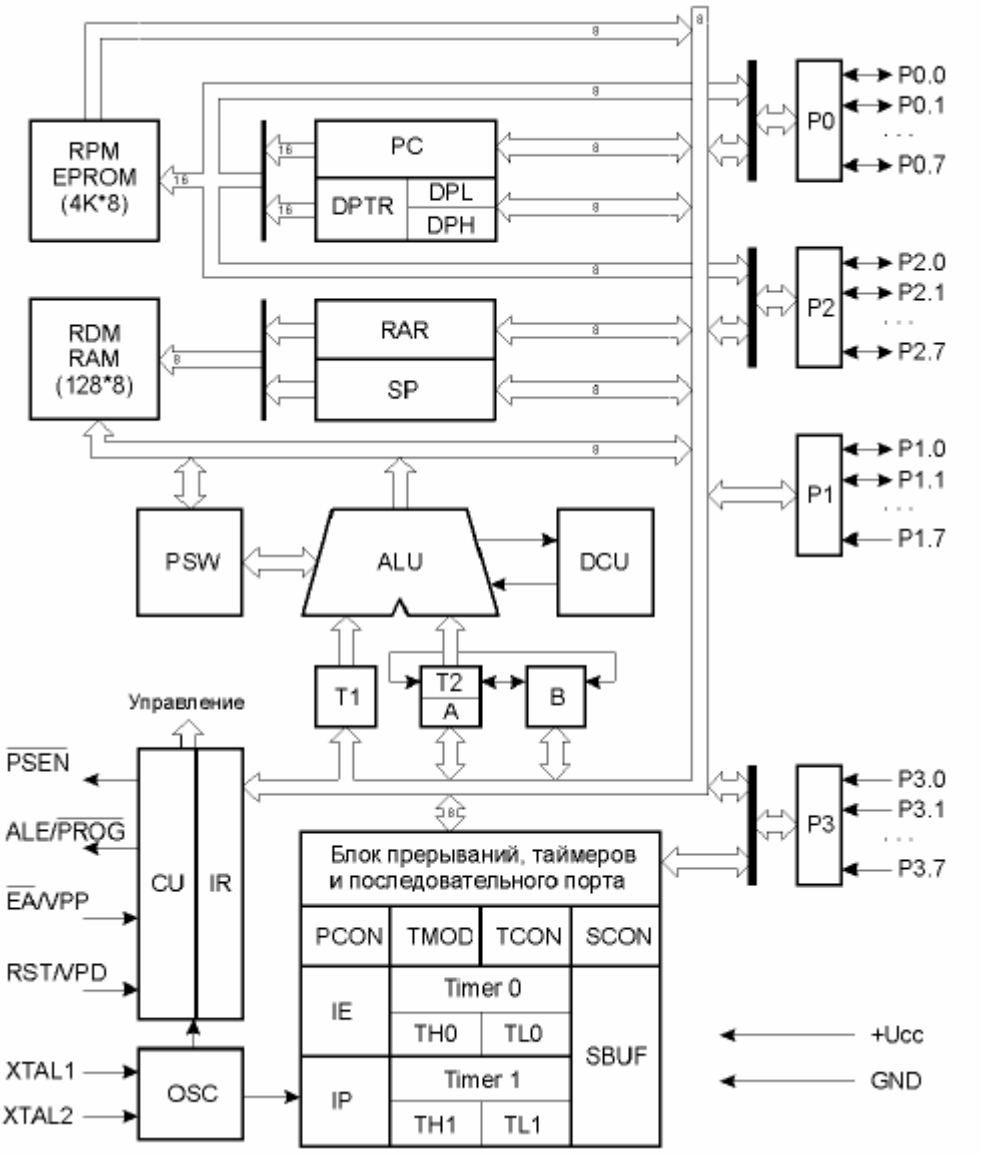

Рис.2. Структурная схема микроконтроллера КМ1816ВЕ51

Важной особенностью ALU 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. Эта способность достаточно важна, поскольку для управления объектами часто применяются алгоритмы, содержащие операции над входными и выходными булевыми переменными, реализация которых средствами обычных микропроцессоров сопряжена с определенными трудностями.

ALU может оперировать четырьмя типами информационных объектов: булевыми (1 бит), цифровыми (4 бита), байтными (8 бит) и адресными (16 бит). В ALU выполняется 51 различная операция пересылки или преобразования этих данных. Поскольку используется 11 режимов адресации (7 для данных и 4 для адресов), то путем комбинирования операции и режима адресации базовое число команд 111 расширяется до 255 из 256 возможных при однобайтном коде операции.

2.2. Резидентная память программ и данных

Резидентные (размещенные на кристалле) память программ (RPM) и память данных (RDM) физически и логически разделены, имеют различные механизмы адресации, работают под управлением различных сигналов и выполняют разные функции.

Память программ RPM имеет емкость 4 Кбайта и предназначена для хранения команд, констант, управляющих слов инициализации, таблиц перекодировки входных и выходных переменных и т.п. Память имеет 16-битную шину адреса, через которую обеспечивается доступ из программного счетчика PC или из регистра указателя данных (DPTR). DPTR выполняет функции базового регистра при косвенных переходах по программе или используется в операциях с таблицами.

Память данных RDM предназначена для хранения переменных в процесс выполнения прикладной программы, адресуется одним байтом и имеет емкость 128 байт. Кроме того, к ее адресному пространству примыкают адреса регистров специальных функций, которые перечислены в табл. 1.

Память программ, так же как и память данных, может быть расширена до 64 Кбайт путем подключения внешних микросхем.

2.3. Аккумулятор, регистры общего назначения и флаги

Аккумулятор (A) является источником операнда и местом фиксации результата при выполнении арифметических, логических операций и ряда операций передачи данных. Кроме того, только с использованием аккумулятора могут быть выполнены операции сдвигов, проверка на нуль, формирование флага паритета и т.п. В распоряжении пользователя имеются 8 регистров общего назначения R0–R7 одного из четырех возможных банков. При выполнении многих команд в ALU формируется ряд признаков операции (флагов), которые фиксируются в регистре PSW. В табл. 2 приводится перечень флагов PSW, даются их символические имена и описываются условия их формирования.

Таблица 1

Блок регистров специальных функций

	Символ	Наименование		Адрес
*	A	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 прерываний		0B8H
*	IE	Регистр маски прерываний		0A8H
	TMOD	Регистр режима таймера/счётчика		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 приёмопередатчиком		98H
	SBUF	Буфер приёмопередатчика		99H
	PCON	Регистр управления мощностью		87H

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

Таблица 2

Формат слова состояния программы PSW

Символ	Разряд	Имя и назначение
C	PSW.7	Флаг переноса. Устанавливается и сбрасывается аппаратно или программно при выполнении арифметических и логических операций
AC	PSW.6	Флаг вспомогательного переноса. Устанавливается и сбрасывается только аппаратно при выполнении команд сложения и вычитания и сигнализирует о переносе или займе 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 программно для выбора рабочего банка регистров (табл. 3)
RS0	PSW.3	
OV	PSW.2	Флаг переполнения. Устанавливается и сбрасывается аппаратно при выполнении арифметических операций
-	PSW.1	Не используется
P	PSW.0	Флаг паритета. Устанавливается и сбрасывается аппаратно в каждом цикле и фиксирует нечётное/чётное число единичных битов в аккумуляторе, т.е. выполняет контроль по четности

Таблица 3

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

Наиболее «активным» флагом PSW является флаг переноса, который принимает участие и модифицируется в процессе выполнения множества операций, включая сложение, вычитание и сдвиги. Кроме того, флаг переноса (C) выполняет функции «булева аккумулятора» в командах, манипулирующих с битами. Флаг переполнения (OV) фиксирует арифметическое переполнение при операциях над целыми числами со знаком и делает возможным использование арифметики в дополнительных кодах. ALU не управляет флагами селекции банка регистров (RS0, RS1), их значение полностью определяется прикладной программой и используется для выбора одного из четырех регистровых банков.

В микропроцессорах, архитектура которых опирается на аккумулятор, большинство команд работают с ним, используя неявную адресацию. В Intel 8051 дело обстоит иначе. Хотя процессор имеет в своей основе аккумулятор, он может выполнять множество команд и без его участия. Например, данные могут быть переданы из любой ячейки RDM в любой регистр, любой регистр может быть загружен непосредственным операндом и т.д. Многие логические операции могут быть выполнены без участия аккумулятора. Кроме того, переменные могут быть инкрементированы, декрементированы и проверены без использования аккумулятора. Флаги и управляющие биты могут быть проверены и изменены аналогично.

2.4. Регистры-указатели

8-битный указатель стека (SP) может адресовать любую область RDM. Его содержимое инкрементируется прежде, чем данные будут запомнены в стеке в ходе выполнения команд PUSH и CALL. Содержимое SP декрементируется после выполнения команд POP и RET. Подобный способ адресации элементов стека называют прединкрементным/постдекрементным. В процессе инициализации микроконтроллера после сигнала RST в SP автоматически загружается код 07H. Это значит, что если прикладная программа не переопределяет стек, то первый элемент данных в стеке будет располагаться в ячейке RDM с адресом 08H.

