

Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра защиты информации

А. Л. Гурский, И. С. Терех

***ЦИФРОВЫЕ И МИКРОПРОЦЕССОРНЫЕ УСТРОЙСТВА СИСТЕМ
ТЕЛЕКОММУНИКАЦИЙ***

Лабораторный практикум
для студентов специальностей I-45 01 03 «Сети телекоммуникаций»,
I-98 01 02 «Защита информации в телекоммуникациях»
всех форм обучения

Минск 2008

УДК 681.325.5 – 181.48 (075.8)
ББК 32.973.26 – 018.2 я 73
Г 95

Р е ц е н з е н т

зав. кафедрой систем и устройств телекоммуникаций БГУИР,
д-р техн. наук, проф. В. К. Конопелько

Гурский, А. Л.

Г 95 Цифровые и микропроцессорные устройства систем телекоммуникаций : лаб. практикум для студ. спец. I-45 01 03 «Сети телекоммуникаций», I-98 01 02 «Защита информации в телекоммуникациях» всех форм обуч. / А. Л. Гурский, И. С. Терех. – Минск : БГУИР, 2008. – 56 с . : ил.

ISBN 978-985-488-273-4

Содержит краткие теоретические сведения по проектированию и архитектуре микропроцессоров семейства 8051, задания к лабораторным работам, список необходимой и дополнительной литературы.

УДК 681.325.5 – 181.48 (075.8)
ББК 32.973.26 – 018.2 я 73

ISBN 978-985-488-273-4

© Гурский А. Л., Терех И. С., 2008
© УО «Белорусский государственный университет информатики и радиоэлектроники», 2008

1. АРХИТЕКТУРА И СТРУКТУРА МИКРОКОНТРОЛЛЕРА (МК) 8051

Основу структурной схемы МК 8051 образует внутренняя двунаправленная 8-разрядная шина, которая связывает между собой все основные узлы и устройства: резидентную память, 8-разрядное арифметико-логическое устройство (АЛУ), устройство управления (УУ), блок регистров специальных функций и порты ввода/вывода (рис. 1.1).

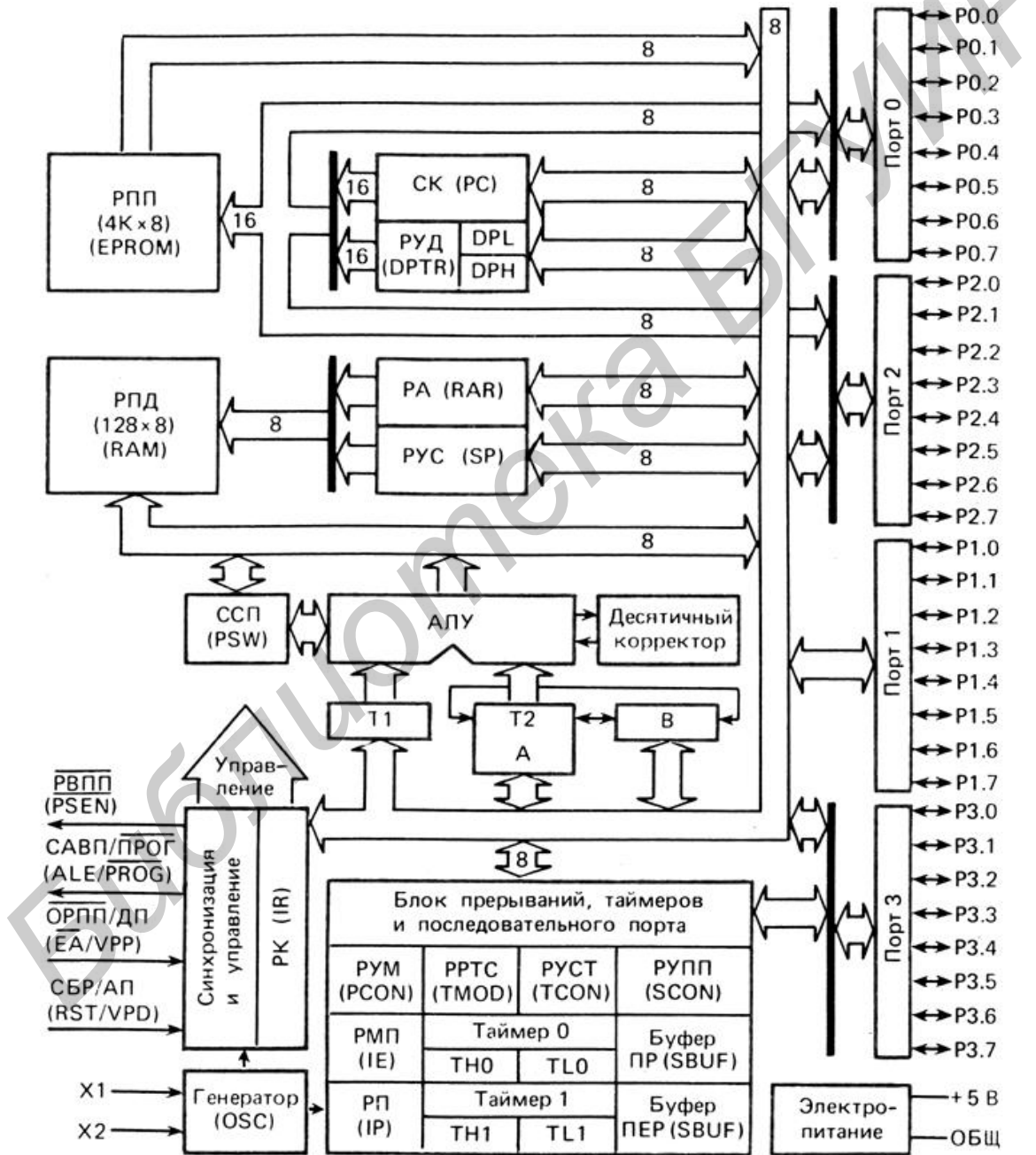


Рис. 1.1. Структурная схема МК 8051

АЛУ

8-разрядное 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 T1 и T2, 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.

АЛУ может оперировать четырьмя типами информационных объектов: булевыми (1 бит), тетрадными (4 бита), байтовыми (8 бит) и адресными (16 бит). В АЛУ выполняется 51 различная операция пересылки или преобразования этих данных.

Аккумулятор и PSW

Аккумулятор 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.

При выполнении многих команд в АЛУ формируется ряд признаков операции (флагов), которые фиксируются в регистре PSW (processor status word – слово состояния процессора). Формат регистра PSW приведен в табл. 1.1.

Устройство управления

В состав устройства управления (УУ) МК 8051 входят:

- IR – регистр команд, в котором хранится код выполняемой команды;
- OSC – встроенный генератор синхроимпульсов X1, X2;
- устройство синхронизации и управления работой МК.

Организация памяти

МК 8051 имеет Гарвардскую архитектуру: память программ и память данных, размещенные на кристалле МК 8051, физически и логически разделены, имеют различные механизмы адресации, работают под управлением

различных сигналов и выполняют разные функции. Память программ (ROM или EPROM) имеет емкость 4 Кб и предназначена для хранения команд, констант, управляющих слов инициализации, таблиц перекодировки входных и выходных переменных и т.п. Резидентная память программ (РПП) имеет 16-битную шину адреса, через которую обеспечивается доступ к памяти из счетчика команд или из регистра-указателя данных. Последний выполняет функции базового регистра при косвенных переходах по программе или используется в командах, оперирующих таблицами.

Таблица 1.1

Формат регистра PSW

Символ	Позиция	Имя и назначение
C	PSW.7	Флаг переноса. Устанавливается и сбрасывается аппаратными средствами или программой при выполнении арифметических и логических операций
AC	PSW.6	Флаг вспомогательного переноса. Устанавливается и сбрасывается только аппаратными 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	RSW.4	Выбор банка регистров. Устанавливается и сбрасывается программой для выбора рабочего банка регистров
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 команды и фиксирует нечетное/четное число единичных бит в аккумуляторе, т.е. выполняет контроль по четности

Память данных (RAM) предназначена для хранения переменных в процессе выполнения прикладной программы. Память данных (ПД) имеет емкость 128 байт.

Память МК можно представить как пять логических адресуемых пространств (или сегментов, SEG), отображаемых на трех физических устройствах памяти (внутренние ROM и RAM и внешнее ЗУ).

Пространство RSEG – пространство внутренних регистров

Пространство внутренних регистров включает 32 8-разрядных регистра, которые объединены в четыре банка по восемь регистров в каждом. Банки регистров (рис. 1.2) соответственно обозначаются RB0, RB1, RB2, RB3. В данный момент времени может быть активен только один из четырех банков. Он называется текущим.

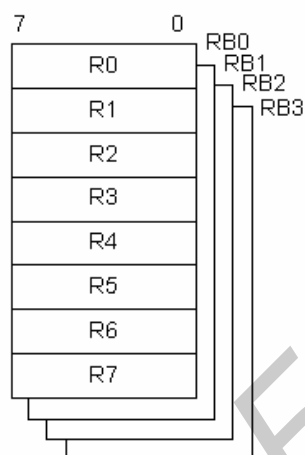


Рис. 1.2. Пространство RSEG

Для доступа к любому регистру текущего банка используется регистровая адресация, при этом номер регистра текущего банка указывается в трех младших разрядах первого байта команды – в поле Rn (КОП – код операции).



Текущий банк регистров выбирается с помощью 2-разрядного поля RS в PSW согласно табл. 1.2.

Таблица 1.2

Выбор рабочего банка регистров

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

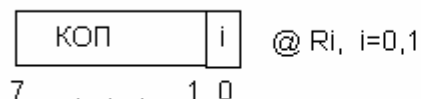
Пространство адресации данных DSEG

Пространство внутренней памяти данных имеет объем 256 байт.

Для доступа к ячейке пространства DSEG используется для способа адресации.

Первый способ – прямая адресация, при этом в команде указывается прямой 8-разрядный адрес ad. При прямой адресации доступно 128 младших байт пространства DSEG или специальные регистры. Если старший (седьмой) разряд прямого адреса равен 0, то происходит обращение к одному из специальных регистров. Если при этом используется адрес, который не приписан ни к одному специальному регистру, то результат команды не определен. В двухадресных командах используется прямой адрес источника операнда ads и прямой адрес приемника операнда add.

Второй способ – косвенная адресация через регистры R0, R1 текущего банка регистров. Признаком косвенной адресации в мнемонике команд является символ «@».



Пространство битовой адресации BSEG

Пространство BSEG представляет собой одноразрядное линейно упорядоченное пространство памяти емкостью 256 бит. В пространстве BSEG используется только прямая адресация, прямой 8-разрядный адрес в пространстве BSEG обозначается bit.

Пространство внешней памяти данных XSEG

Пространство внешней памяти данных представляет собой линейное адресное пространство объемом 64 Кб. Адрес в XSEG может принимать значение от 0000h до FFFFh.

Используются два способа адресации.

Первый способ – косвенная адресация через 16-разрядный регистр-указатель DPTR. Такая адресация возможна только в двух командах: MOVX A, @DPTR и MOVX @DPTR, A;

Второй способ – косвенная страничная адресация, при этом номер страницы задается содержимым порта P2, а смещение в странице – содержимым регистра R0 (или R1) текущего банка, т.е. адрес в XSEG находится как конкатенация (P2)*(Ri). Для этой цели можно использовать также только две команды: MOVX A, @Ri или MOVX @Ri, A.

Пространство памяти программ CSEG

Объем пространства памяти программ 64 Кб. Оно построено как однородное линейное пространство с двумя основными способами передачи управления. Первые (младшие) 4 Кб физически принадлежат EPROM внутри МК, остальные 60 Кб – реализуются при необходимости внешним запоминающим устройством.

К пространству CSEG возможно также обращение как к источнику операндов (констант). Для этого используется непосредственная адресация и косвенная адресация через регистры DPTR или PC с переменным смещением.

При непосредственной адресации операнд (константа) указывается в коде команды. Признаком непосредственной адресации в мнемонике команды является символ «#». Возможны команды с 8-разрядным операндом #d и 16-разрядным операндом #d16.

Косвенная адресация возможна только с помощью двух команд. Команда MOVC A, @A+DPTR считывает в аккумулятор байт из CSEG по адресу, равному сумме содержимого A (смещение) и DPTR. Команда MOVC A, @A+PC считывает в аккумулятор байт из CSEG по адресу, найденному как сумма содержимого A и PC.

Блок регистров специальных функций

Наименования и функции регистров этого блока указаны в табл. 1.3.

8-битный указатель стека (SP) может адресовать любую область ПД. 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. В процессе инициализации МК 8051 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 ПД с адресом 08H.

Двухбайтный регистр-указатель данных (DPTR) обычно используется для фиксации 16-битного адреса в операциях с обращением к внешней памяти. Командами МК 8051 регистр-указатель данных может быть использован или как 16-битный регистр, или как два независимых 8-битных регистра (DPH и DPL).

В составе средств МК 8051 имеются регистровые пары с символическими именами TH0, TL0 и TH1, TL1, на основе которых функционируют два независимых программно-управляемых 16-битных таймера/счетчика событий.

Таблица 1.3

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	50H
P1	Порт 1	30H
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)	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

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 (UART, универсальный асинхронно-синхронный приемопередатчик). Когда байт считывается из SBUF, это значит, что его источником является приемник последовательного порта.

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 МК 8051. Их организация будет описана ниже при рассмотрении особенностей работы МК 8051 в различных режимах.

Порты ввода–вывода

Все четыре порта МК 8051 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 выходы порта 3 могут быть использованы для реализации альтернативных функций, перечисленных в табл. 1.4. Альтернативные 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, 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, которая закрывает МОП-транзистор выходной цепи.