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

2.5. Регистры специальных функций

Регистры с символическими именами IP, IE, TMOD, TCON, SCON и PCON используются для фиксации и программного изменения управляющих битов и битов состояния схемы прерывания, таймера/счетчика, приемопередатчика последовательного порта и для управления энергопотреблением. Их организация будет описана ниже при рассмотрении особенностей работы микроконтроллера в различных режимах.

2.6. Устройство управления и синхронизации

Кварцевый резонатор, подключаемый к внешним выводам микроконтроллера, управляет работой внутреннего генератора, который в свою очередь формирует сигналы синхронизации. Устройство управления (CU) на основе сигналов синхронизации формирует машинный цикл фиксированной длительности, равной 12 периодам резонатора. Большинство команд микроконтроллера выполняется за один машинный цикл. Некоторые команды, оперирующие с 2-байтными словами или связанные с обращением к внешней памяти, выполняются за два машинных цикла. Только команды деления и умножения требуют четырех машинных циклов. На основе этих особенностей работы устройства управления производится расчет времени исполнения прикладных программ.

На схеме микроконтроллера к устройству управления примыкает регистр команд (IR). В его функцию входит хранение кода выполняемой команды.

Входные и выходные сигналы устройства управления и синхронизации:

* PSEN – разрешение программной памяти,
* ALE – выходной сигнал разрешения фиксации адреса,
* PROG – сигнал программирования,
* EA – блокировка работы с внутренней памятью,
* VPP – напряжение программирования,
* RST – сигнал общего сброса,

* VPD – вывод резервного питания памяти от внешнего источника,

* XTAL – входы подключения кварцевого резонатора.

2.7. Параллельные порты ввода/вывода информации

Все четыре порта (P0-P3) предназначены для ввода или вывода информации побайтно. Каждый порт содержит 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. При этом через порт 0 в режиме временного мультиплексирования сначала выводится младший байт адреса, 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 табл. 4. Эти функции могут быть задействованы путем записи 1 в соответствующие биты регистра-защелки (P3.0-P3.7) порта 3.

Порт 0 является двунаправленным, а порты 1-3 – квазидвунаправленными. Каждая линия портов может быть использована независимо для ввода или вывода. По сигналу RST 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 система имеет внешнюю память, связь с которой организуется через общую разделяемую шину адреса/данных, работающую в режиме временного мультиплексирования.

Обращение к портам ввода/вывода возможно с использованием команд, оперирующих с байтом, отдельным битом, произвольной комбинацией битов. 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».

Таблица 4

Альтернативные функции порта P3

Символ	Разряд	Имя и назначение
RD	P3.7	Чтение. Активный сигнал низкого уровня формируется аппаратно при обращении к внешней памяти данных
WR	P3.6	Запись. Активный сигнал низкого уровня формируется аппаратно при обращении к внешней памяти данных
T1	P3.5	Вход таймера/счётчика 1 или тест-вход
T0	P3.4	Вход таймера/счётчика 0 или тест-вход
INT1	P3.3	Вход запроса прерывания 1. Воспринимается сигнал низкого уровня или срез
INT0	P3.2	Вход запроса прерывания 0. Воспринимается сигнал низкого уровня или срез
TXD	P3.1	Выход передатчика последовательного порта в режиме UART. Выход синхронизации в режиме регистра сдвига
RXD	P3.0	Вход приёмника последовательного порта в режиме UART. Ввод/вывод данных в режиме регистра сдвига

Этот режим обращения предполагает ввод сигналов не с внешних выводов порта, а из его регистра защелки, что позволяет исключить неправильное считывание ранее выведенной информации. Этот механизм обращения к портам реализован в командах:

ANL – логическое И, например ANL P1,A;

ORL – логическое ИЛИ, например ORL P2;

XRL – исключающее ИЛИ, например XRL;

JBC – переход, если в адресуемом бите единица, например JBC P1.1, LABEL;

CPL – инверсия бита, например CPL P3.3;

INC – инкремент порта, например INC P2;

DEC – декремент порта, например, DEC P2;

DJNZ – декремент порта и переход, например DJNZ r, LABEL;

MOV PX.Y,C – передача бита переноса в бит C;

SET PX.Y – установка бита Y порта X;

CLR PX.Y – сброс бита Y порта X.

2.8. Таймер/счетчик

В составе микроконтроллера имеются регистровые пары с символическими именами TH0, TL0 и TH1, TL1, на основе которых функционируют два независимых программно-управляемых 16-битных таймера/счетчика событий (T/C0 и T/C1). При работе в качестве таймера содержимое T/C инкрементируется в каждом машинном цикле, т. е. через каждые 12 периодов резонатора. При работе в качестве счетчика содержимое T/C инкрементируется под воздействием перехода из 1 в 0 внешнего входного сигнала, подаваемого на соответствующий (T0, T1) вход микроконтроллера. Опрос сигналов выполняется в каждом машинном цикле. Так как на распознавание перехода требуется два машинных цикла, то максимальная частота подсчета входных сигналов равна 1/24 частоты резонатора. На длительность периода входных сигналов ограничений сверху нет. Для гарантированного прочтения входного считываемого сигнала он должен удерживать значение 1, как минимум, в течение одного машинного цикла.

Для управления режимами работы и для организации взаимодействия таймеров с системой прерывания используются два регистра специальных функций TMOD и TCON, описание которых приводится в табл. 5-7. Для обоих T/C режимы работы 0, 1 и 2 одинаковы. Режим 3 для T/C0 и T/C1 различен.

Режим 0.

Перевод любого T/C в этот режим делает его 8-разрядным таймером, 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. Входной синхросигнал таймера 1 разрешен (поступает на вход T/C), когда управляющий бит TR1 установлен в 1 и либо управляющий бит GATE (блокировка) равен 0, либо на внешний вход запроса прерывания INT1 поступает уровень 1.

Установка бита GATE в 1 позволяет использовать таймер для измерения длительности импульсного сигнала, подаваемого на вход запроса прерывания.

Таблица 5

Регистр режима работы таймера/счётчика

Символ	Разряд	Имя и назначение
GATE	TMOD.7 для T/C1 TMOD.3 для T/C0	Управление блокировкой. Если бит установлен, то таймер/счётчик "x" разрешен до тех пор, пока на входе "INT x" высокий уровень и бит управления "TRx" установлен. Если бит сброшен, то T/C разрешается, как только бит управления "TRx" устанавливается
C/T	TMOD.6 для T/C1 TMOD.2 для T/C0	Бит выбора режима таймера или счётчика событий. Если бит сброшен, то работает таймер от внутреннего источника сигналов синхронизации. Если бит установлен, то работает счётчик от внешних сигналов на входе "Tx"
M1	TMOD.5 для T/C1 TMOD.1 для T/C0	Режим работы (см. табл. 6)
M0	TMOD.4 для T/C1 TMOD.0 для T/C0	

Таблица 6

Режимы работы таймера/счётчика

M1	M0	Режим работы
0	0	"TLx" работает как 5-битный предделитель
0	1	16-битный таймер/счётчик. "THx" и "TLx" включены последовательно
1	0	8-битный автоперезагружаемый таймер/счётчик. "THx" хранит значение, которое должно быть перезагружено в "TLx" каждый раз по переполнению
1	1	Таймер/счётчик 1 останавливается. Таймер/счётчик 0: TL0 работает как 8-битный таймер/счётчик, и его режим определяется управляющими битами таймера 0. TH0 работает только как 8-битный таймер, и его режим определяется управляющими битами таймера 1

Таблица 7

Регистр управления/статуса таймера

Символ	Разряд	Имя и назначение

Режим 1.

Работа любого Т/С в этом режиме такая же, как и в режиме 0, за исключением того, что таймерный регистр имеет разрядность 16 бит.

Режим 2.

В этом режиме 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, но и автоматически перезагружает в TL1 содержимое старшего байта (TH1) таймерного регистра, которое предварительно было задано программным путем. Перезагрузка оставляет содержимое TH1 неизменным. В режиме 2 Т/С0 и Т/С1 работают совершенно одинаково.

Режим 3.

В этом режиме Т/С0 и Т/С1 работают по-разному. Т/С1 сохраняет неизменным свое текущее содержимое. Иными словами, эффект такой же, как и при

сбросе управляющего бита TR1 в нуль. В этом режиме TL0 и TH0 функционируют как два независимых 8-битных 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 микроконтроллера), определяет управляющий бит TR1. При этом TH0 использует флаг переполнения TF1.

Режим 3 используется в тех случаях, когда требуется наличие дополнительного 8-битного таймера или счетчика событий. Можно считать, что в режиме 3 микроконтроллер имеет в своем составе три таймера/счетчика.

В том случае, если T/C0 используется в режиме 3, T/C1 может быть или включен, или выключен, или переведен в свой собственный режим 3, или может быть использован последовательным портом в качестве генератора частоты передачи, или, наконец, может быть использован в любом применении, не требующем прерывания.