Таблица 1.4

Альтернативные функции порта 3

Символ	Позиция	Имя и назначение
RD	P3.7	Чтение. Активный сигнал низкого уровня формируется аппаратно при обращении к ВПД
WR	P3.6	Запись. Активный сигнал низкого уровня формируется аппаратно 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
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 сдвигающего регистра
RXD	P3.0	Вход приемника последовательного порта в режиме UART. Ввод–вывод данных в режиме сдвигающего регистра

По сигналу сброса (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 МК-система имеет внешнюю память, связь с которой организуется с помощью этих 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 (TCNT0 и TCNT1) могут быть использованы как таймеры или счетчики внешних событий. При работе в качестве таймера содержимое TCNT инкрементируется в каждом машинном цикле, т.е. через каждые 12 периодов резонатора. При работе в качестве счетчика содержимое TCNT инкрементируется под воздействием перехода из 1 в 0 внешнего входного сигнала, подаваемого на соответствующий (T0, T1) вывод МК. 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 МК.

Для управления режимами работы TCNT и для организации взаимодействия таймеров с системой прерывания используются два регистра специальных функций (TMOD и TCON), описание которых приводится в табл. 1.5 и 1.6 соответственно. Как следует из описания управляющих бит TMOD, для обоих TCNT режимы работы 0, 1 и 2 одинаковы. Режимы 3 для TCNT0 и TCNT1 различны.

Рассмотрим кратко работу таймеров/счетчиков (TCNT – от англ. timer/counter) во всех четырех режимах.

Режим 0. Перевод любого TCNT в режим 0 делает его похожим на таймер МК 8048 (8-битный счетчик), на вход которого подключен 5-битный делитель частоты на 32. В режиме 0 работы TCNT 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 TCNT), когда управляющий бит TR1 установлен в 1 и либо управляющий бит GATE (блокировка) равен 0, либо на внешний вывод 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 1.5

Регистр режима работы таймера/счетчика

Символ	Позиция	Имя и назначение
GATE	TMOD.7 для TCNT1 и TMOD.3 для TCNT0	Управление блокировкой. Если бит установлен, то таймер/счетчик «х» разрешен до тех пор, пока на входе «INTх» высокий уровень и бит управления «TRх» установлен. Если бит сброшен, то TCNT разрешается, как только бит управления «TRх» устанавливается
C/T	TMOD.6 для TCNT1 и TMOD.2 для TCNT0	Бит выбора режима таймера 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 «Tx»
M1	TMOD.5 для TCNT1 и TMOD.1 для TCNT0	Режим работы (примечание)
M0	TMOD.4 для TCNT1 и TMOD 0 для TCNT0	

Примечание.

M1	M0	Режим работы
0	0	Таймер МК 8048. «TLx» работает как 5-битный предделитель
0	1	16-битный таймер/счетчик. «THx» и «TLx» включены последовательно
1	0	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

Регистр управления/статуса таймера

Символ	Позиция	Имя и назначение
TF1	TCON.7	Флаг переполнения таймера 1. Устанавливается аппаратно при переполнении таймера/счетчика. Сбрасывается при обслуживании прерывания аппаратно
TR1	TCON.6	Бит управления таймера 1. Устанавливается/сбрасывается программой для пуска/останова
TF0	TCON.5	Флаг переполнения таймера 0. Устанавливается аппаратно. 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 аппаратно, когда детектируется срез внешнего сигнала INT1. Сбрасывается при обслуживании прерывания
IT1	TCON.2	Бит управления типом прерывания 1. Устанавливается/сбрасывается программно для спецификации запроса INT1 (срез/низкий уровень)
IE0	TCON.1	Флаг фронта прерывания 0. Устанавливается по срезу сигнала INT0. Сбрасывается при обслуживании прерывания
IT0	TCON.0	Бит управления типом прерывания 0. Устанавливается/сбрасывается программно для спецификации запроса INT0 (срез/низкий уровень)

Режим 1. Работа любого TCNT 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») регистра управления/статуса таймера 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 (табл. 1.5). Перезагрузка оставляет содержимое TH1 неизменным. В режиме 2 TCNT0 и TCNT1 работают совершенно одинаково.

Режим 3. В режиме 3 TCNT0 и TCNT1 работают по-разному. TCNT1 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 8-битных счетчика. Работу TL0 определяют управляющие биты TCNT0 (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 ТН0 использует флаг переполнения TF1. Режим 3 используется в тех случаях применения МК 8051, когда требуется наличие дополнительного 8-битного таймера или счетчика событий. Можно считать, что в режиме 3 МК 8051 имеет в своем составе три таймера/счетчика. В том случае, если TCNT0 используется в режиме 3, TCNT1 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.

Последовательный интерфейс

Через универсальный асинхронный приемопередатчик (UART) осуществляется прием и передача информации, представленной последовательным кодом (младшими битами вперед), в полном дуплексном режиме обмена (табл. 1.7). В состав UART, 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 МК 8051 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; 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.

Таблица 1.7

Регистр SCON управления/статуса UART

Символ	Позиция	Имя и назначение
SM0 SM1	SCON.7 SCON.6	Биты управления режимом работы UART. Устанавливаются / сбрасываются программно (примечание)
SM2	SCON.5	Бит управления режимом UART. 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 UART-9 бит
RB8	SCON.2	Прием 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 приемника. Устанавливается аппаратно при приеме байта. Сбрасывается программно после обслуживания прерывания

Примечание.

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

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

Управление режимом работы UART 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). Функциональное назначение бит регистра управления/статуса UART приводится в табл. 1.7.

Прикладная программа путем загрузки в старшие биты спецрегистра SCON 2-битного кода определяет режим работы UART. Во всех четырех режимах работы передача из UART инициируется любой командой, в которой буферный регистр SBUF указан как получатель байта. Прием в UART в режиме 0 осуществляется при условии, что $R1 = 0$ 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 T1 устанавливается аппаратно 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 T1. Флаг прерывания приемника R1 устанавливается аппаратно 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 R1.

Скорость приема/передачи, т.е. частота работы UART в различных режимах, определяется различными способами. В режиме 0 частота передачи зависит только от резонансной частоты кварцевого резонатора $f_{рез}$: $f_0 = f_{рез} / 12$. За один машинный цикл последовательный порт передает 1 бит информации.

В режимах 1, 2 и 3 скорость приема/передачи зависит от значения управляющего бита SMOD в регистре специальных функций PCON (табл. 1.8).

В режиме 2 частота передачи определяется выражением $f_2 = (2^{SMOD}/64) * f_{рез}$. Иными словами, при $SMOD = 0$ частота передачи равна $(1/64) f_{рез}$, а при $SMOD = 1$ равна $(1/32) f_{рез}$.

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: $f_{1,3} = (2^{SMOD}/32) * f_{OVT1}$. Прерывание от таймера 1 в этом случае должно быть заблокировано. Сам TCNT1 может работать и как таймер, и как счетчик событий в любом из трех режимов. Однако наиболее удобно использовать режим таймера с автоперезагрузкой (старшая тетрада TMOD = 0010B). При этом частота передачи определяется выражением $f_{1,3} = (2^{SMOD}/32) (f_{рез} / 2) (256 - (TH1))$. В табл. 1.9 приводится описание способов настройки TCNT1 для получения типовых частот передачи данных через UART.

Таблица 1.8

Регистр управления мощностью PCON

Символ	Позиция	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 PCON.5 PCON.4	Не используются
GF1 GF0	PCON.3 PCON.2	Флаги, специфицируемые пользователем (флаги общего назначения)
PD	PCON.1	Бит пониженной мощности (если бит установлен в 1)
IDL	PCON.0	Бит холостого хода (если бит установлен в 1)

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

Таблица 1.9

Настройка таймера 1 для управления частотой работы UART

Частота приема/передачи (BAUD RATE)	Частота резонатора, МГц	SMOD	С/Т	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	0FEEDH

Система прерываний

Упрощенная схема прерываний МК 8051 показана в табл. 1.10. Внешние прерывания INT0 и INT1 могут быть вызваны либо уровнем, либо переходом сигнала из 1 в 0 на входах МК 8051 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 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.

Таблица 1.10

Схема прерываний МК 8051

Прерывание	Порядок опроса при равных приоритетах	Адрес вектора	Примечание
INT0	1	0003H	JMP(AJMP,SJMP,LJMP...) RETI
TF0	2	000BH	JMP(AJMP,SJMP,LJMP...) RETI
INT1	3	0013H	JMP(AJMP,SJMP,LJMP...) RETI
TF1	4	001BH	JMP(AJMP,SJMP,LJMP...) RETI
UART (TI,RI)	5	0023H	JMP(AJMP,SJMP,LJMP...) RETI

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

Прерывания могут быть вызваны или отменены программой, так как все перечисленные флаги программно доступны и могут быть установлены/сброшены программой с тем же результатом, как если бы они были установлены/сброшены аппаратными средствами.

В блоке регистров специальных функций есть два регистра, предназначенных для управления режимом прерываний и уровнями приоритета. Форматы этих регистров, имеющих символические имена IE и IP, описаны в табл. 1.11 и 1.12 соответственно. Возможность программной установки/сброса любого управляющего бита в этих двух регистрах делает систему прерываний МК51 исключительно гибкой.

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

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

текущий машинный цикл – не последний в цикле выполняемой команды; выполняется команда RETI или любая команда, связанная с обращением к регистрам IE или IP.

Таблица 1.11

Регистр масок прерывания (IE)

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

Таблица 1.12

Регистр приоритетов прерываний (IP)

Символ	Позиция	Имя и назначение
-	IP.7– IP.5	Не используются
PS	IP.4	Бит приоритета UART. Установка/сброс программой для присваивания прерыванию от UART 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

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

По аппаратно-сформированному коду LCALL система прерывания помещает в стек только содержимое счетчика команд (PC) и загружает в счетчик команд адрес вектора соответствующей подпрограммы обслуживания. По адресу вектора должна быть расположена команда безусловной передачи управления (JMP) к начальному адресу подпрограммы обслуживания прерывания. Подпрограмма обслуживания в случае необходимости должна начинаться командами записи в стек (PUSH) слова состояния программы (PSW), аккумулятора, расширителя, указателя данных и т.д. и заканчиваться командами восстановления из стека (POP). Подпрограммы обслуживания прерывания обязательно завершаются командой RETI, по которой в счетчик команд перезагружается из стека сохраненный адрес возврата в основную программу. Команда RET также возвращает управление прерванной основной программе, но при этом не снимает блокировку прерываний, что приводит к необходимости иметь программный механизм анализа окончания процедуры обслуживания данного прерывания.

Сброс, режим холостого хода и режим пониженного энергопотребления

Сброс МК 8051 осуществляется путем подачи на вход RST сигнала 1. Для уверенного сброса этот сигнал должен быть удержан на входе RST по меньшей мере в течение двух машинных циклов (24 периода резонатора). Квазидвухнаправленные буферные схемы внешних выводов ALE и PSEN находятся при этом в режиме ввода. Под воздействием сигнала RST сбрасывается содержимое регистров: PC, ACC, B, PSW, DPTR, TMOD, TCON, T/C0, T/C1, IE, IP и SCON, в регистре PCON сбрасывается только старший бит, в регистр-указатель стека загружается код 07H, а в порты P0–P3 – коды 0FFH. Состояние регистра SBUF неопределенное. Сигнал RST не воздействует на содержимое ячеек ПД. Когда включается электропитание (VCC), содержимое ПД неопределенно, за исключением операции возврата из режима пониженного энергопотребления.

Любая команда, по которой установится управляющий бит IDL(PCON.0), в регистре управления мощностью переведет МК 8051 в режим холостого хода. При этом продолжает работу внутренний генератор синхросигналов. Все регистры сохраняют свое значение. На выводах всех портов удерживается то логическое состояние, которое на них было в момент перехода в режим холостого хода. На выводах ALE и PSEN формируется уровень 1. Выйти из режима холостого хода можно или по сигналу RST, или по прерыванию. Любой

из разрешенных сигналов прерывания приведет к аппаратному сбросу бита PCON.0 и прекратит тем самым режим холостого хода. После исполнения команды RETI (выход из подпрограммы обслуживания прерывания) будет исполнена команда, которая следует в программе за командой, переведшей МК 8051 в режим холостого хода.

Перевод в режим пониженного энергопотребления возможен по команде, которая установит бит PCON.1 в регистре управления мощностью (см. табл. 1.8). В этом режиме останавливается генератор синхросигналов, содержимое ПД и регистров специальных функций сохраняется, а на выходных контактах портов удерживаются значения, соответствующие содержимому их буферных регистров. Выходы сигналов ALE и PSEN сбрасываются. При этом электропитание осуществляется через вывод RST/VPD. В режиме пониженного энергопотребления напряжение электропитания (VCC) может быть отключено. Перед выходом из режима оно должно быть восстановлено до номинального значения. Выход из режима пониженного энергопотребления возможен только по сигналу RST. При этом переопределяются все регистры специальных функций, но содержимое ПД не изменяется.

2. СИСТЕМА КОМАНД МК 8051

Язык Ассемблера

Ассемблер – это программа, преобразующая последовательность мнемонических инструкций (текст программы на языке Ассемблера) в последовательность машинных кодов команд и двоичных данных, называемую объектным кодом. Именно эти коды считываются из памяти и выполняются микропроцессором или микроконтроллером.

Язык Ассемблера относится к группе машинно-ориентированных языков. Иначе говоря, каждому типу микропроцессоров или микроконтроллеров соответствует свой ассемблерный язык.

Оператором языка Ассемблера микроконтроллера 8051 является строка исходного текста микроконтроллерной программы, имеющая следующий формат:

<метка> <операция> <операнды> <;комментарий>

Поле <операция> содержит мнемоническое обозначение команды или директивы Ассемблера, которое является сокращением (аббревиатурой) полного английского наименования выполняемого действия. Например: MOV – move – переслать, JMP – jump – перейти, DB – define byte – определить байт.

Кроме того, поле <операция> может содержать символическое имя ассемблерной макрокоманды.

Поле <операнды> зависит от поля <операция> и может указывать группу разделенных запятой операндов либо может быть исключено вообще.