2.9. Последовательный порт

Универсальный асинхронный приемопередатчик UART (Universal

Asynchronous Receiver-Transmitter) педназначен для передачи информации в последовательном коде. В состав UART, называемого часто последовательным портом, входят принимающий и передающий сдвигающие регистры, а также специальный буферный регистр (SBUF) приемопередатчика.

2.9.1. Регистр 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. 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, то он будет потерян.

Последовательный порт может работать в четырех различных режимах.

Режим 0.

Информация передается и принимается через вход приемника

RXD. Принимаются и передаются 8 бит данных. Через внешний выход передатчика TXD выдаются импульсы сдвига, которые сопровождают каждый бит. Частота передачи равна 1/12 частоты резонатора.

Режим 1.

Через TXD передаются или из RXD принимаются 10 бит: старт-

бит (0), 8 бит данных и стоп-бит (1). Скорость приема/передачи – величина переменная и задается таймером.

Режим 2.

Через TXD передаются или из RXD принимаются 11 бит: старт-бит, 8 бит данных, программируемый девятый бит и стоп-бит. При передаче девятый бит может использоваться для повышения достоверности передачи путем контроля по четности и в него можно поместить значение признака паритета из PSW. Частота приема/передачи выбирается программно и может быть равна 1/32 или 1/64 частоты резонатора в зависимости от SMOD.

Режим 3.

Совпадает с режимом 2, но частота приема/передачи является величиной переменной и задается таймером.

2.9.2. Регистр SCON

Регистр предназначен для управления режимом работы UART. Регистр содержит управляющие биты и девятый бит принимаемых или передаваемых

данных RB8 и TB8, а также биты прерывания приемопередатчика RI и TI. Функциональное назначение битов указано в табл. 8 и 9.

Таблица 8

Регистр управления/статуса UART

Символ	Разряд	Имя и назначение
SM0	SCON.7	Биты управления режимом работы UART. Устанавливаются/сбрасываются программно (табл. 9)
SM1	SCON.6	
SM2	SCON.5	Бит управления режимом UART. Устанавливается программно для запрета приёма сообщения, в котором девятый бит равен 0
REN	SCON.4	Бит разрешения приёма. Устанавливается/сбрасывается программно для разрешения/запрета приёма последовательных данных
TB8	SCON.3	Передача бита 8. Устанавливается/сбрасывается программно для задания девятого передаваемого бита в режиме UART - 9 бит
RB8	SCON.2	Приём бита 8. Устанавливается/сбрасывается аппаратно для фиксации девятого принимаемого бита в режиме UART - 9 бит
TI	SCON.1	Флаг прерывания передатчика. Устанавливается аппаратно при окончании передачи байта. Сбрасывается программно после обслуживания прерывания
RI	SCON.0	Флаг прерывания приёмника. Устанавливается аппаратно при приёме байта. Сбрасывается программно после обслуживания прерывания

Таблица 9

Режим работы UART

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

Прикладная программа путем загрузки в два старших разряда SCON определяет режим работы UART. Во всех режимах передача инициируется любой командой, где SBUF указан как получатель байта. Прием в UART в режиме 0 происходит при условии RI=0 и REN=1. В режимах 1 – 3 прием начинается с приходом старт-бита, если REN=1.

В TB8 программно устанавливается значение девятого бита данных, который будет передан в режиме 2 или 3. В 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 RB8 не используется.

Флаг прерывания передатчика TI устанавливается аппаратно в конце периода передачи восьмого бита данных в режиме 0 и в начале периода передачи стоп бита в режимах 1-3. Подпрограмма обслуживания этого прерывания должна сбрасывать бит TI.

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

2.9.3. Работа UART в мульти контроллерных системах

В системах децентрализованного управления, которые используются для управления и регулирования в распределенных объектах, возникает задача обмена информацией между множеством микроконтроллеров, объединенных в локальную вычислительно-управляющую сеть. Как правило, локальные сети на основе Intel 8051 имеют магистральную архитектуру с разделяемым моноканалом (коаксиальный кабель, витая пара, оптическое волокно), по которому осуществляется обмен информацией между контроллерами.

Бит SM2 в SCON позволяет простыми средствами реализовать меж – контроллерный обмен. Механизм обмена построен на том, что в режимах 2 и 3 программируемый девятый бит данных при приеме фиксируется в бите RB8.

UART может быть запрограммирован таким образом, что при получении стоп-бита прерывание от приемника будет возможно только при условии RB8=1. Ведущий контроллер всем ведомым передает широковещательное сообщение с байтом – идентификатором абонента, которое отличается от байтов данных только тем, что в его девятом бите содержится 1. Ведомые по этому признаку вызывают подпрограммы сравнения байта – идентификатора с кодом собственного сетевого адреса.

Адресуемый контроллер сбрасывает свой SM2 и готовится к приему блока данных. Остальные ведомые микроконтроллеры оставляют неизменными свои SM2=1 и передают управление основной программе. При SM2=1 информационные байты в сети прерывания не вызывают.

В режиме 1 автономного микроконтроллера SM2 используется для контроля истинности стоп-бита. В режиме 0 SM2 не используется и должен быть сброшен.

2.9.4. Скорость приема/передачи

Скорость зависит от режима работы UART. В режиме 0 частота зависит только от резонатора: ***fo =fрез./ 12.*** За один машинный цикл передается один бит.

В режимах 1 – 3 скорость зависит от значения управляющего бита SMOD в регистре специальных функций PCON (табл. 10).

В режиме 2 частота передачи $f_2 = (2^{\text{SMOD}} / 64) f_{\text{рез.}}$.

В режимах 1 и 3 в формировании частоты передачи кроме управляющего бита SMOD принимает участие таймер 1. При этом частота передачи зависит от частоты переполнения (OVT1) и определяется следующим образом: $f_{1,3} = (2^{\text{SMOD}} / 32) f_{\text{OVT1}}$.

Прерывание от таймера 1 в этом случае должно быть заблокировано. Сам Т/С1 может работать и как таймер, и как счетчик событий в любом из трех режимов. Однако наиболее удобно использовать режим таймера с авто – перезагрузкой (старшая тетрада TMOD=0010B). При этом частота передачи определяется выражением $f_{1,3} = (2^{\text{SMOD}} / 32)(f_{\text{рез.}} / 12)(256 - (\text{TH1}))$

В табл. 11 приводится описание способов настройки T/C1 для получения типовых частот передачи данных через UART.

Таблица 10

Регистр управления мощностью PCON

Символ	Разряд	Наименование и функция
SMOD	PCON.7	Удвоенная скорость передачи. Если бит установлен в 1, то скорость передачи вдвое больше, чем при SMOD=0
-	PCON.6-4	Не используются
GF1	PCON.3	Флаги, специфицируемые пользователем (флаги общего назначения)
GF0	PCON.2	
PD	PCON.1	Бит пониженной мощности. При установке в 1 микроконтроллер переходит в режим пониженного энергопотребления
IDL	PCON.0	Бит холостого хода. Если бит установлен в 1, то микроконтроллер переходит в режим холостого хода

Примечание. *При одновременной записи 1 в PD и IDL бит PD имеет преимущество. Сброс PCON выполняется путем загрузки в него кода 0XXX0000.*

Таблица 11

Настройка таймера 1 для управления частотой работы UART

Частота приёма/ передачи (BAUD RATE)		Частота резонатора, МГц	SMOD	Таймер/счётчик 1		
				C/T	Режим (MODE)	Перезагружаемое число
Режим 0, макс.:	1 МГц	12	X	X	X	X
Режим 2, макс.:	375 кГц	12	1	X	X	X
Режимы 1,3:	62,5 кГц	12	1	0	2	0FFH
	19,2 кГц	11,059	1	0	2	0FDH
	9,6 кГц	11,059	0	0	2	0FDH
	4,8 кГц	11,059	0	0	2	0FAH
	2,4 кГц	11,059	0	0	2	0F4H
	1,2 кГц	11,059	0	0	2	0E8H
	137,5 Гц	11,059	0	0	2	1DH
	110 Гц	6	0	0	2	72H
	110 Гц	12	0	0	1	0FEEBH

2.10. Система прерываний

Внешние прерывания INT0 и INT1 (рис. 3) могут быть вызваны уровнем или переходом сигнала из 1 в 0 на входах микроконтроллера в зависимости от значе-

ний управляющих битов 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 аппаратно 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.