Операнды ассемблерных команд определяют тип используемых данных (бит, байт, 2-байтное слово), способ адресации этих данных и адреса переходов в области памяти программ микроконтроллера. Для МК 8051 различают следующие способы адресации данных: регистровый, прямой, косвенно-регистровый и непосредственный.

После поля <операнды> может присутствовать необязательное поле комментария, выделяемое вначале точкой с запятой.

Регистровая адресация обеспечивает обращение к байтовому содержимому регистров А, В или регистров R0-R7 выбранного банка, к 2-байтному содержимому регистра DPTR и к битовому содержимому флага переноса С, при этом в качестве операндов используются принятые имена перечисленных программно-доступных элементов, а также символические имена (только для регистров R0–R7), определяемые пользователем.

Прямая адресация применяется для обращения к байтовому содержимому 128 ячеек резидентной памяти данных или 21 регистра специальных функций (РСФ), а также к битовому содержимому 16 ячеек ПД или 11 РСФ, допускающих побитовое обращение. При прямой адресации данных в поле <операнды> указывается прямой адрес используемых ячейки РПД, регистра специальных функций или бита. Этот адрес может быть задан числом, символическим именем, выражением, именем (только для РСФ). Кроме того, имя бита РСФ может быть представлено символической структурой вида <имя РСФ>.<номер бита>. Например, имя пятого бита регистра TCON можно записать как TCON.5, имя второго бита аккумулятора – как A.2 и т.д.

С помощью косвенно-регистровой адресации обеспечивается обращение к байтовому содержимому 128 ячеек ПД, при этом адрес используемой ячейки определяется содержимым указателя стека SP или одного из регистров R0, R1 выбранного банка. Косвенно-регистровая адресация используется также для обращения к внешней памяти данных. В этом случае регистром-указателем может быть 16-разрядный указатель данных DPTR или один из упомянутых выше регистров R0, R1. Для работы с данными, сохраненными в виде констант в память программ микроконтроллера, применяется косвенно-регистровая адресация по сумме: базовый регистр (содержимое DPTR или программного счетчика PC) плюс индексный регистр (содержимое аккумулятора А). Любая такая константа может быть выбрана по адресу, который вычисляется сложением содержимого DPTR (PC) с содержимым А. Операнд, определяющий

косвенно-регистровую адресацию данных, задается именем регистра-указателя или символическим именем (только для R0 и R1) с обязательным префиксом @.

При непосредственной адресации данные, предназначенные для обработки, непосредственно указываются в поле <операнды> и могут быть представлены в нем числом, символическим именем или выражением с обязательным префиксом #.

Аналогичным образом (за исключением префикса #) представляется операнд, определяющий адрес перехода в памяти программ микроконтроллера.

В качестве операндов ассемблерных директив и макрокоманд обычно используются числа, символические имена, выражения, имена программно-доступных элементов микроконтроллера (только для директивы REG и макрокоманд), а в ряде случаев мнемоники языка Ассемблера (только для макрокоманд).

Символические имена, являющиеся операндами команд или директив, должны быть обязательно определены с помощью соответствующих директив (EQU, VAR или REG) языка Ассемблера. Кроме того, символическое имя адреса в памяти программ может быть определено использованием этого имени в поле <метка> одной из строк исходного текста МК-программы. Отметим, что корректное символическое имя должно быть представлено комбинацией букв латинского алфавита и цифр и начинаться с буквы, при этом указанная комбинация допускает использование символа подчеркивания.

Выражение, используемое в поле <операнды>, вычисляется в процессе трансляции исходной МК-программы и представляет собой совокупность символических имен и (или) чисел (в формате 2-байтных слов), содержащую следующие основные операторы:

- + – сложение (третий уровень приоритета);
- – вычитание (третий уровень приоритета);
- * – умножение (четвертый уровень приоритета);
- / – деление (четвертый уровень приоритета);
- ** – возведение в степень (пятый уровень приоритета);
- .OR. – ИЛИ (первый уровень приоритета);
- .AND. – И (второй уровень приоритета);
- .XOR. – исключающее ИЛИ (первый уровень приоритета);
- .NOT. – отрицание (шестой уровень приоритета);
- < (>) – выделение младшего (старшего) байта 2-байтного слова (шестой уровень приоритета).

Оператор с более высоким уровнем приоритета выполняется в первую очередь. Если в выражении присутствуют операторы с одинаковым уровнем приоритета, то вычисления производятся слева направо. Чтобы изменить

указанный порядок выполнения расчетов, допускается использовать скобки. В качестве примера приведем выражение `<.NOT.13H+1`, реализующее процедуру преобразования числа 13H в дополнительный код и эквивалентное числу 0EDH, которое будет получено при трансляции исходной МК-программы.

Поле <метка> не является обязательным, отделяется от поля <команда/директива> пробелом и может содержать символическое имя непосредственных данных, одного из регистров R0-R7, прямого адреса, макрорасширения или адреса перехода в памяти программ. Если метка заканчивается двоеточием, то она может быть расположена в любом месте строки, в противном случае метка должна начинаться в начале строки.

Поле <комментарий> содержит пояснения различного характера – может объяснять применение той или иной команды или директивы, содержать описание алгоритма участка или МК-программы в целом и др. Это поле не является обязательным и при использовании должно начинаться символом «;».

Команды МК 8051

Система команд МК 8051 содержит 111 базовых команд. Путем комбинирования «операция/режим адресации» базовое число команд 111 расширяется до 255 из 256 возможных при однобайтном коде операции. Большинство команд (94) имеют формат один или два байта и выполняются за один или два машинных цикла. При тактовой частоте 12 МГц длительность машинного цикла составляет 1 мкс.

Команды удобно разделить по функциональному признаку на пять групп: команды передачи данных, арифметических операций, логических операций, передачи управления и операций с битами.

В табл. 2.1 приведены обозначения, используемые в описании команд.

Таблица 2.1

Обозначения, используемые в описании команд

Символ	Обозначение
#d	Непосредственный операнд
ad	Адрес операнда в DSEG
bit	Адрес бита в BSEG
rel	Относительный адрес в CSEG
ad11	Адрес 11 разрядов
ad16	Адрес 16 разрядов
#d16	Непосредственный 16-разрядный операнд
ads, add	Непосредственные адреса источника и приемника в DSEG

При выполнении команд могут устанавливаться/сбрасываться флаги результата в регистре слова состояния (PSW), который включает в себя четыре флага: С – перенос, АС – вспомогательный перенос, ОV – переполнение и Р – паритет.

Флаг паритета напрямую зависит от текущего значения аккумулятора. Если число единичных бит аккумулятора нечетное, то флаг Р устанавливается, а если четное – сбрасывается. Все попытки изменить флаг Р, присваивая ему новое значение, будут безуспешными, если содержимое аккумулятора при этом останется неизменным. Флаг АС устанавливается в случае, если при выполнении операции сложения/вычитания между тетрадами байта возник перенос/заем. Флаг С устанавливается, если в старшем бите результата возникает перенос или заем. При выполнении операций умножения и деления флаг С сбрасывается. Флаг ОV (overflow) устанавливается, если результат операции сложения/вычитания не укладывается в семи битах и старший (восьмой) бит результата не может интерпретироваться как знаковый. При выполнении операции деления флаг ОV сбрасывается, а в случае деления на нуль устанавливается. При умножении флаг ОV устанавливается, если результат больше 255.

В табл. 2.2 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 МК 8051 приведены в табл. 2.3, 2.4. В них Б – число байт в команде, Ц – время выполнения в машинных циклах.

Таблица 2.2

Команды, модифицирующие флаги результата

Команды	Флаги	Команды	Флаги
ADD	С, ОV, АС	CLR C	С=0
ADDC	С, ОV, АС	CPL C	С=С
SUBB	С, ОV, АС	ANL C,b	С
MUL	С = 0, ОV	ANL C,/b	С
DIV	С = 0, ОV	ORL C,b	С
DA	С	ORL C,/b	С
RRC	С	MOV C,b	С
RLC	С	CJNE	С
SETBC	С=1		

Таблица 2.3

Группа команд передачи данных

Название команды	Мнемокод	Б	Ц	Операция
Пересылка в аккумулятор из регистра ($n = 0 - 7$)	MOV A,Rn	1	1	(A) \dot{i} (Rn)
Пересылка в аккумулятор прямоадресуемого байта	MOV A,ad	2	1	(A) \dot{i} (ad)
Пересылка в аккумулятор байта из РПД ($i = 0,1$)	MOV A,@R1	1	1	(A) \dot{i} ((R1))
Загрузка в аккумулятор константы	MOV A,#d	2	1	(A) \dot{i} (#d)
Пересылка в регистр из аккумулятора	MOV Rn,A	1	1	(Rn) \dot{i} (A)
Пересылка в регистр прямоадресуемого байта	MOV Rn,ad	2	2	(Rn) \dot{i} (ad)
Загрузка в регистр константы	MOV Rn,#d	2	1	(Rn) \dot{i} (#d)
Пересылка по прямому адресу аккумулятора	MOV ad,A	2	1	(ad) \dot{i} (A)
Пересылка по прямому адресу регистра	MOV ad,Rn	2	2	(ad) \dot{i} (Rn)
Пересылка прямоадресуемого байта по прямому адресу	MOV add,ads	3	2	(add) \dot{i} (ads)
Пересылка байта из РПД по прямому адресу	MOV ad,@Ri	2	2	(ad) \dot{i} ((Ri))
Пересылка по прямому адресу константы	MOV ad,#d	3	2	(ad) \dot{i} (#d)
Пересылка в РПД из аккумулятора	MOV @M,A	1	1	((Ri)) \dot{i} (A)
Пересылка в РПД прямоадресуемого байта	MOV @Ri,ad	2	2	((Ri)) \dot{i} (ad)
Пересылка в РПД константы	MOV @Ri,#d	2	1	((Ri)) \dot{i} (#d)
Загрузка указателя данных	MOV DPTR,#d16	3	2	(DPTR) \dot{i} (#16)
Пересылка в аккумулятор байта из ПП	MOVC A,@A + DPTR	1	2	(A) \dot{i} ((A) + DPTR)
Пересылка в аккумулятор байта из ПП	MOVC A,@A+PC	1	2	(PC) \dot{i} (PC)+1 (A) \dot{i} ((A)+(PC))
Пересылка в аккумулятор байта из ВПД	MOVX A,@Ri	1	2	(A) \dot{i} ((Ri))
Пересылка в аккумулятор байта из расширенной ВПД	MOVX A,@DPTR	1	2	(A) \dot{i} ((DPTR))
Пересылка в ВПД из аккумулятора	MOVX @Ri,A	1	2	((Ri)) \dot{i} (A)
Пересылка в расширенную ВПД из аккумулятора	MOVX @DPTR,A	1	2	((DPTR)) \dot{i} (A)
Загрузка в стек	PUSH ad	2	2	(SP) \dot{i} (SP) + 1 ((SP)) \dot{i} (ad)
Извлечение из стека	POP ad	2	2	(ad) \dot{i} (SP) (SP) \dot{i} (SP) - 1
Обмен аккумулятора с регистром	XCH A,Rn	1	1	(A) \dot{o} (Rn)
Обмен аккумулятора с прямоадресуемым байтом	XCH A, ad	2	1	(A) \dot{o} (ad)
Обмен аккумулятора с байтом из РПД	XCH A,@Ri	1	1	(A) \dot{o} ((Ri))
Обмен младшей тетрады аккумулятора с младшей тетрадой по косвенному адресу	XCHD A,@Ri	1	1	(A ₀₋₃) \dot{o} ((Ri) ₀₋₃)

Таблица 2.4

Группа команд арифметических операций

Название команды	Мнемокод	Б	Ц	Операция
Сложение аккумулятора с регистром ($n = 0 - 7$)	ADD A,Rn	1	1	$(A) \dot{\bar{i}} (A) + (Rn)$
Сложение аккумулятора с прямоадресуемым байтом	ADD A,ad	2	1	$(A) \dot{\bar{i}} (A) + (ad)$
Сложение аккумулятора с байтом из РПД ($i = 0,1$)	ADD A,@Ri	1	1	$(A) \dot{\bar{i}} (A) + ((Ri))$
Сложение аккумулятора с константой	ADD A,#d	2	1	$(A) \dot{\bar{i}} (A) + \#d$
Сложение аккумулятора с регистром и переносом	ADDC A,Rn	1	1	$(A) \dot{\bar{i}} (A) + (Rn) + (C)$
Сложение аккумулятора с прямоадресуемым байтом и переносом	ADDC A,ad	2	1	$(A) \dot{\bar{i}} (A) + (ad) + (C)$
Сложение аккумулятора с байтом из РПД и переносом	ADDC A,@Ri	1	1	$(A) \dot{\bar{i}} (A) + ((Ri)) + (C)$
Сложение аккумулятора с константой и переносом	ADDC A,#a	2	1	$(A) \dot{\bar{i}} (A) + \#d + (C)$
Десятичная коррекция аккумулятора	DA A	1	1	Если $(A_{0-3}) > 9 \dot{\bar{U}}((AC) = 1)$, то $(A_{0-3}) \dot{\bar{i}} (A_{0-3}) + 6$, затем если $(A_{4-7}) > 9 \dot{\bar{U}}((C) = 1)$, то $(A_{4-7}) \dot{\bar{i}} (A_{4-7}) + 6$
Вычитание из аккумулятора регистра и заема	SUBB A,Rn	1	1	$(A) \dot{\bar{i}} (A) - (C) - (Rn)$
Вычитание из аккумулятора прямоадресуемого байта и заема	SUBB A,ad	2	1	$(A) \dot{\bar{i}} (A) - (C) - ((ad))$
Вычитание из аккумулятора байта РПД и заема	SUBB A,@Ri	1	1	$(A) \dot{\bar{i}} (A) - (C) - ((Ri))$
Вычитание из аккумулятора константы и заема	SUBB A,d	2	1	$(A) \dot{\bar{i}} (A) - (C) - \#d$
Инкремент аккумулятора	INC A	1	1	$(A) \dot{\bar{i}} (A) + 1$
Инкремент регистра	INC Rn	1	1	$(Rn) \dot{\bar{i}} (Rn) + 1$
Инкремент прямоадресуемого байта	INC ad	2	1	$(ad) \dot{\bar{i}} (ad) + 1$
Инкремент байта в РПД	INC @Ri	1	1	$((Ri)) \dot{\bar{i}} ((Ri)) + 1$
Инкремент указателя данных	INC DPTR	1	2	$(DPTR) \dot{\bar{i}} (DPTR) + 1$
Декремент аккумулятора	DEC A	1	1	$(A) \dot{\bar{i}} (A) - 1$
Декремент регистра	DEC Rn	1	1	$(Rn) \dot{\bar{i}} (Rn) - 1$
Декремент прямоадресуемого байта	DEC ad	2	1	$(ad) \dot{\bar{i}} (ad) - 1$
Декремент байта в РПД	DEC @Ri	1	1	$((Ri)) \dot{\bar{i}} ((Ri)) - 1$
Умножение аккумулятора на регистр В	MUL AB	1	4	$(B)(A) \dot{\bar{i}} (A) * (B)$
Деление аккумулятора на регистр В	DIV AB	1	4	$(A),(B) \dot{\bar{i}} (A) / (B)$

Таблица 2.5

Группа команд логических операций

Название команды	Мнемокод	Б	Ц	Операция
Логическое И аккумулятора и регистра	ANL A,Rn	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (Rn)$
Логическое И аккумулятора и прямоадресуемого байта	ANL A,ad	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (ad)$
Логическое И аккумуля-ра и байта из ПД	ANL A,@Ri	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} ((Ri))$
Логическое И аккумулятора и константы	ANL A,#d	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} \#d$
Логическое И прямоадресуемого байта и аккумулятора	ANL ad,A	2	1	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} (A)$
Логическое И прямоадресуемого байта и константы	ANL ad,#d	3	2	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} \#d$
Логическое ИЛИ аккумуля-ра и регистра	ORL A,Rn	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (Rn)$
Логическое ИЛИ аккумулятора и прямоадресуемого байта	ORL A,ad	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (ad)$
Логическое ИЛИ аккумулятора и байта из РПД	ORL A,@Ri	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} ((Ri))$
Логическое ИЛИ аккумулятора и константы	ORL A,#d	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} \#d$
Логическое ИЛИ прямоадресуемого байта и аккумулятора	ORL ad,A	2	1	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} (A)$
Логическое ИЛИ прямоадресуемого байта и константы	ORL ad,#d	3	2	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} \#d$
Исключающее ИЛИ аккумуля-ра и регистра	XRL A,Rn	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (Rn)$
Исключающее ИЛИ аккумулятора и прямоадресуемого байта	XRL A,ad	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} (ad)$
Исключающее ИЛИ аккумулятора и байта из РПД	XRL A,@Ri	1	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} ((Ri))$
Исключающее ИЛИ аккумулятора и константы	XRL A,#d	2	1	$(A) \dot{\bar{i}} (A) \dot{\bar{U}} \#d$
Исключающее ИЛИ прямоадресуемого байта и аккумулятора	XRL ad,A	2	1	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} (A)$
Исключающее ИЛИ прямоадресуемого байта и константы	XRL ad,#d	3	2	$(ad) \dot{\bar{i}} (ad) \dot{\bar{U}} \#d$
Сброс аккумулятора	CLR A	1	1	$(A) \dot{\bar{i}} 0$
Инверсия аккумулятора	CPL A	1	1	$(A) \dot{\bar{i}} (\bar{A})$
Сдвиг аккумулятора влево циклический	RL A	1	1	$(A_{n+1})\dot{\bar{i}} (A_n), n=0-6, (A_0)\dot{\bar{i}} (A_7)$
Сдвиг аккумулятора влево через перенос	RLC A	1	1	$(A_{n+1})\dot{\bar{i}} (A_n), n=0-6, (A_0)\dot{\bar{i}} (C), (C)\dot{\bar{i}} (A_7)$
Сдвиг аккумулятора вправо циклический	RR A	1	1	$(A_n)\dot{\bar{i}} (A_{n+1}), n=0-6, (A_7)\dot{\bar{i}} (A_0)$
Сдвиг аккумулятора вправо через перенос	RRC A	1	1	$(A_n)\dot{\bar{i}} (A_{n+1}), n=0-6, (A_7)\dot{\bar{i}} (C), (C)\dot{\bar{i}} (A_0)$
Обмен местами тетрад в аккумуляторе	SWAP A	1	1	$(A_{0-3}) \dot{\bar{O}} (A_{4-7})$

Таблица 2.6

Группа команд операций с битами

Название команды	Мнемокод	Б	Ц	Операция
Сброс переноса	CLR C	1	1	(C) $\bar{0}$
Сброс бита	CLR bit	2	1	(b) $\bar{0}$
Установка переноса	SETB C	1	1	(C) $\bar{1}$
Установка бита	SETB bit	2	1	(b) $\bar{1}$
Инверсия переноса	CPL C	1	1	(C) $\bar{\bar{C}}$
Инверсия бита	CPL bit	2	1	(b) $\bar{\bar{b}}$
Логическое И бита и переноса	ANL C,bit	2	2	(C) $\bar{\bar{C}} \bar{\bar{U}}$ (b)
Логическое И инверсии бита и переноса	ANL C,/bit	2	2	(C) $\bar{\bar{C}} \bar{\bar{U}}$ ($\bar{\bar{b}}$)
Логическое ИЛИ бита и переноса	ORL C,bit	2	2	(C) $\bar{\bar{C}} \bar{\bar{U}}$ (b)
Логическое ИЛИ инверсии бита и переноса	ORL C,/bit	2	2	(C) $\bar{\bar{C}} \bar{\bar{U}}$ ($\bar{\bar{b}}$)
Пересылка бита в перенос	MOV C,bit	2	1	(C) $\bar{\bar{b}}$
Пересылка переноса в бит	MOV bit,C	2	2	(b) $\bar{\bar{C}}$

Таблица 2.7

Группа команд передачи управления

Название команды	Мнемокод	Б	Ц	Операция
1	2	3	4	5
Длинный переход в полном объеме памяти программ	LJMP ad16	3	2	(PC) $\bar{\bar{ad16}}$
Абсолютный переход внутри страницы в 2 Кб	AJMP ad11	2	2	(PC) $\bar{\bar{(PC)+2}}$ (PC ₀₋₁₀) $\bar{\bar{ad11}}$
Короткий относительный переход внутри страницы в 256 байт	SJMP rel	2	2	(PC) $\bar{\bar{(PC)+2}}$ (PC) $\bar{\bar{(PC) + rel}}$
Косвенный относительный переход	JMP @A+DPTR	1	2	(PC) $\bar{\bar{(A)+(DPTR)}}$
Переход, если аккумулятор равен нулю	JZ rel	2	2	(PC) $\bar{\bar{(PC) + 2}}$, если (A) = 0 to (PC) $\bar{\bar{(PC) + rel}}$
Переход, если аккумулятор не равен нулю	JNZ rel	2	2	(PC) $\bar{\bar{(PC) + 2}}$, если (A) $\neq 0$ to (PC) $\bar{\bar{(PC) + rel}}$
Переход, если перенос равен единице	JC rel	2	2	(PC) $\bar{\bar{(PC) + 2}}$, если (C) = 1 to (PC) $\bar{\bar{(PC) + rel}}$
Переход, если перенос равен нулю	JNC rel	2	2	(PC) $\bar{\bar{(PC) + 2}}$, если (C) = 0 to (PC) $\bar{\bar{(PC) + rel}}$
Переход, если бит равен единице	JB bit,rel	3	2	(PC) $\bar{\bar{(PC) + 3}}$, если (b) = 1 to (PC) $\bar{\bar{(PC) + rel}}$
Переход, если бит равен нулю	JNB bit,rel	3	2	(PC) $\bar{\bar{(PC) + 3}}$, если (b) = 0 to (PC) $\bar{\bar{(PC) + rel}}$

Окончание табл. 2.7

1	2	3	4	5
Переход, если бит установлен, с последующим сбросом бита	JBC bit,rel	3	2	$(PC) \dot{=} (PC) + 3$, если $(b) = 1$ то $(b) \dot{=} 0$ и $(PC) \dot{=} (PC) + rel$
Декремент регистра и переход, если не нуль	DJNZ Rn,rel	2	2	$(PC) \dot{=} (PC) + 2$, $(Rn) \dot{=} (Rn) - 1$, если $(Rn) \dot{>} 0$ то и $(PC) \dot{=} (PC) + rel$
Декремент прямоадресуемого байта и переход, если не нуль	DJNZ ad,rel	3	2	$(PC) \dot{=} (PC) + 2$, $(ad) \dot{=} (ad) - 1$, если $(ad) \dot{>} 0$ то и $(PC) \dot{=} (PC) + rel$
Сравнение аккумулятора с прямоадресуемым байтом и переход, если не равно	CJNE A,ad,rel	3	2	$(PC) \dot{=} (PC) + 3$, если $(A) \dot{<} (ad)$, то $(PC) \dot{=} (PC) + rel$, если $(A) < (ad)$, то $(C) \dot{=} 1$, иначе $(C) \dot{=} 0$
Сравнение аккумулятора с константой и переход, если не равно	CJNE A,#d,rel	3	2	$(PC) \dot{=} (PC) + 3$, если $(A) \dot{<} \#d$, то $(PC) \dot{=} (PC) + rel$, если $(A) < \#d$, то $(C) \dot{=} 1$, иначе $(C) \dot{=} 0$
Сравнение регистра с константой и переход, если не равно	CJNE Rn,#d,rel	3	2	$(PC) \dot{=} (PC) + 3$, если $(Rn) \dot{<} \#d$, то $(PC) \dot{=} (PC) + rel$, если $(Rn) < \#d$, то $(C) \dot{=} 1$, иначе $(C) \dot{=} 0$
Сравнение байта в РПД с константой и переход, если не равно	CJNE @Ri,#d,rel	3	2	$(PC) \dot{=} (PC) + 3$, если $(Ri) \dot{<} \#d$, то $(PC) \dot{=} (PC) + rel$, если $(Ri) < \#d$, то $(C) \dot{=} 1$, иначе $(C) \dot{=} 0$
Длинный вызов подпрограммы	LCALL ad16	3	2	$(PC) \dot{=} (PC) + 3$, $(SP) \dot{=} (SP) + 1$ $((SP)) \dot{=} (PC_{0-7})$, $(SP) \dot{=} (SP) + 1$ $((SP)) \dot{=} (PC_{8-15})$, $(PC) \dot{=} ad16$
Абсолютный вызов подпрограммы в пределах страницы в 2 Кб	ACALL ad11	2	2	$(PC) \dot{=} (PC) + 3$, $(SP) \dot{=} (SP) + 1$ $((SP)) \dot{=} (PC_{0-7})$, $(SP) \dot{=} (SP) + 1$ $((SP)) \dot{=} (PC_{8-15})$, $(PC_{0-10}) \dot{=} ad11$
Возврат из подпрограммы	RET	1	2	$(PC_{8-15}) \dot{=} ((SP))$, $(SP) \dot{=} (SP) - 1$ $(PC_{0-7}) \dot{=} ((SP))$, $(SP) \dot{=} (SP) - 1$
Возврат из подпрограммы обработки прерывания	RETI	1	2	$(PC_{8-15}) \dot{=} ((SP))$, $(SP) \dot{=} (SP) - 1$ $(PC_{0-7}) \dot{=} ((SP))$, $(SP) \dot{=} (SP) - 1$
Холостая команда	NOP	1	1	$(PC) \dot{=} (PC) + 1$

3. ОПИСАНИЕ И РАБОТА С ПРОГРАММОЙ EDSIM51

Симулятор EdSim51 представляет собой Java-приложение. Он запускается двойным щелчком по иконке на рабочем столе. При этом запускается пакетный файл, командная строка которого запускает приложение. Общий вид окна симулятора при запуске показан на рис. 3.1. Окно включает панель микроконтроллера, панель программ, панель сигналов на выводах портов и панель периферийных устройств.