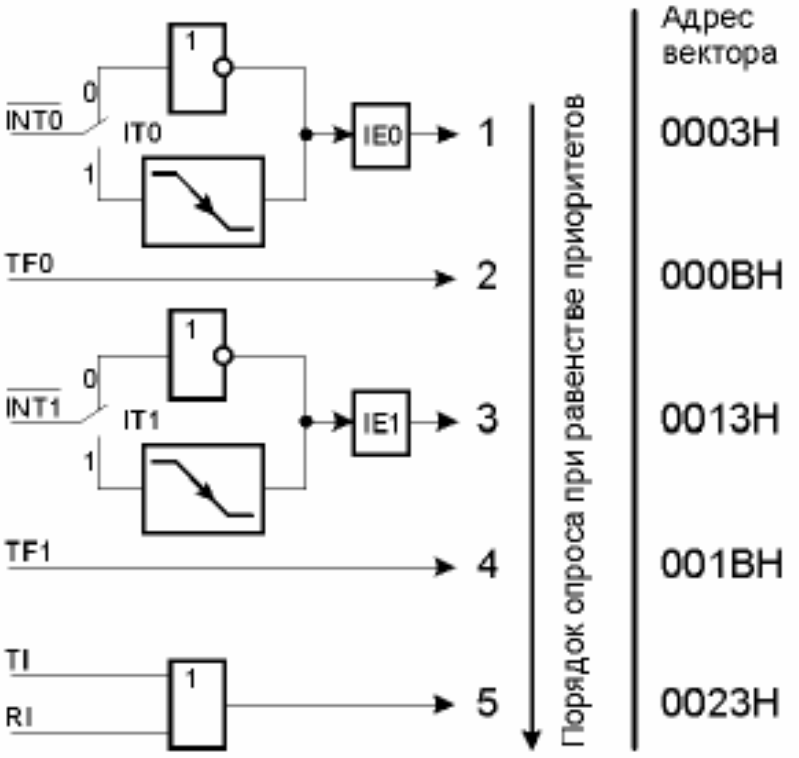

Рис. 3. Схема прерываний.

Флаги запросов прерывания от таймеров TF0 и TF1 сбрасываются автоматически при передаче управления подпрограмме обслуживания. Флаги запросов прерывания RI и TI устанавливаются UART аппаратно, но сбрасываться должны программой. Прерывания могут быть вызваны или отменены программой, так как все перечисленные флаги программно доступны. В блоке регистров специальных функций есть два регистра, предназначенных для управления ре-

жимом прерываний и уровнями приоритета. Форматы этих регистров, имеющих символические имена IE и IP, описаны в табл. 12 и 13 соответственно.

Таблица 12

Регистр масок прерывания IE

Символ	Разряд	Имя и назначение
EA	IE.7	Снятие блокировки прерываний. Сбрасывается программно для запрета всех прерываний независимо от состояний IE4-IE0
-	IE.6, 5	Не используются
ES	IE.4	Бит разрешения прерывания от UART. Установка/сброс программой для разрешения/запрета прерываний от флагов TI, 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. Установка/сброс программой для разрешения/запрета прерываний
ET0	IE.1	Разрешение прерывания от таймера 0. Работает аналогично IE.3
EX0	IE.0	Разрешения внешнего прерывания 0. Работает аналогично IE.2

Таблица 13

Регистр приоритетов прерывания IP

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

Возможность программной установки/сброса любого управляющего бита в этих двух регистрах делает систему прерываний исключительно гибкой.

Флаги прерываний опрашиваются в каждом машинном цикле. Ранжирование прерываний по приоритету выполняется в течение следующего машинного цикла.

Система прерываний сформирует аппаратно вызов LCALL соответствующей подпрограммы обслуживания, если она не заблокирована одним из условий:

1) в данный момент обслуживается запрос прерывания равного или более высокого уровня приоритета;

2) текущий машинный цикл – не последний в цикле выполняемой команды;

3) выполняется команда RETI или любая команда, связанная с обращением к регистрам IE или IP.

Примечание. Если флаг прерывания был установлен, но по одному из перечисленных условий не получил обслуживания и к моменту окончания блокировки уже был сброшен, то запрос прерывания теряется.

По аппаратно сформированному коду команды LCALL система прерывания помещает в стек содержимое программного счетчика PC и загружает в PC адрес вектора прерывания соответствующей подпрограммы обслуживания. По этому адресу должна быть расположена команда безусловного перехода JMP к начальному адресу подпрограммы обслуживания прерывания. Эта подпрограмма в случае необходимости должна начинаться командами записи в стек PUSH слова состояния программы PSW, аккумулятора A, расширителя аккумулятора B, указателя данных DPTR и т.д. и заканчиваться командами восстановления из стека POP. Подпрограммы обслуживания прерывания обязательно завершаются командой RETI, по которой в программный счетчик перезагружается из стека сохраненный адрес возврата в основную программу. Команда RET также возвращает управление, но при этом не снимает блокировку прерывания.

2.11. Контрольные вопросы

1. Перечислите характерные черты архитектуры однокристальных микроконтроллеров.

2. Укажите программно-доступные узлы К1816ВЕ51 и назначение регистров специальных функций.

3. Дайте определение понятию «булев» процессор.

4. Назовите и охарактеризуйте четыре типа информационных объектов, с которыми может оперировать арифметико-логическое устройство микроконтроллера.

5. Какова емкость адресуемой памяти программ и данных в К1816ВЕ51?

6. Какой регистр выполняет функции базового регистра при косвенных переходах в программе?

7. Какие операции могут быть выполнены только с использованием аккумулятора?

8. Какие операции могут быть выполнены без участия аккумулятора?

9. Какой формат имеет слово состояния программы К1816ВЕ51? Укажите назначение флагов.

10. Какие возможности предоставляет наличие нескольких банков регистров общего назначения?

11. Как переключить банк регистров общего назначения?

12. Какой регистр используется для адресации внешней памяти данных?

13. Как совместить адресные пространства памяти программ и данных?

14. Охарактеризуйте способ адресации элементов стека в микроконтроллере.

15. Какова длительность исполнения команд в микроконтроллере?

16. Охарактеризуйте режимы работы таймера/счетчика в К1816ВЕ51.

17. Как с помощью таймера можно измерить длительность импульса?

18. Как выводится адрес внешней памяти?

19. Какова нагрузочная способность портов?

20. Перечислите альтернативные функции портов.

21. Охарактеризуйте режимы работы последовательного порта в К1816ВЕ51.

22. Как изменить скорость передачи данных через последовательный порт?

23. Для чего используется девятый бит при передаче данных через последовательный порт?

24. Нарисуйте схему прерываний в К1816ВЕ51. Перечислите и охарактеризуйте типы прерываний.

25. Для чего нужен регистр масок прерывания? Как изменить приоритеты прерываний?

26. Как переводится микроконтроллер в режим пониженного энергопотребления?

27. Охарактеризуйте режим загрузки и верификации программ.

28. Перечислите этапы технологии разработки программ для микроконтроллеров.

29. Укажите назначение основных модулей ProView.

30. Что такое объектный код, какие функции выполняет компоновщик?

31. Укажите основные тенденции развития микроконтроллеров.

3. СИСТЕМА КОМАНД МИКРОКОНТРОЛЛЕРОВ INTEL 8051

3.1. Общие сведения

При описании системы команд микроконтроллера используются обозначения:

Rn – один из регистров R0-R7 в банке регистров, определенного битами RS1,RS0 регистра PSW;

Ad – восьмибитный адрес ячейки внутренней памяти данных;

@Ri – косвенно адресуемая ячейка РПД через регистры R0 или R1 текущего банка регистров;

#d – восьмибитная константа, включенная в команду;

#d16 – шестнадцатибитная константа, включенная в команду;

ad16 –шестнадцатиразрядный адрес перехода в командах LCALL и LJMP;

ad11– одиннадцатиразрядный адрес перехода в командах ACALL и AJMP;

rel – восьмиразрядная константа со знаком. Определяет величину смещения в команде SJMP и всех командах условных переходов;

bit – восьмиразрядный адрес бита в РПД;

add – адрес ячейки приемника данных;

ads – адрес ячейки источника данных.

Система содержит 111 базовых команд, которые по функциональному признаку могут быть разделены на пять групп:

- команды передачи данных,
- арифметические операции,
- логические операции,
- операции с битами,
- команды передачи управления.

Большинство команд имеют формат в один или два байта и выполняются за один или два машинных цикла. При тактовой частоте 12 МГц длительность машинного цикла составляет 1 мкс. На рис. 4 показаны 13 типов команд.

Первый байт команды любых типа и формата всегда содержит код операции (КОП). Второй и третий байты содержат либо адреса операндов, либо непосредственные операнды.

В приложении приведены таблицы всей системы команд микроконтроллера КМ1816ВЕ51, в которых указаны названия команды, символьное обозначение, машинный код, длина команды в байтах (Б) и время выполнения в циклах (Ц) и тактах (Т).

3.1.1. Типы операндов

Состав операндов включает в себя операнды четырех типов: биты, 4-битные цифры, байты и 16-битные слова.

Микроконтроллер имеет 128 программно-управляемых флагов пользователя. Имеется также возможность адресации отдельных битов блока регистров специальных функций и портов. Для адресации битов используется прямой 8-битный адрес (bit). Косвенная адресация битов невозможна. Карты адресов отдельных битов представлены на рис. 5 и 6.

Четырехбитные операнды используются только при операциях обмена

SWAP и XCHD.