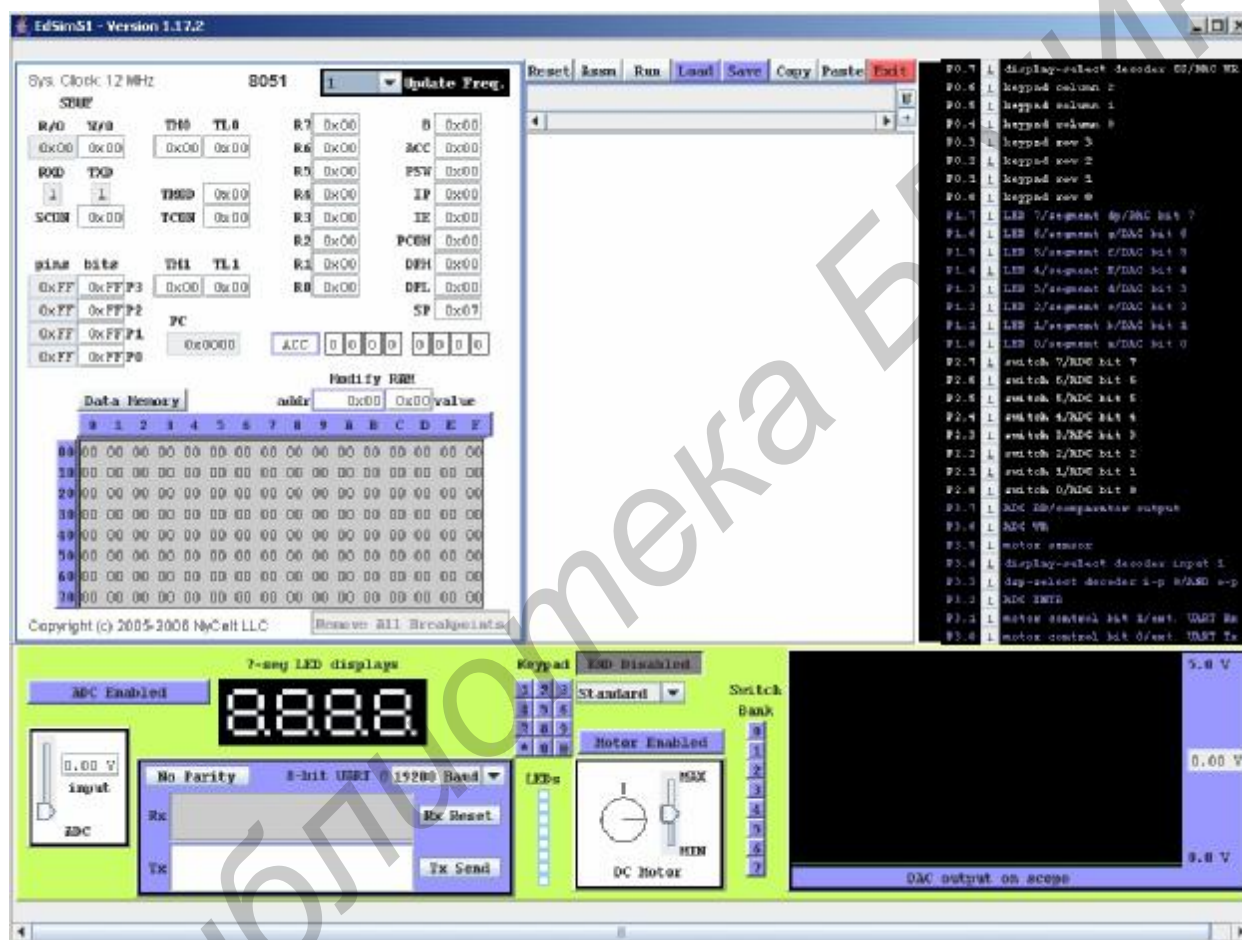


Рис. 3.1. Окно программы EdSim51

Панель микроконтроллера

Внешний вид панели представлен на рис. 3.2. Эта панель включает поле регистров, битовое поле и поле памяти программ/данных и дает доступ ко всем регистрам 8051 и памяти программ/данных. На ней также расположены кнопки переключения частоты обновления значений и сброса точек останова (breakpoints).

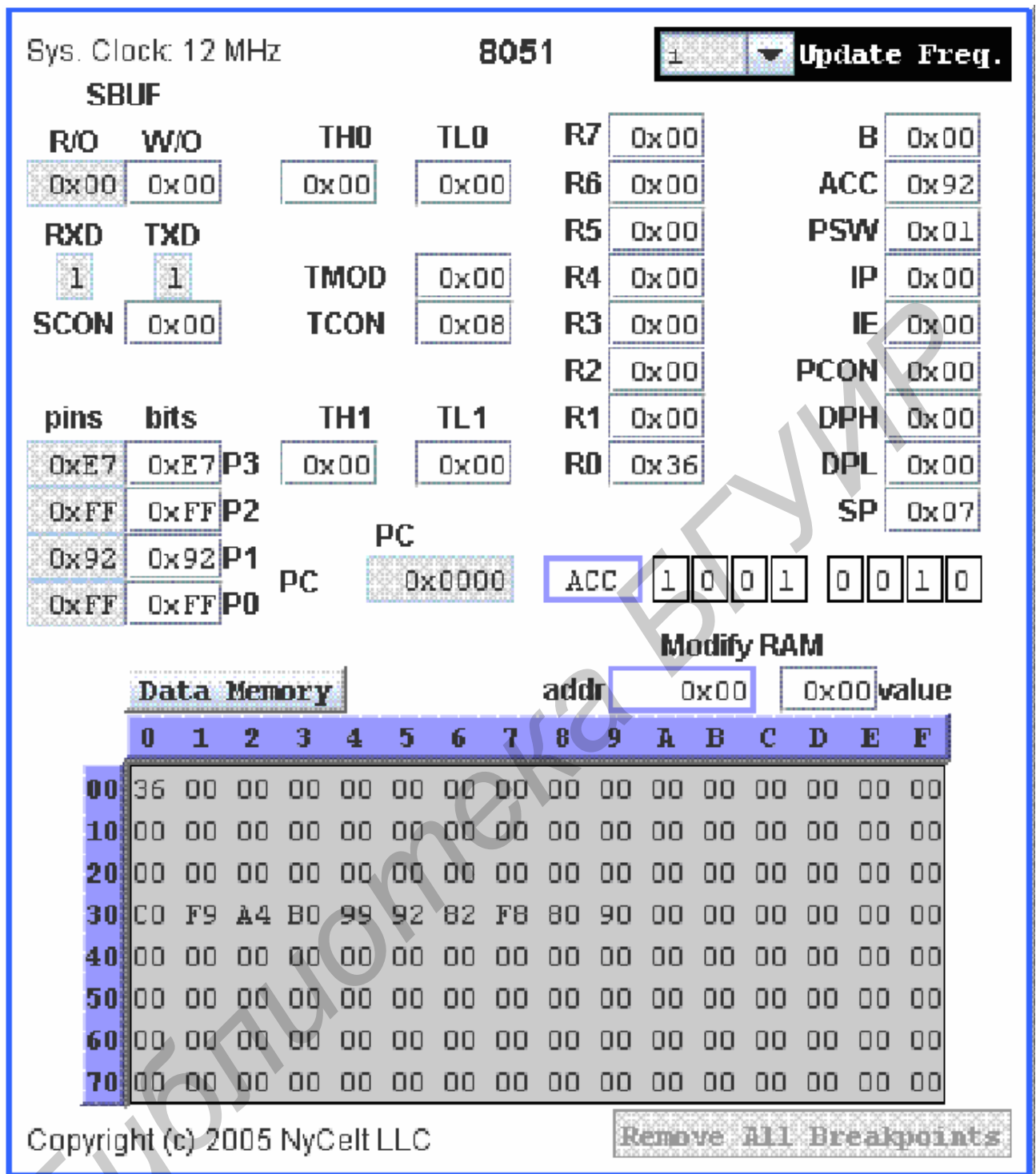


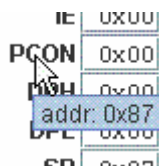
Рис. 3.2. Панель микроконтроллера

Поле регистров

В верхней части панели расположено поле регистров. Ячейки с белым фоном могут редактироваться непосредственно, в отличие от ячеек, выделенных серым. К примеру, биты регистров-защелок портов могут изменяться пользователем, в то время как сигналы на выводах портов зависят

от состояния как регистров защелок, так и внешних периферийных устройств, и поэтому не могут редактироваться. Не редактируется также содержимое программного счетчика.

Если указатель мыши установить поверх одной из подписей названий регистров, высвечивается адрес этого регистра, как показано ниже на примере регистра PCON:



Битовое поле

На рис. 3.2 в битовом поле показаны биты аккумулятора (ACC). Пользователь может ввести любой адрес или имя регистра спецфункций в ячейку, выделенную синим, вместо ACC, тогда будут выведены биты для заданного адреса/регистра. Если поместить указатель мыши на один из битов, будет выведено его описание:



В данном случае показано битовое поле для регистра TMOD. Отметим, что бит 2 – это бит таймер/счетчик (counter/timer C/T). Фон является серым, поскольку регистр TMOD не допускает прямую адресацию отдельных бит, и поэтому пользователь не может их редактировать. Регистр PSW допускает прямую адресацию отдельных бит, поэтому фон ячеек белый, и содержимое можно непосредственно изменять:



Битовое поле можно использовать для просмотра двоичного вида ячеек памяти RAM (по любому адресу от 0 до 7FH), набрав адрес в ячейке с синим контуром. Если возможна прямая адресация отдельных бит, фон будет белым и значения всех восьми бит можно изменять.

Поле памяти программ/данных

Поле памяти программ/данных показано на рис. 3.3. По умолчанию выводится содержимое памяти данных. Содержимое по любому адресу RAM диапазона 00H–7FH может быть изменено, если ввести адрес в ячейку addr и отредактировать значение поля value. Содержимое памяти программ также

может быть просмотрено и отредактировано после нажатия кнопки Data Memory.

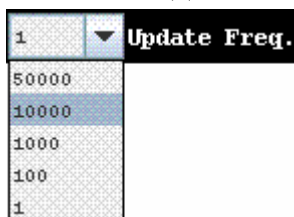
Data Memory																		
															Modify RAM			
															addr	0x00	0x00	value
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F			
00	36	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
30	C0	F9	A4	B0	99	92	82	F8	80	90	00	00	00	00	00			
40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			

Рис. 3.3. Поле памяти программ/данных

При этом название кнопки сменится на Code Memory. Показывается содержимое первых 127 байт памяти программ. Для просмотра другой области памяти нужно ввести начальный адрес в ячейку адреса справа вверху – будет показано содержимое 127 байт по адресам, начиная с введенного начального. Изменение значения value по некоторому адресу изменит программный код.

Частота обновления

Можно выбрать пошаговое или непрерывное выполнение программы. В последнем случае частота обновления экрана будет определяться установками меню частоты обновления (Update Frequency). Выпадающее меню позволяет выбирать обновление экрана после каждого выполнения команды, после каждых 100, 1000, 10 000 или 50 000 команд:

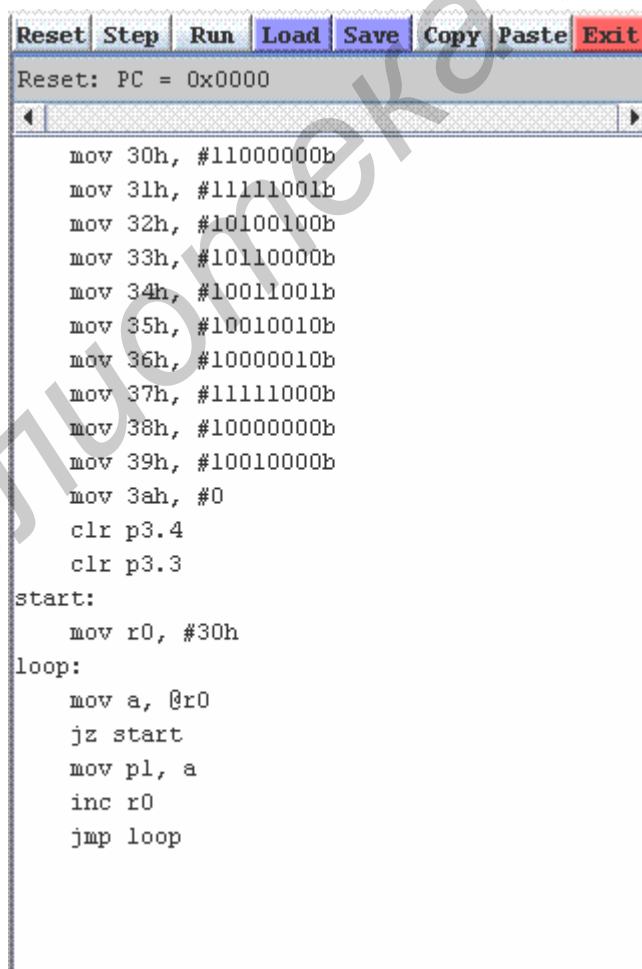


Переключение может производиться и в процессе выполнения программы.

Панель программ

Панель программ показана на рис. 3.4. Вверху находятся кнопки управления программами и кнопка выхода из EdSim51 (красная кнопка Exit), под которой находится кнопка Zoom (+) для мониторов высокого разрешения. Ниже расположено окно команд с полосой прокрутки. В нем указывается адрес и мнемоника текущей команды либо сообщения об ошибках. Еще ниже

расположено окно кода программы на языке Ассемблера. Оно может быть развернуто в отдельное окно большего размера нажатием кнопки U, расположенной между кнопками Exit и Zoom. На рис. 3.4 в нем виден листинг простой программы, выводящей в непрерывном цикле возрастающие числа от 0 до 9 и затем убывающие до 0 в первую (правую) позицию 7-сегментного индикатора. Если фон окна кода белый – текст в нем можно редактировать. Программу на языке Ассемблера можно писать либо прямо в этом окне, либо загрузить ее из файла, нажав кнопку Load. Сохранить текст программы в файле на диске можно, используя кнопку Save. Поддерживаются два типа файлов. По умолчанию файлы сохраняются/загружаются в текстовом формате с расширением .a или .asm. Другой тип файлов – файлы исполняемого кода Intel HEX. Этот формат можно выбрать в меню типов файлов в диалоговом окне Save. Формат Intel HEX поддерживается большинством программаторов, аппаратно загружающих исполняемый программный код в ПЗУ микроконтроллеров. Программа запоминает последний из используемых каталогов, который и открывается при последующем нажатии кнопок Load или Save.



```
Reset Step Run Load Save Copy Paste Exit
Reset: PC = 0x0000
mov 30h, #11000000b
mov 31h, #11111001b
mov 32h, #10100100b
mov 33h, #10110000b
mov 34h, #10011001b
mov 35h, #10010010b
mov 36h, #10000010b
mov 37h, #11111000b
mov 38h, #10000000b
mov 39h, #10010000b
mov 3ah, #0
clr p3.4
clr p3.3
start:
  mov r0, #30h
loop:
  mov a, @r0
  jz start
  mov pl, a
  inc r0
  jmp loop
```

Рис. 3.4. Панель программ

Поддерживается работа с буфером обмена (clipboard), для этого используются кнопки Copy и Paste. С их помощью можно обмениваться фрагментами текста программы с другими приложениями.