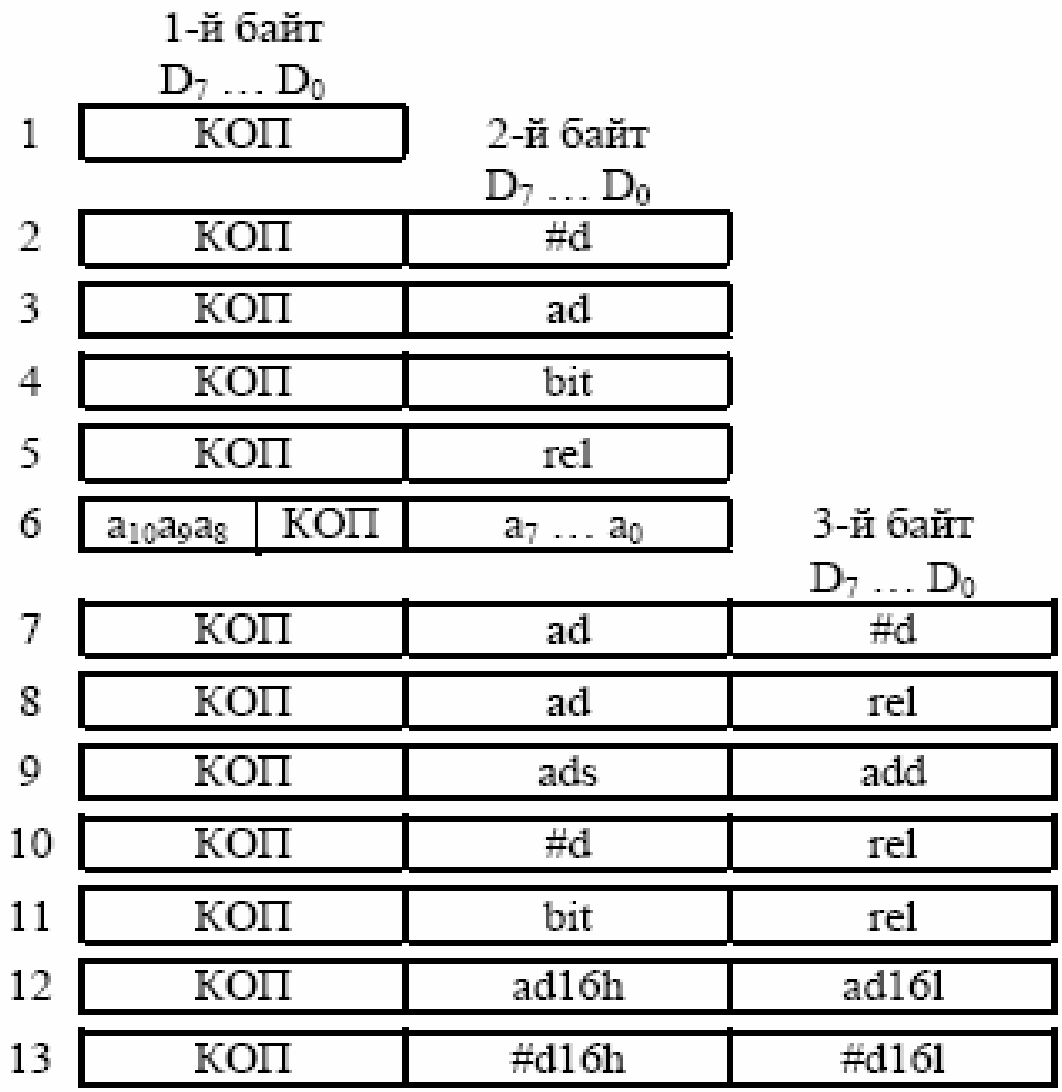

Рис. 4. Типы команд

Восьмибитным операндом может быть ячейка памяти программ (ПП) или данных (резидентной (РПД) или внешней (ВПД)), константа (непосредственный операнд), регистры специальных функций, а также порты ввода/вывода. Порты и регистры специальных функций адресуются только прямым способом. Байты памяти могут адресоваться также и косвенным образом через адресные регистры R0, R1, DPTR и PC.

Двухбайтные операнды – это константы и прямые адреса, для представления которых используются второй и третий байты команды.

3.1.2. Способы адресации данных

В семействе микроконтроллеров MCS используются следующие способы адресации данных: прямая, непосредственная, косвенная регистровая, регистровая и прямая адресация типа память-память.

При косвенном способе адресации резидентной памяти данных используются все восемь битов адресных регистров R0 и R1и шестнадцати разрядный указатель DPTR.

Адреса	(D_7)							(D_0)
7FH								
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	1F	1E	1D	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							
18H								
17H								

Рис. 5. Карта адресуемых битов в резидентной памяти данных

3.1.3. Флаги результата

Слово состояния программы PSW включает в себя четыре флага: C – перенос, AC – вспомогательный перенос, OV – переполнение и P – паритет.

Флаг паритета напрямую зависит от текущего значения аккумулятора. Если число единичных битов аккумулятора нечетное, то флаг P устанавливается, а если четное – сбрасывается. Все попытки изменить флаг P, присваивая ему новое значение, бесполезны, если содержимое аккумулятора при этом останется неизменным.

Флаг AC устанавливается, если при выполнении операции сложения или вычитания между тетрадами байта (полубайтами) возник перенос или заем.

Флаг C устанавливается, если в старшем бите результата возникает перенос или заем. При выполнении операций умножения и деления флаг C сбрасывается.

Прямой адрес бита	(D_7)							(D_0)	Имя регистра
0FFH									
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	A
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
0B8H	-	-	-	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
0A8H	AF	-	-	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
80H	87	86	85	84	83	82	81	80	P0

Рис. 6. Карта адресуемых битов в блоке регистров специальных функций

Флаг OV устанавливается, если результат операции сложения или вычитания не укладывается в семи битах и старший (восьмой) бит результата не может интерпретироваться как знаковый. При выполнении операции деления флаг OV сбрасывается, а в случае деления на нуль устанавливается. При умножении флаг OV устанавливается, если результат больше 255.

В табл. 14 перечисляются команды, при выполнении которых модифицируются флаги результата. В таблице отсутствует флаг паритета, так как его значение изменяется всеми командами, которые изменяют содержимое аккумулятора. Кроме команд, приведенных в таблице, флаги модифицируются командами, в которых местом назначения результата определены PSW или его отдельные биты, а также командами операций над битами.

Команды, модифицирующие флаги результата ***Таблица14***

Команды	Флаги	Команды	Флаги
ADD	C, OV, AC	CLR C	C = 0
ADDC	C, OV, AC	CPL C	C = NOT(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		

3.1.4. Символическая адресация

При использовании ассемблера для получения объектных кодов программ допускается применение в программах символических имен регистров специальных функций, портов и их отдельных битов (см. рис. 6).

Для адресации отдельных битов и портов (такая возможность имеется не у всех регистров специальных функций) можно использовать символическое имя бита следующей структуры: <имя регистра или порта>.<номер бита>.

Например, символическое имя пятого бита аккумулятора будет следующим: ACC.5. Символические имена являются зарезервированными словами, и их не надо определять с помощью директив ассемблера.

3.2. Команды передачи данных

Большую часть команд данной группы (табл. П.1) составляют команды передачи и обмена байтов. Команды пересылки битов представлены в группе команд битовых операций. Все команды данной группы не модифицируют флаги результата, за исключением команд загрузки PSW и аккумулятора (флаг паритета).

3.2.1. Структура информационных связей

В зависимости от способа адресации и места расположения операнда можно выделить девять типов операндов, между которыми возможен информационный обмен. Граф возможных операций передачи данных показан на рис. 7. Передачи данных могут выполняться без участия аккумулятора.