Когда программа готова к тестированию, можно нажать кнопку Step для работы в пошаговом режиме или кнопку Run для запуска программы в непрерывном режиме. В любом случае вначале происходит автоматическая обработка текста программы Ассемблером и генерация программного кода. При выявлении ошибок ассемблирования в окно команд выводится сообщение об ошибке (на красном фоне), а строка текста программы с ошибкой в окне программы выделяется красным. Если ассемблирование прошло без ошибок, фон окна программ меняется на серый (в пошаговом режиме) или голубой (в непрерывном режиме). В это время редактирование кода невозможно. Возврат в режим редактирования осуществляется с помощью кнопки Reset.

Встроенный Ассемблер

Встроенный 2-проходный Ассемблер не является полнофункциональным. Он не позволяет связывать несколько файлов (link) и реализует сокращенный набор директив. Однако этого вполне достаточно для начинающих. Ниже перечислены свойства встроенного Ассемблера.

Обработка полного набора команд МК 8051

Команда JMP rel транслируется в SJMP rel либо AJMP rel. Инструкцию LJMP rel нужно программировать особо.

Аналогично, CALL транслируется в ACALL. Инструкцию LCALL нужно программировать особо.

Включены директивы SET, EQU, ORG, USING (последняя определяет используемый банк регистров).

ARn указывает адрес регистров в соответствии с директивой USING (по умолчанию используется банк 0).

Имена регистров специальных функций (SFR) и наименования их бит соответствуют их адресам в МК.

HIGH перед операндом в скобках означает взятие старшего байта операнда.

LOW перед операндом в скобках означает взятие младшего байта операнда.

Метки обозначаются в конце точкой с запятой (;)

Формат чисел по умолчанию – десятичный. Шестнадцатеричные значения могут вводиться с добавлением H после числа либо 0x перед числом. В случае использования символа H число не может начинаться с буквы (например F5H должно быть записано как 0F5H). Двоичные числа вводятся с добавлением B после числа.

Ассемблер нечувствителен к выбору регистра (прописные/строчные буквы).

Отладка

Независимо от того, выполняется программа или нет, после безошибочного ассемблирования кода слева в окне программ выводятся шестнадцатеричные адреса кодов инструкций, как показано на рис. 3.5.

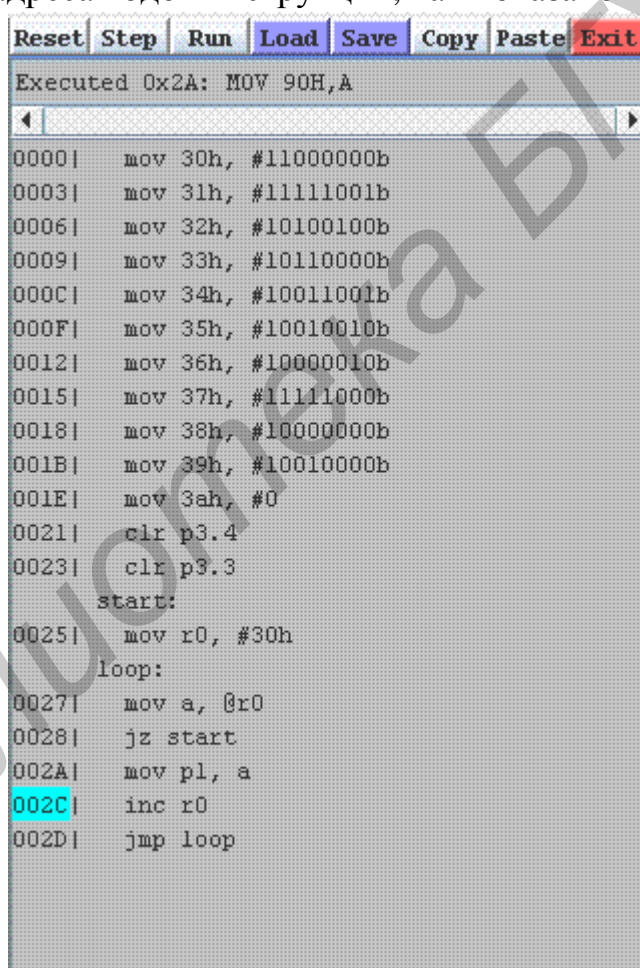


Рис. 3.5. Вид окна программ в режиме выполнения программы

В пошаговом режиме выполняемая инструкция и ее адрес показываются в окне команд (Executed 0x2A: MOV 90H, A), а адрес следующей за ней инструкции выделяется в окне программ (002CH). На рис. 3.5 фон окна

программ серый, что указывает на пошаговый режим. В режиме непрерывного выполнения фон был бы голубым.

Точки останова

Точка останова может быть установлена двойным щелчком при установке указателя мыши на адрес команды в окне программ. При установке точки останова вертикальная черта (|) справа от адреса заменяется звездочкой (*). При исполнении программы происходит ее останов ПЕРЕД точкой останова. Далее можно либо продолжить в пошаговом режиме, либо опять запустить программу дальше. Для удаления точки останова достаточно двойного щелчка при расположении указателя мыши поверх адреса точки останова. Можно воспользоваться кнопкой удаления всех точек останова, нажав кнопку Remove All Breakpoints справа внизу панели микроконтроллера.

Панель сигналов на выводах портов

P0.7	1	display-select decoder CS/DAC WR
P0.6	1	keypad column 2
P0.5	1	keypad column 1
P0.4	1	keypad column 0
P0.3	1	keypad row 3
P0.2	1	keypad row 2
P0.1	1	keypad row 1
P0.0	1	keypad row 0
P1.7	1	LED 7/segment dp/DAC bit 7
P1.6	0	LED 6/segment g/DAC bit 6
P1.5	1	LED 5/segment f/DAC bit 5
P1.4	0	LED 4/segment E/DAC bit 4
P1.3	0	LED 3/segment d/DAC bit 3
P1.2	1	LED 2/segment c/DAC bit 2
P1.1	0	LED 1/segment b/DAC bit 1
P1.0	0	LED 0/segment a/DAC bit 0
P2.7	1	switch 7/ADC bit 7
P2.6	1	switch 6/ADC bit 6
P2.5	1	switch 5/ADC bit 5
P2.4	1	switch 4/ADC bit 4
P2.3	1	switch 3/ADC bit 3
P2.2	1	switch 2/ADC bit 2
P2.1	1	switch 1/ADC bit 1
P2.0	1	switch 0/ADC bit 0
P3.7	1	ADC ED/comparator output
P3.6	1	ADC WR
P3.5	1	motor sensor
P3.4	0	display-select decoder input 1
P3.3	0	display-select decoder input 0
P3.2	1	ADC INTR
P3.1	1	motor control bit 1/ext. UART Rx
P3.0	1	motor control bit 0/ext. UART Tx

Рис. 3.6. Панель сигналов на выводах портов

Эта панель показана на рис. 3.6. На ней указаны обозначения выводов портов, уровни сигналов (логический 0 либо 1), а также англоязычные наименования подключенных к выводам сигналов периферийных устройств.

Панель периферийных устройств

Внешний вид панели периферийных устройств показан на рис. 3.7.

Симулятор включает следующие периферийные устройства:

АЦП (ADC);

компаратор;

4-знаковый 7-сегментный светодиодный (LED) дисплей;

последовательный порт (UART);

клавиатура (Keypad);

банк светодиодов (LED Bank);

реверсивный двигатель постоянного тока (DC Motor);

банк переключателей (Switch Bank);

ЦАП (DAC). Выходной сигнал ЦАП демонстрируется в окне осциллографа, расположенном в правой части панели;

ЖКИ дисплей Hitachi HD44780 с организацией 2 строки по 16 символов, взаимодействующий с МК по 4-разрядной шине.

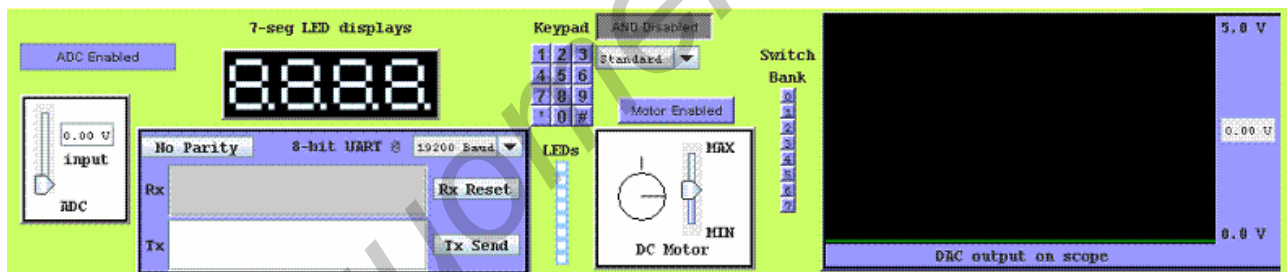


Рис. 3.7. Панель периферийных устройств

Принципиальная схема МК системы и логика управления периферией

Полная принципиальная схема микроконтроллерного устройства, поведение которого симулируется программой EdSim51, показана на рис. 3.8.

Для уяснения логики управления и деталей работы периферийных устройств рассмотрим группы устройств по отдельности.

Банк светодиодов, ЦАП и 7-сегментные индикаторы

Как показано на схеме рис. 3.8, банк светодиодов, входы ЦАП и линии данных 7-сегментных светодиодных индикаторов подключены вместе к выводам порта 1.

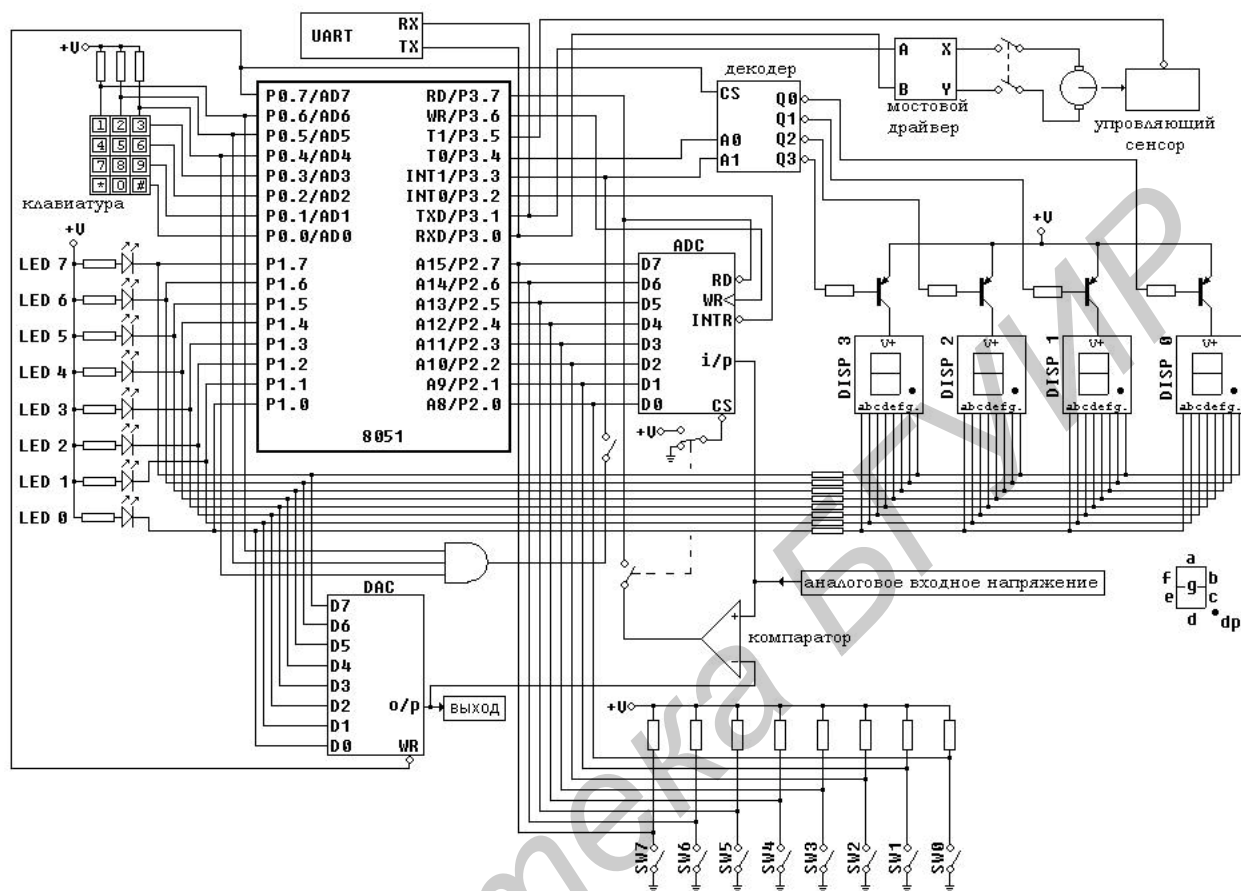


Рис. 3.8. Принципиальная схема микроконтроллерного устройства, симулируемого с помощью программы EdSim51

Выбор включения знакомест осуществляется с помощью дешифратора, на который подаются сигналы с выводов P3.3 и P3.4 (порт 3). Выходы этого дешифратора подключены к базам управляющих транзисторов включения того или иного знакоместа.

Включение дешифратора осуществляется подачей сигнала логической единицы на его вход CS с выхода P0.7 (порт 0). Этот вывод подключен также ко входу WR ЦАП с низким активным уровнем. Поэтому при записи данных в ЦАП запрещается вывод на 7-сегментный дисплей. Данные на входах ЦАП проходят в него при высоком уровне на входе WR. Таким образом, для смены сигнала на выходе ЦАП необходимо подать на вход WR вначале высокий, а затем низкий уровень.

Банк светодиодов подключен к выходам порта P1. Для зажигания светодиодов необходимо подать низкий уровень на соответствующий вывод порта.

Банк переключателей и АЦП

Схема включения банка переключателей и АЦП показана на рис. 3.8. При разомкнутом ключе на входы порта P2 поступает логическая единица (через нагрузочные резисторы), при замкнутом – логический ноль. К этим же выводам подключены и выходы АЦП. Поэтому при использовании ЦАП ВСЕ КЛЮЧИ ДОЛЖНЫ БЫТЬ РАЗОМКНУТЫ! В симуляторе кнопки разомкнутых ключей имеют синий цвет, замкнутых – темно-серый. Замыкание одного из ключей означает замыкание соответствующего вывода АЦП на землю.