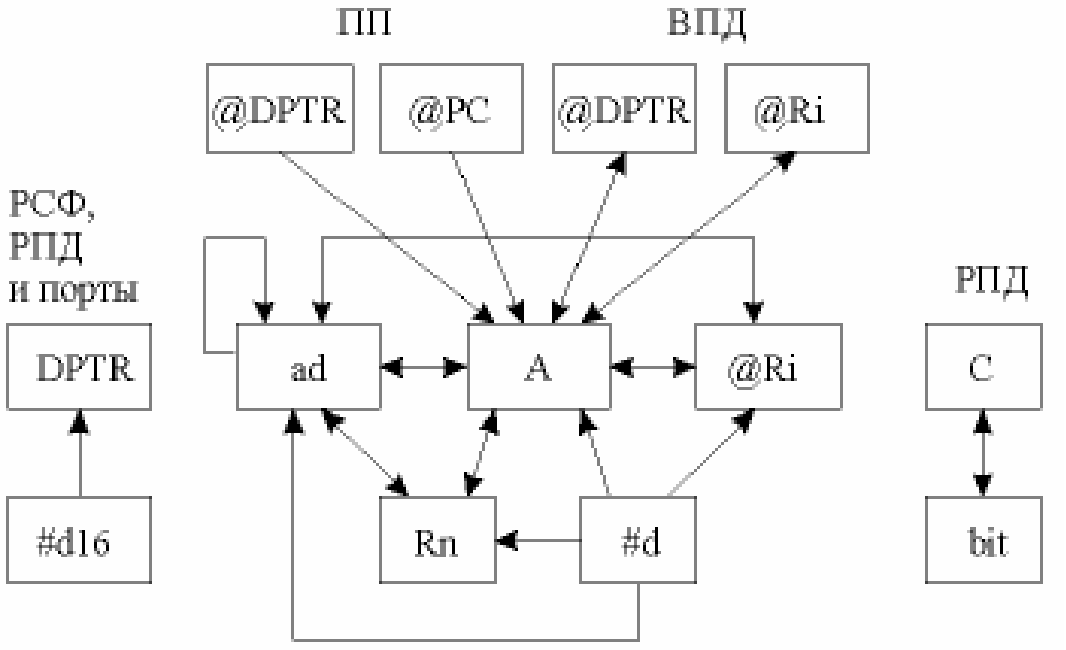

Рис. 7. Граф путей передачи данных

3.2.2. Обращение к аккумулятору

Обращение к аккумулятору может быть выполнено с использованием неявной и прямой адресации. В зависимости от способа адресации аккумулятора применяется одно из символических имен: A или ACC (прямой адрес). При прямой адресации обращение к аккумулятору производится как к одному из регистров специальных функций, и его адрес указывается во втором байте команды. Использование неявной адресации аккумулятора предпочтительнее, но не всегда возможно, например, при обращении к отдельным битам аккумулятора.

3.2.3. Обращение к внешней памяти данных

При использовании команд MOVX @Ri обеспечивается доступ к 256 байтам внешней памяти данных. Существует также режим обращения к расширенной внешней памяти данных, когда для доступа используется 16-битный адрес, хранящийся в регистре-указателе данных DPTR. Команды MOVX @DPTR обеспечивают доступ к 65536 байтам внешней памяти данных.

3.3. Арифметические операции

Данную группу образуют 24 команды (табл. П.2), выполняющие операции сложения, десятичной коррекции, инкремента/декремента байтов. Имеются команды вычитания, умножения и деления байтов.

Команды ADD и ADDC допускают сложение аккумулятора с большим числом операндов. Аналогично командам ADDC существуют четыре команды SUBB, что позволяет достаточно просто производить вычитание байтов и многобайтных двоичных чисел.

В микроконтроллере реализуется расширенный список команд инкремента/декремента байтов, команда инкремента 16-битного регистра-указателя данных.

3.4. Логические операции

Данную группу образуют 25 команд (табл. П.3), реализующих функционально полную систему логических операции над байтами. В микроконтроллере расширено число типов операндов, участвующих в операциях.

Имеется возможность производить операцию «исключающее ИЛИ» с содержимым портов. Команда XRL может быть эффективно использована для инверсии отдельных битов портов.

3.5. Команды передачи управления

К данной группе команд (табл. П.4) относятся команды, условного и безусловного ветвления, вызова подпрограмм и возврата из них, а также команда пустой операции NOP. В большинстве команд используется прямая адресация, т.е. адрес перехода целиком (или его часть) содержится в самой команде передачи управления. Можно выделить три разновидности команд ветвления по разрядности указываемого адреса перехода.

3.5.1. Длинный переход

Переход по всему адресному пространству памяти программ. В команде содержится полный 16-битный адрес перехода (ad16). Трехбайтные команды длинного перехода содержат в мнемокоде букву L (Long). Всего существует две такие команды: LJMP – длинный переход и LCALL – длинный вызов подпрограммы. На практике редко возникает необходимость перехода в пределах всего адресного пространства и чаще используются укороченные команды перехода, занимающие меньше места в памяти.

3.5.2. Абсолютный переход

Переход в пределах одной страницы памяти программ размером 2048 байтов. Такие команды содержат только 11 младших битов адреса перехода (ad11).

Команды абсолютного перехода имеют формат 2 байта. Начальная буква мнемокода – A (Absolute). При выполнении команды в вычисленном адресе сле-

дующей по порядку команды ((PC) = (PC) + 2) 11 младших битов заменяются на ad11 из тела команды абсолютного перехода.

3.5.3. Относительный переход

Короткий относительный переход позволяет передать управление в пределах от – 128 до +127 байт относительно адреса следующей команды (команды, следующей по порядку за командой относительного перехода). Существует одна команда короткого безусловного перехода SJMP (Short). Все команды условного перехода используют данный метод адресации. Относительный адрес перехода (rel) содержится во втором байте команды.

3.5.4. Косвенный переход

Команда JMP @A + DPTR позволяет передавать управление по косвенному адресу. Эта команда удобна тем, что предоставляет возможность организации перехода по адресу, вычисляемому самой программой и неизвестному при написании исходного текста программы.

3.5.5. Условные переходы

Система условных переходов предоставляет возможность осуществлять ветвление по следующим условиям: аккумулятор содержит нуль (JZ), содержимое аккумулятора не равно нулю (JNZ), перенос равен единице (JC), перенос равен нулю (JNC), адресуемый бит равен единице (JB), адресуемый бит равен нулю (JNB).

Для организации программных циклов удобно пользоваться командой DJNZ. В качестве счетчика циклов может использоваться не только регистр, но и прямо адресуемый байт (например, ячейка резидентной памяти данных).

Команда CJNE эффективно используется в процедурах ожидания какого либо-события. Например, команда WAIT: CJNE A, P0, WAIT

будет выполняться до тех пор, пока на линиях порта 0 не установится информация, совпадающая с содержимым аккумулятора.

Все команды данной группы, за исключением CJNE и JBC, не оказывают воздействия на флаги. Команда CJNE устанавливает флаг C, если первый операнд оказывается меньше второго. Команда JBC сбрасывает флаг C в случае перехода.

3.5.6. Подпрограммы

Для обращения к подпрограммам необходимо использовать команды вызова подпрограмм LCALL и ACALL. Эти команды в отличие от команд перехода LJMP и AJMP сохраняют в стеке адрес возврата в основную программу. Для возврата из подпрограммы необходимо выполнить команду RET. Команда RETI отличается от команды RET тем, что разрешает прерывания обслуженного уровня.

3.6. Операции с битами

Отличительной особенностью данной группы команд (табл. П. 5) является то, что они оперируют с однобитными операндами. В качестве таких операндов могут выступать отдельные биты некоторых регистров специальных функций и портов, а также 128 программных флагов пользователя.

Существуют команды сброса (CLR), установки (SETB) и инверсии (CPL) битов, а также конъюнкции и дизъюнкции бита и флага переноса. Для адресации битов используется прямой восьмиразрядный адрес (bit). Косвенная адресация битов невозможна.

3.7. Контрольные вопросы

1. По каким функциональным группам можно классифицировать команды микроконтроллера?

2. Какой формат может иметь команда?

3. Как длительность машинного цикла микроконтроллера соотносится с его тактовой частотой?

4. Как определить время выполнения команды?

5. С какими типами данных может оперировать микроконтроллер?

6. Для чего используются четырехбитные операнды?

7. Какие команды работают с четырехбитными операндами?

8. Для чего используются двухбайтные операнды?

9. Как косвенно адресуются байты памяти?

10. Укажите назначение флагов слова состояния программы PSW.

11. Сформулируйте условия установки флага OV.

12. Каково назначение регистров указателей?

13. Может ли порт одновременно являться источником операнда и приемником результата операции?

14. Какие способы адресации используются в микроконтроллере?

15. Можно ли адресовать порты и регистры специальных функций косвенно?

16. Приведите примеры команд передачи данных с различными способами адресации.

17. Расшифруйте команду MOVC A, @A+DPTR.

18. Приведите примеры команд доступа к 256 байтам резидентной памяти данных, к внешней памяти данных.

19. Приведите примеры логических и арифметических команд.

20. Как выполнить вычитание многобайтных операндов?

21. Перечислите команды операций с битами.

22. Как инвертировать отдельные биты портов?

23. Можно ли адресовать биты косвенно?

24. Какие переходы возможны в командах управления?

25. Для чего используются косвенные переходы в программах?

26. Поясните отличия длинного, абсолютного и относительного переходов в программах.

27. Как организовать процедуру ожидания с помощью одной команды?

28. Какие команды используются при организации подпрограмм?

29. Какие команды модифицируют флаги результата?
30. Какие регистры специальных функций допускают битовую адресацию.
31. Какие флаги используются командами условных переходов?
32. Чем отличаются команды RET и RETI?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Сташин В. В., Урусов А. В., Мологонцева О. Ф. Проектирование цифровых устройств на однокристальных микроконтроллерах. М. : Энергоатомиздат, 1990. – 224 с.
2. Микропроцессорные системы: Учебное пособие для вузов / Е. К. Александров [и др.]; Под общ. ред. Д. В. Пузанкова. – СПб.: Политехника, 2002. – 935 с.
3. Бродин В. Б., Шагурин И. И. Микроконтроллеры. Архитектура, программирование, интерфейс. – М. : Издательство ЭКОМ, 1999. – 400 с.
4. Яценков В. С. Микроконтроллеры Microchip. Практическое руководство. М. : Горячая линия – Телеком, 2002. – 296 с.
5. Евстифеев А В. Микроконтроллеры AVR семейств Tiny и Mega фирмы «ATMEL» – М. : Изд. дом «Додека-XX1», 2004. – 560 с.