Выходы АЦП имеют три состояния. На линии RD, подключенной к выводу P3.7, должен быть низкий уровень для того, чтобы цифровое представление сигнала на входе АЦП появилось на его выходных линиях. Линия WR, подсоединенная к выводу P3.6, используется для стробирования (старта преобразования). Запуск преобразования происходит по фронту положительного импульса на этой линии. По окончании преобразования на линии INTR появляется низкий уровень. Он остается на этой линии до начала следующего преобразования. Эта линия подключена ко входу внешнего прерывания 0 (INT0). Сигнал прерывания поступает на этот вход по окончании аналого-цифрового преобразования. Функции выводов ЦАП указаны в табл. 3.1.

Таблица 3.1

Функции выводов ЦАП

Вывод	Функция
RD	Разрешение по выходу, активный уровень низкий
WR	Старт преобразования, по фронту положительного импульса
INTR	Устанавливается в 0 по окончании преобразования (до начала следующего)
i/p	Аналоговый вход
CS	Выбор устройства, активный уровень низкий
D0-D7	Выходы данных (с тремя состояниями)

Компаратор и ЦАП

Одной из стандартных схем реализации АЦП является схема «компаратор + ЦАП». Для реализации такой схемы выход ЦАП подключен к инвертирующему входу компаратора, как показано на рис. 3.8.

В этом случае штатный АЦП запрещен, поэтому на рис. 3.8 он не подключен к МК 8051. Аналоговый сигнал, кроме штатного АЦП, подается также на неинвертирующий вход компаратора (см. рис. 3.8). На панели периферийных устройств над рисунком АЦП с движковым регулятором уровня аналогового сигнала на его входе есть кнопка ADC Enabled. При нажатии на нее АЦП заменяется компаратором, движок на панели позволяет регулировать уровень аналогового сигнала уже не на входе АЦП, а на входе компаратора. Надпись на кнопке заменяется на Comparator.

Двигатель постоянного тока

Выводы P3.0 и P3.1 подключены к так называемому двойному мостовому драйверу, представляющему из себя четыре зависимых транзисторных ключа, включенных по мостовой схеме. В диагональ моста включен реверсируемый электродвигатель постоянного тока (см. рис. 3.8). Логика управления двигателем показана в табл. 3.2.

Таблица 3.2

Логика управления двигателем

A	B	Функция
0	0	Стоп
0	1	Вперед
1	0	Реверс
1	1	Стоп

Начальное положение метки на валу двигателя – «3 часа» (см. рис. 3.7). Над двигателем расположена вертикальная черта черного цвета, отображающая состояния датчика оборотов. При совмещении метки на валу с позицией датчика его черта становится красного цвета, сигнализируя о замыкании контактов датчика. Контакт датчика подключен к линии порта P3.5, при замыкании здесь появляется логический ноль. Это происходит при каждом обороте вала двигателя, т.к. контакт P3.5 является входом таймера/счетчика 1. Таким образом, можно написать программу счета оборотов двигателя с использованием таймера/счетчика 1. Скорость вращения двигателя может устанавливаться вручную с помощью движкового регулятора, находящегося на панели справа от двигателя.

Линии управления двигателем совмещены с линиями TxD и RxD внутреннего последовательного порта МК 8051. Как видно из рис. 3.8, эти линии подключены также и к внешнему последовательному порту (UART). Поэтому при выдаче сигналов двигателю на экране UART возможен «мусор».

Двигатель может быть запрещен нажатием на кнопку Motor Enabled. Это может понадобиться для работы с UART.

Последовательный порт (UART)

К выходам TxD и RxD внутреннего последовательного порта МК 8051 (выводы P3.0 и P3.1) подключен внешний последовательный порт. Эти порты могут обмениваться сигналами между собой. Данные, полученные с выхода порта UART 8051, появляются в окне Rx. Эти данные можно в любой момент стереть нажатием кнопки Rx Reset.

Из внешнего UART (с выхода Tx) данные могут передаваться на вход RxD UART МК 8051. Для этого надо набрать текст в окне Tx и нажать кнопку Send для начала передачи. Кнопка Send заменяется на Tx Reset, позволяющую очистить окно. Во время передачи фон окна серый и текст в нем недоступен для редактирования. По окончании передачи можно снова нажать кнопку Send либо редактировать текст в окне. Признаком окончания передачи – ASCII-код 0DH. Таким образом, если передается текст abc, то фактически будет передана последовательность ASCII-кодов 61H 62H 63H 0DH.

Скорость передачи UART по умолчанию равна 19 200 бит/с. Можно выбрать одну из четырех скоростей передачи в выпадающем меню (верхний правый угол окна UART). При смене скорости передачи происходит сброс внешнего UART. Контроль четности может быть установлен в три состояния: нет контроля (по умолчанию), контроль четности (Even), контроль нечетности (Odd). При обмене данными с UART сигналы на линиях P3.0 и P3.1 будут переменными, что может оказывать влияние на электродвигатель, подключенный к этим же линиям, вызывая хаотические движения его вала. Чтобы этого не происходило, можно запретить двигатель нажатием на кнопку Motor Enabled. Повторное нажатие вновь активирует двигатель.

Клавиатура (Keypad)

Клавиатура с организацией 4X3 подключена стандартным образом, как видно из рис. 3.8. Используются все выводы порта 0, кроме вывода P0.7.

Организация прерывания от клавиатуры выполнена следующим образом. Три столбца клавиатуры подключены ко входам элемента 3И, выход которого подсоединен к контакту P3.3, входу внешнего прерывания 1. По умолчанию схема 3И запрещена (ее выход отключен от P3.3), поскольку вход прерывания 1 используется также декодером выбора дисплея/индикатора. Чтобы включить

прерывание от клавиатуры, необходимо нажать кнопку AND Disabled. При этом использование мультиплексного режима индикаторов и использование прерываний от клавиатуры станет невозможным. В этом случае следует применять программный опрос клавиатуры.

Справа от клавиатуры расположено меню (см. рис. 3.7), дающее пользователю возможность выбора трех режимов. По умолчанию выбран стандартный режим (Standard). Возможны еще два режима – «Pulse» и «Radio». В стандартном режиме клавиатура работает как набор кнопок с независимой фиксацией. Может быть нажато любое количество кнопок. Повторное нажатие приводит к отпусканию кнопки. Режим «Pulse» соответствует отсутствию фиксации. Нажатие кнопки осуществляется левой кнопкой мыши, когда указатель мыши стоит над этой кнопкой.

Отпускание кнопки мыши соответствует отпусканию кнопки. Режим «Radio» соответствует набору кнопок с зависимой фиксацией. Возможен, однако, переход в состояние, когда все кнопки не нажаты, как в случае с независимой фиксацией (при повторном нажатии на нажатую кнопку). При переключении режимов клавиатуры все кнопки автоматически переходят в отжатое состояние.

Сохранение установок

При выходе из программы многие установки сохраняются в файле edsim51Settings.ser, находящемся в том же каталоге, что и исполняемый модуль edsim51.jar. При последующем запуске симулятор пытается открыть этот файл. Если такой файл не существует, закрыт для доступа или поврежден, используются установки по умолчанию, приведенные в табл. 3.3.

Таблица 3.3

Установки программы, используемые по умолчанию

Параметр	Установка
Частота обновления экрана	1 команда
Адрес в битовом поле	АСС
Память программ/данных	Память данных
Каталог по умолчанию	Зависит от ОС
Разрешен АЦП/компаратор	Разрешен АЦП
Четность внешнего UART	Нет контроля
Скорость передачи UART	19 200 бод
Прерывание от клавиатуры	Запрещено
Тип клавиатуры	Стандартный
Запрет/разрешение двигателя	Разрешен
LED/ЖКИ дисплей	LED дисплей

4. ЛАБОРАТОРНАЯ РАБОТА №1

ОЗНАКОМЛЕНИЕ С ПРОГРАММОЙ EDSIM

Цель: ознакомление с программным эмулятором EdSim.

Для этого необходимо запустить программу EdSim51 двойным щелчком по иконке на рабочем столе. После этого поочередно загрузить на исполнение примеры, представленные вместе с программой, и постараться разобраться в функционировании различных кодов программ и результатов, получаемых при работе программ-примеров.

4.1. Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Краткое описание разобранных примеров программ.
4. Вывод.

4.2. Контрольные вопросы

1. Что такое пошаговый режим выполнения программы?
2. Для чего используется режим отладки?
3. Для чего используются точки останова?
4. Назовите периферийные модули, используемые в программе EdSim. Дайте их краткую характеристику.

5. ЛАБОРАТОРНАЯ РАБОТА №2

АРИФМЕТИКО-ЛОГИЧЕСКИЕ ОПЕРАЦИИ

Цель: приобретение навыков программирования арифметико-логических операций в кодах микроконтроллера.

Описание команд приведено в табл. 2.2–2.7. На рис. 5.1 представлена организация сдвигов различными командами.

5.1. Задание к лабораторной работе

Пусть в памяти программ, начиная с ячейки ADR2, расположена таблица кодов длиной N ($X_i, i=1,2, \dots, N$).

Записать в кодах МК K1816BE1 программу, которая выполняет вычисление заданной функции F над этими кодами. Результат вычисления разместить в регистр В. Программа должна начинаться с ячейки ADR1. Варианты заданий приведены в табл. 5.1.

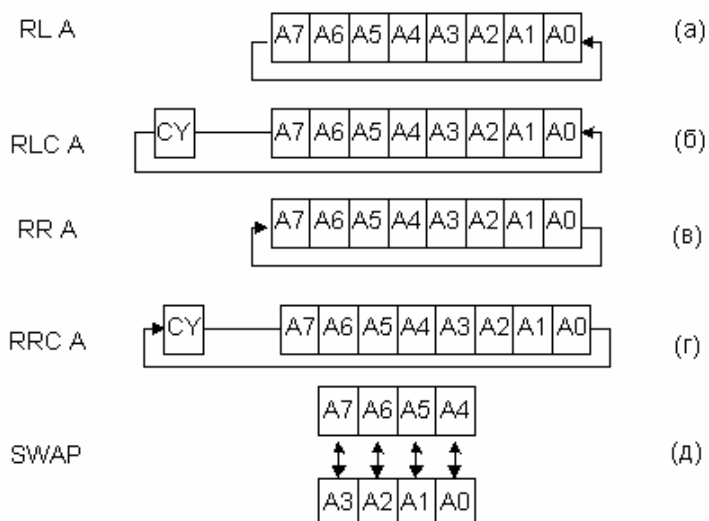


Рис. 5.1. Организация сдвигов командами RL A (а), RLC A (б) RR A (в), RRC A (г), SWAP (д)

Таблица 5.1

Таблица вариантов заданий

№ варианта	ADR1	ADR2	N	Функция
1	714	541	E	Сумма
2	62F	431	F	Max
3	256	621	D	Min
4	87A	711	C	Сумма
5	512	345	A	Max
6	45C	45A	B	Min

Коды данных задать произвольно. Выполнить контрольный просчет.

5.2. Пример программы

Пусть в памяти программ, начиная с ячейки ADR2 = 0F20, расположена таблица кодов длиной N = 5 (21,32,43,54,65).

Необходимо составить и отладить программу, которая вычисляет сумму по модулю 2 этих кодов, результат поместить в ячейку 7F RAM. Программа должна начинаться с ячейки ADR1 = 0200.

Программная реализация:

```

0010 0001 21
0011 0010 32
0100 0011 43
0101 0100 54
0110 0101 65
0110 0001 61

```

Исходные данные в EPROM

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0F20	21	32	43	54	65	00	00	00	00	00	00	00	00	00	00	00

Текст программы:

0200	75D000	MOV PSW, #00	; Задаем нулевой банк
0203	E4	CLR A	; Очищаем аккумулятор
0204	F57F	MOV 7F, A	; Очищаем результат
0206	7A05	MOV R2, #05	; Задаем счетчик
0208	900F1F	MOV DPTR, #0F1F	; Задаем нач. адрес массива
020B	EA	MOV A, R2	; Индекс элемента в A
020C	93	MOVC A, @A+DPTR	; Элемент массива в A
020D	627F	XRL 7F, A	; Сложение по модулю 2
020F	DAFA	DJNZ R2, 020B	; Вычитание и проверка на
0201100		NOP	; выход

Результат работы программы

	6	7	8	9	A	B	C	D	E	F
0070	00	00	00	00	00	00	00	00	00	00

5.3. Содержание отчета

1. Таблицы арифметико-логических команд.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями и контрольным просчетом.
4. Черновик выполнения работы.
5. Вывод.

5.4. Контрольные вопросы

1. Какие команды относят к арифметическим? Дайте пояснение их работы.
2. Поясните различие команд RLC и RR.
3. Назовите и охарактеризуйте два способа адресации в микроконтроллере МК 8051.
4. Назовите разрядность АЛУ МК 8051.

6. ЛАБОРАТОРНАЯ РАБОТА №3

КОМАНДЫ ПЕРЕДАЧИ УПРАВЛЕНИЯ. ОРГАНИЗАЦИЯ ЦИКЛОВ

Цель: изучение организации пространства памяти программ микроконтроллера МК 8051, программных средств управления ходом выполнения программы, приобретение навыков программирования циклических алгоритмов в кодах микроконтроллера.

Для хранения программ и неизменяемых данных в МК 8051 используется логическое однородное линейное пространство памяти CSEG объемом 64 Кб. Память программ адресуется 16-разрядным счетчиком РС. Младшие 4 Кб этого пространства соответствуют встроенному EPROM микроконтроллера, остальные 60 Кб реализуются внешними относительно МК схемами.

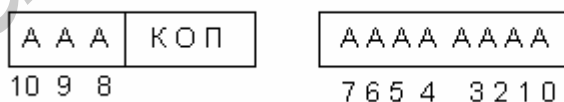
В пространстве CSEG выделяются следующие точки (адреса):

- 0000H - RESET – стартовый адрес при сбросе системы;
- 0003H - EXTI0 – внешнее прерывание 0;
- 000BH - TIMER0 – прерывание таймера/счетчика 0;
- 0013H - EXTI1 – внешнее прерывание 1;
- 001BH - TIMER1 – прерывание таймера/счетчика 1;
- 0023H - SINT – прерывание последовательного порта;
- 002BH - TIMER2 – прерывание таймера 2 (для MCS 52).

В CSEG определены два способа передачи управления:

- 1) прямая адресация с помощью 16-разрядного прямого адреса ad16;
- 2) относительная адресация, имеющая два варианта: с помощью 8-разрядного смещения (целое двоичное со знаком) относительно РС [(A)+(PC0)] или DPTR [(A)+(DPTR)].

Для двух команд (AJMP, ACALL) предусмотрена страничная адресация в CSEG с помощью 11-разрядного адреса ad11. В этом случае 8 младших разрядов адреса располагаются во втором байте команды, а 3 старших – в трех старших разрядах первого байта команды:



Номер страницы задается пятью старшими разрядами программного счетчика РС.

Необходимо подчеркнуть также, что в группе пересылки существуют специальные команды MOVC A,@A+DPTR и MOVC A,@A+PC (см. табл. 2.3–2.7), которые позволяют считывать содержимое памяти программ. Как правило, эта возможность используется для организации таблиц констант в CSEG.

Группу команд передачи управления образуют команды безусловного перехода, команды вызова подпрограмм и команды возврата из подпрограммы.

В большинстве команд используется прямая адресация. Характеристики команд приведены в табл. 2.3–2.7.

6.1. Задание к лабораторной работе

Пусть в памяти программ, начиная с ячейки ADR2, расположена таблица кодов длиной N.

Записать в кодах МК 8051 программу, которая выполняет пересылку данного массива в RAM, начиная с адреса ADR3. Программа должна начинаться с ячейки ADR1. Варианты заданий приведены в табл. 6.1.

Таблица 6.1

Таблица вариантов заданий

№ варианта	ADR1	ADR2	N	ADR3
1	714	541	E	5F
2	62F	431	F	AD
3	256	621	D	23
4	87A	711	C	18F
5	512	345	A	4D
6	45C	45A	B	5E

Коды данных задать произвольно.

6.2. Пример программы

В памяти команд с адреса ADR2 = 0D80 расположено N = 0CH шестнадцатеричных кодов, например: FF, 00, 11, 22, 33, 44, 55, 66, 77, 88, 99, AA.

Необходимо переписать их в память данных, начиная с адреса ADR3 = 65H. Программа должна начинаться с адреса ADR1 = 0F00H.

Текст программы:

```

0F00 900D7F    MOV DPTR,#0D7F
0F03 7870     MOV R0,#70
0F05 790C     MOV R1,#0C
0F07 E9       MOV A,R1
0F08 93       MOV A,@A+DPTR
0F09 F6       MOV @R0, A
0F0A D800     DJNZ R0, 0F0C
0F0C D9F9     DJNZ R1, 0F07
0F0E 00       NOP
    
```

Содержимое памяти программ до выполнения программы

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0D80H	FF	00	11	22	33	44	55	66	77	88	99	AA	00	00	00	00
0D90H	00															

Содержимое памяти данных после выполнения программы

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0060H	00	00	00	00	00	FF	00	11	22	33	44	55	66	77	88	99
0070H	AA															

6.3. Содержание отчета

1. Таблицы команд передачи управления.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями и контрольным просчетом.
4. Черновик выполнения работы.
5. Вывод.

6.4. Контрольные вопросы

1. Поясните принцип работы команд передачи управления.
2. Поясните на примере текста программы организацию циклов в микроконтроллере.
3. Что входит в состав устройства управления микроконтроллера?
4. Какую архитектуру организации памяти имеет микроконтроллер 8051?

7. ЛАБОРАТОРНАЯ РАБОТА №4

ФОРМИРОВАНИЕ ВРЕМЕННЫХ ЗАДЕРЖЕК

Цель: формирование временных интервалов при помощи вложенных циклов и ветвлений в программах для микроконтроллера 8051.

Рассмотрим пример программы формирования временного интервала с использованием вложенных циклов. В качестве счетчиков циклов используем регистры R1 и R0, в которые занесем числа N и M:

```

MOV R1, N
LOOP 2: MOV R0, M
LOOP 1: NOP
        DJNZ R0, LOOP 1
        DJNZ R1, LOOP 2
    
```

Рассчитаем длительность выполнения данного фрагмента. Зная время выполнения каждой команды (MOV – 1 такт; NOP – 1 такт; DJNZ – 2 такта) и учитывая, что первая команда выполняется однократно, внутренний цикл – M раз, внешний цикл – N раз, получаем длительность выполнения данной программы:

$$T = 1 + N(1 + 2 + M(1 + 2 + 2)) \text{ тактов.}$$

При заданной величине T , зная тактовую частоту процессора, легко можно подобрать целые числа N и M , чтобы обеспечить требуемую задержку. При необходимости можно добавлять дополнительные незначащие команды (например, NOP).

7.1. Задание к лабораторной работе

Записать в кодах МК 8051 программу, которая выполняет задержку на время, определенное в варианте задания, и переключает светодиод на панели симулятора в инверсное состояние. Программа может начинаться с любой ячейки. Варианты заданий приведены в табл. 7.1.

Таблица 7.1

Таблица вариантов заданий

№ варианта	Задержка	№ светодиода
1	1	1
2	1,5	2
3	1	3
4	2	4
5	0,5	5
6	0,3	7

7.2. Содержание отчета

1. Расчет констант для организации задержки.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями и контрольным просчетом.
4. Черновик выполнения работы.
5. Вывод.

7.3. Контрольные вопросы

1. Поясните принцип организации задержек на примере кода программы для лабораторного задания.
2. Как управлять выводами портов в МК 8051?
3. Что входит в пространство RSEG?
4. Поясните принцип формирования условных переходов на примере кода программы для лабораторной работы.

8. ЛАБОРАТОРНАЯ РАБОТА №5

РАБОТА С СЕМИСЕГМЕНТНЫМ ИНДИКАТОРОМ

Цель: использование управляющих сигналов для формирования различной информации на семисегментном индикаторе эмулятора микроконтроллера 8051.

8.1. Задание к лабораторной работе

Ознакомьтесь с примером «sevenSegDisplays.asm» к программе EdSym51.

На примере представленного кода сформируйте программу, которая выводит на дисплей индикатора эмулятора строку из 4 символов. Информация о строке представлена в табл. 8.1.

Таблица 8.1

Таблица вариантов заданий

№ варианта	Строка
1	1234
2	2345
3	3456
4	4567
5	1834
6	9876

8.2. Содержание отчета

1. Расчет констант для каждого символа дисплея.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями.
4. Вывод.

8.2. Контрольные вопросы

1. Поясните принцип динамической индикации.
2. Как управлять выводами портов в МК 8051?
3. Что такое банки регистров?
4. Какие типы адресаций используются в микроконтроллере МК 8051?

9. ЛАБОРАТОРНАЯ РАБОТА №6

РАБОТА С ДВИГАТЕЛЕМ

Цель: написание программы работы с клавиатурой, вывод управляющих сигналов на выводы портов и информации на семисегментный индикатор для эмулятора микроконтроллера 8051.

9.1. Задание по лабораторной работе

Ознакомьтесь с примером «motor.asm» к программе EdSym51.

На примере представленного кода сформируйте программу, которая включает/останавливает двигатель по нажатию клавиши и производит подсчет числа его оборотов, выводя результат на семисегментный индикатор.

9.2. Содержание отчета:

1. Алгоритм работы программы.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями.
4. Вывод.

9.3. Контрольные вопросы

1. Как управлять выводами портов в МК 8051 для запуска/останова двигателя?
2. Назовите максимальный период заполнения 16-разрядного таймера при тактовой частоте 10 МГц.
3. Что такое прерывания в микроконтроллере?
4. Что такое аккумулятор в микроконтроллере?

10. ЛАБОРАТОРНАЯ РАБОТА №7

ПРОГРАММИРОВАНИЕ КЛАВИАТУРЫ И ДИСПЛЕЯ

Цель: написание программы работы с клавиатурой и вывод информации на семисегментный индикатор для эмулятора микроконтроллера 8051.

10.1. Задание к лабораторной работе

Ознакомьтесь с примером «scanKeypad.asm» к программе EdSym51.

На примере представленного кода сформируйте программу, которая выводит на дисплей индикатора эмулятора код нажатой на клавиатуре цифры.

10.2. Содержание отчета

1. Алгоритм работы программы считывания клавиатуры.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями.
4. Вывод.

10.3. Контрольные вопросы

1. Как управлять выводами портов в МК 8051 для опроса клавиатуры?
2. Какими типами информационных объектов может оперировать АЛУ?
3. Сколько портов имеет микроконтроллер МК 8051.
4. Какую информацию несут флаги PSW МК 8051?

11. ЛАБОРАТОРНАЯ РАБОТА №8

ПРОГРАММИРОВАНИЕ ЦИФРОАНАЛОГОВОГО ПРЕОБРАЗОВАТЕЛЯ

Цель: написание программы для работы с цифроаналоговым преобразователем для эмулятора микроконтроллера 8051.

11.1. Задание к лабораторной работе

Ознакомьтесь с примером «dac.asm» к программе EdSym51.

```

clr r0.7      ;включаем ЦАП
loop:
  mov r1, a   ;посылаем данные из аккумулятора в порт ЦАП (P1)
  add a, #8   ;увеличиваем аккумулятор на 8
  jmp loop   ;возвращаемся на цикл
  
```

На примере представленного кода сформируйте программу, которая выводит на цифроаналоговый преобразователь сигналы, заданные в варианте задания (табл. 11.1).

Таблица 11.1

Таблица вариантов заданий

№ варианта	Сигнал на выходе цифроаналогового преобразователя
1	Меандр частотой 1 Гц и скважностью 2
2	Меандр частотой 1 Гц и скважностью 4
3	Меандр частотой 1 Гц и скважностью 8
4	Треугольный, $T_{\text{фр}}=1$ с, $T_{\text{ср}}=0,5$ с
5	Треугольный, $T_{\text{фр}}=2$ с, $T_{\text{ср}}=1,5$ с
6	Треугольный, $T_{\text{фр}}=3$ с, $T_{\text{ср}}=0,3$ с

11.2. Содержание отчета

1. Алгоритм работы программы формирования сигналов ЦАП.
2. Задание к лабораторной работе.
3. Текст программы с пояснениями.
4. ВыводД.

11.3. Контрольные вопросы

1. Что такое цифроаналоговый преобразователь, для чего он используется?
2. Как управлять ЦАП в представленном для работы эмуляторе микроконтроллера МК 8051 (на примере программы для лабораторного задания)?
3. Для чего нужен тактовый генератор в микроконтроллере 8051?
4. Как выбираются банки регистров в микроконтроллере 8051?

ЛИТЕРАТУРА

1. Сташин, В. В. Проектирование цифровых устройств на однокристалльных микроконтроллерах / В. В. Сташин, А. В. Урусов, О. Ф. Мологонцева. – М. : Энергоатомиздат, 1990.

2. Андреев, Д. В. Программирование микроконтроллеров MCS-51 : учеб. пособие / Д. В. Андреев. – Ульяновск : УлГТУ, 2000.

3. Edsim51 – the 8051 Simulator for teachers and students. – Электронный ресурс <http://www.edsim51.com/index.html>.

4. Лапшин, С. М. Проектирование микропроцессорных систем передачи и обработки информации : метод. пособие по курсовому и дипломному проектированию по курсам «Устройства цифровой техники» и «Микропроцессоры в системах телекоммуникаций» для студ. спец. «Телекоммуникационные системы» / С. М. Лапшин. – Минск : БГУИР, 1997. – 51 с.

Библиотека БГУИР

СОДЕРЖАНИЕ

1. АРХИТЕКТУРА И СТРУКТУРА МИКРОКОНТРОЛЛЕРА (МК) 8051	3
2. СИСТЕМА КОМАНД МК 8051	23
3. ОПИСАНИЕ И РАБОТА С ПРОГРАММОЙ EDSIM51	33
4. ЛАБОРАТОРНАЯ РАБОТА №1	47
5. ЛАБОРАТОРНАЯ РАБОТА №2	47
6. ЛАБОРАТОРНАЯ РАБОТА №3	50
7. ЛАБОРАТОРНАЯ РАБОТА №4	52
8. ЛАБОРАТОРНАЯ РАБОТА №5	53
9. ЛАБОРАТОРНАЯ РАБОТА №6	54
10. ЛАБОРАТОРНАЯ РАБОТА №7	55
11. ЛАБОРАТОРНАЯ РАБОТА №8	56
ЛИТЕРАТУРА	57

Учебное издание

Гурский Александр Леонидович
Терех Иван Сергеевич

**ЦИФРОВЫЕ И МИКРОПРОЦЕССОРНЫЕ УСТРОЙСТВА СИСТЕМ
ТЕЛЕКОММУНИКАЦИЙ**

Лабораторный практикум
для студентов специальностей I-45 01 03 «Сети телекоммуникаций»,
I-98 01 02 «Защита информации в телекоммуникациях»
всех форм обучения

Редактор Н. В. Гриневич
Корректор М. В. Тезина

Подписано в печать 31.08.2008.
Гарнитура «Таймс».
Уч.-изд. л. 3,0.

Формат 60×84 1/16.
Печать ризографическая.
Тираж 75 экз.

Бумага офсетная.
Усл. печ. л. 3,49.
Заказ 78.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0056964 от 01.04.2004. ЛП №02330/0131666 от 30.04.2004.
220013, Минск, П. Бровки, 6