ПРИЛОЖЕНИЕ

Система команд микроконтроллера К1816ВЕ51

Таблица П. 1

Команды передачи данных

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Пересылка в аккумулятор регистра (n=0..7)	MOV A,Rn	11101rrr	1	1	1	(A)<- (Rn)
Пересылка в аккумуллятор прямоадресуемого байта	MOV A,ad	11100101	3	2	1	(A)<- (ad)
Пересылка в аккумуллятор байта из РПД (i=0,1)	MOV A,@Ri	1110011i	1	1	1	(A)<- ((Ri))
Загрузка в аккумулятор константы	MOV A,#d	01110100	2	2	1	(A)<- #d
Пересылка в регистр из аккумулятора	MOV Rn,A	11111rrr	1	1	1	(Rn)<- (A)
Пересылка в регистр прямоадресуемого байта	MOV Rn,ad	10101rrr	3	2	2	(Rn)<- (ad)
Загрузка в регистр константы	MOV Rn,#d	01111rrr	2	2	1	(Rn)<- #d
Пересылка по прямому адресу аккумулятора	MOV ad,A	11110101	3	2	1	(ad)<- (A)
Пересылка по прямому адресу регистра	MOV ad,Rn	10001rrr	3	2	2	(ad)<- (Rn)
Пересылка прямоадресуемого байта по прямому адресу	MOV add,ads	10000101	9	3	2	(add)<- (ads)
Пересылка байта из РПД по прямому адресу	MOV ad,@Ri	1000011i	3	2	2	(ad)<- ((Ri))
Пересылка по прямому адресу константы	MOV ad,#d	01110101	7	3	2	(ad)<- #d
Пересылка в РПД из аккумулятора	MOV @Ri,A	1111011i	1	1	1	((Ri))<- (A)
Пересылка в РПД прямоадресуемого байта	MOV @Ri,ad	0110011i	3	2	2	((Ri))<- (ad)
Пересылка в РПД константы	MOV @Ri,#d	0111011i	2	2	1	((Ri))<- #d
Загрузка указателя данных	MOV DPTR,#d16	10010000	13	3	2	(DPTR)<-#d16

Пересылка в аккумулятор байта из ПП	MOVC A,@A+DPTR	10010011	1	1	2	(A)<- ((A)+ +(DPTR))
Пересылка в аккумулятор байта из ПП	MOVC A,@A+PC	10000011	1	1	2	(PC)<-(PC)+1 (A)<- ((A)+(PC))
Пересылка в аккумулятор байта из ВПД	MOVX A,@Ri	1110001i	1	1	2	(A)<- (Ri)
Пересылка в аккумулятор байта из расширенной ВПД	MOVX A,@DPTR	11100000	1	1	2	(A)<- ((DPTR))
Пересылка в ВПД из аккумулятора	MOVX @Ri,A	1111001i	1	1	2	((Ri))<- (A)
Пересылка в расширенную ВПД из аккумулятора	MOVX @DPTR,A	11110000	1	1	2	((DPTR))<- (A)
Загрузка в стек	PUSH ad	11000000	3	2	2	(SP)<-(SP)+1 ((SP))<- (ad)
Извлечение из стека	POP ad	11010000	3	2	2	(ad)<-(SP) (SP)<-(SP)-1
Обмен аккумулятора с регистром	XCH A,Rn	11001rrr	1	1	1	(A)↔(Rn)
Обмен аккумулятора с прямоадресуемым байтом	XCH A,ad	11000101	3	2	1	(A)↔(ad)
Обмен аккумулятора с байтом из РПД	XCH A,@Ri	1100011i	1	1	1	(A)↔((Ri))
Обмен младшей тетрады аккумулятора с младшей тетрадой байта РПД	XCHD A,@Ri	1101011i	1	1	1	$(A_{0..3})\leftrightarrow((Ri)_{0..3})$

Таблица П. 2.

Команды арифметических операций

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Сложение аккумулятора с регистром (n=0..7)	ADD A,Rn	00101rrr	1	1	1	(A)<- (A)+(Rn)
Сложение аккумулятора с прямоадресуемым байтом	ADD A,ad	00100101	3	2	1	(A)<- (A)+(ad)
Сложение аккумулятора с байтом из РПД (i=0,1)	ADD A,@Ri	0010011i	1	1	1	(A)<- (A)+((Ri))
Сложение аккумулятора с константой	ADD A,#d	00100100	2	2	1	(A)<- (A) + #d
Сложение аккумулятора с регистром и переносом	ADDC A,Rn	00111rrr	1	1	1	(A)-(A)+(Rn)+(C)
Сложение аккумулятора с прямоадресуемым байтом и переносом	ADDC A,ad	00110101	3	2	1	(A)<-(A)+(ad)+(C)
Сложение аккумулятора с байтом из РПД и переносом	ADDC A,@Ri	0011011i	1	1	1	(A)<- A)+((Ri))+(C)
Сложение аккумулятора с константой и переносом	ADDC A,#d	00110100	2	2	1	(A)<-(A)+#d+(C)
Десятичная коррекция аккумулятора	DA A	11010100	1	1	1	Если ($A_{0..3}$)>9 V ((AC)=1), то ($A_{0..3}$)<-($A_{0..3}$)+6, затем,если ($A_{4..7}$)>9 ((C)=1), то ($A_{4..7}$)<-($A_{4..7}$)+6
Вычитание из ккумулятора регистра и заема	SUBB A,Rn	10011rrr	1	1	1	(A)<-(A)-(C)-(Rn)
Вычитание из ккумулятора прямоадресуемого байта и заема	SUBB A,ad	10010101	3	2	1	(A)<-(A)-(C)-((ad))
Вычитание из ккумулятора байта РПД и заема	SUBB A,@Ri	1001011i	1	1	1	(A)<-(A)-(C)-((Ri))
Вычитание из ккумулятора константы и заема	SUBB A,d	10010100	2	2	1	(A)<-(A)-(C)-#d
Инкремент аккумулятора	INC A	00000100	1	1	1	(A)< - (A)+1
Инкремент регистра	INC Rn	00001rrr	1	1	1	(Rn)< - (Rn)+1
Инкремент прямоадресуемого байта	INC ad	00000101	3	2	1	(ad)< - (ad)+1
Инкремент байта в РПД	INC @Ri	0000011i	1	1	1	(Ri)< - (Ri)+1
Инкремент указателя данных	INC DPTR	10100011	1	1	2	(DPTR)<-(DPTR)+1
Декремент аккумулятора	DEC A	00010100	1	1	1	(A)< - (A)-1

Декремент регистра	DEC Rn	00011rrr	1	1	1	(Rn)< - (Rn)-1
Декремент прямоадресуемого байта	DEC ad	00010101	3	2	1	(ad)< - (ad)-1
Декремент байта в РПД	DEC @Ri	0001011i	1	1	1	(Ri)< - (Ri)-1
Умножение аккумулятора на регистр В	MUL AB	10100100	1	1	4	(B)(A)< - (A)*(B)
Деление аккумулятора на регистр В	DIV AB	10000100	1	1	4	(A).(B)< - (A)/(B)

Таблица П. 3.

Команды логических операций

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Логическое И аккумулятора и регистра	ANL A,Rn	01011rrr	1	1	1	(A)<-(A)and(Rn)
Логическое И аккумулятора и прямоадресуемого байта	ANL A,ad	01010101	3	2	1	(A)<-(A)and(ad)
Логическое И аккумулятора и байта из РПД	ANL A,@Ri	0101011i	1	1	1	(A)<-(A)and((Ri))
Логическое И аккумулятора и константы	ANL A,#d	01010100	2	2	1	(A)<-(A)and#d
Логическое И прямоадресуемого байта и аккумулятора	ANL ad,A	01010010	3	2	1	(ad)<-(ad)and(A)
Логическое И прямоадресуемого байта и константы	ANL ad,#d	01010011	7	3	2	(ad)<-(ad)and#d
Логическое ИЛИ аккумулятора и регистра	ORL A,Rn	01001rrr	1	1	1	(A)<-(A)V(Rn)
Логическое ИЛИ аккумулятора и прямоадресуемого байта	ORL A,ad	01000101	3	2	1	(A)<-(A)V(ad)
Логическое ИЛИ аккумулятора и байта из РПД	ORL A,@Ri	0100011i	1	1	1	(A)<-(A)V((Ri))
Логическое ИЛИ аккумулятора и константы	ORL A,#d	01000100	2	2	1	(A)<-(A)V#d
Логическое ИЛИ прямоадресуемого байта и аккумулятора	ORL ad,A	01000010	3	2	1	(ad)<-(ad)V(A)
Логическое ИЛИ прямоадресуемого байта и константы	ORL ad,#d	01000011	7	3	2	(ad)<-(ad)V#d
Исключающее ИЛИ аккумулятора и регистра	XRL A,Rn	01101rrr	1	1	1	(A)<-(A)xor(Rn)
Исключающее ИЛИ аккумулятора и прямоадресуемого байта	XRL A,ad	01100101	3	2	1	(A)<-(A)xor(ad)
Исключающее ИЛИ аккумулятора и байта из РПД	XRL A,@Ri	0110011i	1	1	1	(A)<-(A)xor((Ri))
Исключающее ИЛИ аккумулятора и константы	XRL A,#d	01100100	2	2	1	(A)<-(A)xor#d
Исключающее ИЛИ прямо-	XRL ad,A	01100010	3	2	1	(ad)<-(ad)xor(A)

адресуемого байта и аккумулятора						
Исключающее ИЛИ прямоадресуемого байта и константы	XRL ad,#d	01100011	7	3	2	(ad)<-(ad)xor#d
Сброс аккумулятора	CLR A	11100100	1	1	1	(A)<-0
Инверсия аккумулятора	CPL A	11110100	1	1	1	(A)<-not(A)
Сдвиг аккумулятора влево циклический	RL A	00100011	1	1	1	(A_{n+1})<-(A_n), n=0..6, (A_0)<-(A_7)
Сдвиг аккумулятора влево через перенос	RLC A	00110011	1	1	1	(A_{n+1})<-(A_n), n=0..6, (A_0)<-(C), (C)<-(A_7)
Сдвиг аккумулятора вправо циклический	RR A	00000011	1	1	1	(A_{n-1})<-(A_n),n=0..6, (A_7)<-(A_0)
Сдвиг аккумулятора вправо через перенос	RRC A	00010011	1	1	1	(A_{n-1})<-(A_n),n=0..6, (A_7)<-(C), (C)<-(A_0)
Обмен местами тетрад в аккумуляторе	SWAP A	11000100	1	1	1	$(A_{0..3})$<- >(A_{4-7})

Таблица П. 4.

Команды операций с битами

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Сброс переноса	CLR C	11000011	1	1	1	(C)<-0
Сброс бита	CLR bit	11000010	4	2	1	(b)<-0
Установка переноса	SETB C	11010011	1	1	1	(C)<-1
Установка бита	SETB bit	11010010	4	2	1	(b)<-1
Инверсия переноса	CPL C	10110011	1	1	1	(C)<-not(C)
Инверсия бита	CPL bit	10110010	4	2	1	(b)<-not(b)
Логическое И бита и переноса	ANL C,bit	10000010	4	2	2	(C)<-(C)and(b)
Логическое И инверсии бита и переноса	ANL C,/bit	10110000	4	2	2	(C)<-(C)andnot(b)
Логическое ИЛИ бита и переноса	ORL C,bit	01110010	4	2	2	(C)<-(C)or(b)
Логическое ИЛИ инверсии бита и переноса	ORL C,/bit	10100000	4	2	2	(C)<-(C)ornot(b)
Пересылка бита в перенос	MOV C,bit	10100010	4	2	1	(C)<-(b)
Пересылка переноса в бит	MOV bit,C	10010010	4	2	2	(b)<-(C)

Таблица П. 5.

Команды передачи управления

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Длинный переход в полном объеме памяти программ	LJMP ad16	00000010	1 2	3	2	(PC)<-ad16
Абсолютный переход внутри страницы в 2 Кбайта	AJMP ad11	$a_{10}a_9a_8$000 01	6	2	2	(PC)<-(PC)+2 ($PC_{0..10}$)<-ad11
Короткий относительный переход внутри страницы в 256 байт	SJMP rel	10000000	5	2	2	(PC)<-(PC)+2 (PC)<-(PC)+rel
Косвенный относительный переход	JMP @A+DPTR	01110011	1	1	2	(PC)<-(A)+(DPTR)
Переход, если аккумулятор равен нулю	JZ rel	01100000	5	2	2	(PC)<-(PC)+2, если (A)=0, то (PC)<-(PC)+rel
Переход, если аккумулятор не равен нулю	JNZ rel	01110000	5	2	2	(PC)<-(PC)+2, если (A)≠0, то (PC)<-(PC)+rel
Переход, если перенос равен единице	JC rel	01000000	5	2	2	(PC)<-(PC)+2, если (C)=1, то (PC)<-(PC)+rel
Переход, если перенос равен нулю	JNC rel	01010000	5	2	2	(PC)<-(PC)+2, если (C)=0, то (PC)<-(PC)+rel
Переход, если бит равен единице	JB bit,rel	00100000	1 1	3	2	(PC)<-(PC)+3, если (b)=1, то (PC)<-(PC)+rel
Переход, если бит равен нулю	JNB bit,rel	00110000	1 1	3	2	(PC)<-(PC)+3, если (b)=0, то (PC)<-(PC)+rel
Переход, если бит установлен, с последующим сбросом бита	JBC bit,rel	00010000	1 1	3	2	(PC)<-(PC)+3,если (b)=1, то (b)<-0 и (PC)< –(PC)+rel
Декремент регистра и переход, если не нуль	DJNZ Rn,rel	11011rrr	5	2	2	(PC)<-(PC)+2, (Rn)<-(Rn)-1, если Rn)≠0, то (PC)<-(PC)+rel
Декремент прямоадресуемого байта и переход, если не нуль	DJNZ ad,rel	11010101	8	3	2	(PC)<-(PC)+2, (ad)<-(ad)-1, если (ad)≠0, то (PC)<–(PC)+rel

Сравнение аккумулятора с прямоадресуемым байтом и переход, если не равно	CJNE A,ad,rel	10110101	8	3	2	(PC)<-(PC)+3, если (A)≠(ad),то (PC)<-(PC)+rel, если (A)<(ad), то (C)<-1, иначе (C)<–0
Сравнение аккумулятора с константой и переход, если не равно	CJNE A,#d,rel	10110100	10	3	2	(PC)<-(PC)+3, если (A)≠#d, то (PC)<-(PC)+rel, если (A)<#d, то (C)<-1, иначе (C)<–0
Сравнение регистра с константой и переход, если не□ра□но	CJNE Rn,#d,rel	10111rrr	10	3	2	(PC)<-(PC)+3, если (Rn)≠#d, то (PC)<-(PC)+rel, если (Rn)< #d, то (C)<-1, иначе (C)<-0
Сравнение байта в РПД с константой и переход, если не равно	CJNE @Ri,#d,rel	1011011i	10	3	2	(PC)<-(PC)+3, если ((Ri))≠#d, то (PC)<-(PC)+rel, если (Ri))< #d, то (C)<-1, иначе (C)<-0
Длинный вызов подпрограммы	LCALL ad16	00010010	12	3	2	(PC)<-(PC)+3, (SP)<-(SP)+1, ((SP))<-($PC_{0..7}$), (SP)<-(SP)+1, ((SP))<-($PC_{8..15}$), (PC)<-ad16
Абсолютный вызов подпрограммы в пределах страницы в 2 Кбайта	ACALL ad11	$a_{10}a_9a_8$10001	6	2	2	(PC)<-(PC)+2, (SP)<-(SP)+1, ((SP))<-($PC_{0..7}$), (SP)<-(SP)+1, ((SP))<-($PC_{8..15}$), ($PC_{0..10}$)<-ad11
Возврат из подпрограммы	RET	00100010	1	1	2	($PC_{8..15}$)<-((SP)), (SP)<-(SP)-1, ($PC_{0..7}$)<-((SP)), (SP)<-(SP)-1
Возврат из подпрограммы обработки прерывания	RETI	00110010	1	1	2	($PC_{8..15}$)<-((SP)), (SP)<-(SP)-1, ($PC_{0..7}$)<-((SP)), (SP)<-(SP)-1
Холостая команда	N0P	00000000	1	1	1	(PC)<-(PC)+1
Примечание. Ассемблер допускает использование обобщенного имени команд JMP и CALL, которые в процессе трансляции заменяются оптимальными по формату командами перехода (AJMP, SJMP, LJMP) или вызова (ACALL, LCALL).						

Учебное электронное текстовое издание

Мокрецов В.П.

МИКРОПРОЦЕССОРЫ И МПС

Часть 2

Микроконтроллеры

Редактор *И.Г. Южакова*
Компьютерная верстка *А.А. Гребенщикова*

Рекомендовано РИС ГОУ ВПО УГТУ-УПИ
Разрешен к публикации 18.04.07.
Электронный формат – PDF
Формат 60×90 1/8

Издательство ГОУ ВПО УГТУ-УПИ
620002, Екатеринбург, ул. Мира, 19
e-mail: sh@uchdep.ustu.ru

Информационный портал
ГОУ ВПО УГТУ-УПИ
http://www.ustu.ru