4.Система команд микроконтроллера семейства 8051.

4.1.1.Общая характеристика.

Микро-ЭВМ рассматриваемого семейства являются типичными микропроцессорными устройствами с архитектурой SISC - со стандартным набором команд. Поэтому их система команд довольно обширна и включает в себя 111 основных команд. Их длина – один, два или три байта, причем большинство из них (94%) – одно- или двухбайтные. Все команды выполняются за один или два машинных цикла (соответственно 1 или 2 мкс при тактовой частоте 12 МГц), исключение – команды умножения и деления, которые выполняются за четыре машинных цикла (4 мкс). Микро-ЭВМ семейства 8051 используют прямую, непосредственную, косвенную и неявную, адресацию данных

В качестве операндов команд микро-ЭВМ семейства 8051 могут использовать отдельные биты, четырехбитные цифры, байты и двухбайтные слова.

Все эти черты обычны для набора команд любого SISC-процессора и по сранению с RISC набором команд обеспечивает большую компактность программного кода и увеличение быстродействия при выполнении сложных операций.

В то же время, набор команд семейства 8051 имеет несколько особенностей, связанных с типичными функциями выполняемыми микроконтроллерами - управлением, для которого типичным является оперирование с одноразрядными двоическими сигналами, большое число операций ввода выводи и ветвлений программы.

Наиболее существенная особенность системы команд рассматриваемых микро ЭВМ это возможность адресации отдельных бит в резидентной памяти данных. Кроме того, как отмечалось, некоторые регистры блока регистров специальных функций также допускают адресацию отдельных бит. Карты адресов отдельных бит в резидентной памяти данных и в блоке регистров специальных функций.

4.1.2.Типы команд

Всего микро-ЭВМ выполняют 13 типов команд, они приведены в таблице. Как следует из нее, первый байт команды всегда содержит код операции (КОП), а второй и третий (если они присутствуют в команде) – адреса операндов или их непосредственные значения.

Тип команды	Первый байт D7...D0	Второй байт D7...D0	Третий байт D7...D0
тип 1	коп		
тип 2	коп	#d	
тип 3	коп	ad	
тип 4	коп	bit	
тип 5	коп	rel	
тип 6	коп	a7...a0	
тип 7	коп	ad	#d
тип 8	коп	ad	rel
тип 9	коп	ads	add
тип 10	коп	#d	rel
тип 11	коп	bit	rel
тип 12	коп	ad16h	ad16l
тип 13	коп	#d16h	#d16l

Таблица. 6. *Типы команд*

4.1.3.Типы операндов

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

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

Четырёхбитные операнды используются только при операциях обмена SWAP и XCHD.

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

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

Адреса	(D_7)							(D_0)
7FH								
2FH	7F	7E	7D	7C	7B	7A	79	78
2EH	77	76	75	74	73	72	71	70
2DH	6F	6E	6D	6C	6B	6A	69	68
2CH	67	66	65	64	63	62	61	60
2BH	5F	5E	5D	5C	5B	5A	59	58
2AH	57	56	55	54	53	52	51	50
29H	4F	4E	4D	4C	4B	4A	49	48
28H	47	46	45	44	43	42	41	40
27H	3F	3E	3D	3C	3B	3A	39	38
26H	37	36	35	34	33	32	31	30
25H	2F	2E	2D	2C	2B	2A	29	28
24H	27	26	25	24	23	22	21	20
23H	1F	1E	1D	1C	1B	1A	19	18
22H	17	16	15	14	13	12	11	10
21H	0F	0E	0D	0C	0B	0A	09	08
20H	07	06	05	04	03	02	01	00
1FH	Банк 3							
18H								
17H	Банк 2							
10H								
0FH	Банк 1							
08H								
07H	Банк 0							
00H								

Рис. 9*. Карта адресуемых битов в резидентной памяти данных*

4.1.4.Группы команд.

Система команд семейства MCS-51 содержит 111 базовых команд, которые по функциональному признаку можно подразделить на пять:

- пересылки данных;
- арифметических операций;
- логических операций;
- операций над битами;
- передачи управления.

Формат команд - одно-, двух- и трехбайтовый, причем большинство команд (94) имеют формат один или два байта. Первый байт любых типа и формата всегда содержит код операции, второй и третий байты содержат либо адреса операндов, либо непосредственные операнды.

Состав операндов включает в себя операнды четырех типов: биты, ниблы (4 разряда), байты и 16-битные слова. Время исполнения команд составляет 1, 2 или 4 машинных цикла. При тактовой частоте 12 мГц длительность машинного цикла составляет 1 мкс, при этом 64 команды исполняются за 1 мкс, 45 команд - за 2 мкс и 2 команды (умножение и деление) - за 4 мкс.

Набор команд MCS-51 поддерживает следующие режимы адресации.

*Прямая адресация (Direct Addressing).*Операнд определяется 8-битным адресом в инструкции. Эта адресация используется только для внутренней памяти данных и регистров SFR.

*Косвенная адресация (Indirect Addressing).*В этом случае инструкция адресует регистр, содержащий адрес операнда. Данный вид адресации может применяться при обращении как к внутреннему, так и внешнему ОЗУ. Для указания 8-битных адресов могут использоваться регистры R0 и R1 выбранного регистрового банка или указатель стека SP.

Для 16-битной адресации используется только регистр "указатель данных" (DPTR - Data Pointer).

*Регистровая адресация (Register Instruction).*Данная адресация применяется для доступа к регистрам R0+R7 выбранного банка. Команды с регистровой адресацией содержат в байте кода операции трехбитовое поле, определяющее номер регистра. Выбор одного из четырех регистровых банков осуществляется программированием битов селектора банка (RS1, RS0) в PSW.

*Непосредственная адресация (Immediate constants).*Операнд содержится непосредственно в поле команды вслед за кодом операции и может занимать один или два байта ($data_8$, $data_{16}$).

*Индексная адресация (Indexed Addressing).*Индексная адресация используется при обращении к памяти программ и только при чтении. В этом режиме осуществляется просмотр таблиц в памяти программ. 16-битовый регистр (DPTR или PC) указывает базовый адрес требуемой таблицы, а аккумулятор указывает на точку входа в нее. Адрес элемента таблицы находится сложением базы с индексом (содержимым аккумулятора).

Другой тип индексной адресации применяется в командах "перехода по выбору" (Case Jump). При этом адрес перехода вычисляется как сумма указателя базы и аккумулятора.

*Неявная адресация (Register-Specific Instructions).*Некоторые инструкции используют индивидуальные регистры (например, операции с аккумулятором, DPTR), при этом данные регистры не имеют адреса, указывающего на них; это заложено в код операции

4.1.5.Обозначения, используемые при описании команд.

Rn (n = 0, 1,..., 7) – регистр общего назначения в выбранном банке регистров;

@Ri(i= 0, 1) – регистр общего назначения в выбранном банке регистров, используемый в качестве регистра косвенного адреса;

ad – адрес прямоадресуемого байта;

ads – адрес прямо адресуемого байта-источника;

add – адрес прямо адресуемого байта-получателя;

ad11 – 11-разрядный абсолютный адрес перехода;

ad16 – 16-разрядный абсолютный адрес перехода;

rel – относительный адрес перехода;

#d – непосредственный операнд;

#d16 – непосредственный операнд (2 байта);

bit – адрес прямо адресуемого бита;

/bit – инверсия прямо адресуемого бита;

A - аккумулятор;

PC – счетчик команд;

DPTR – регистр указатель данных;

( ) – содержимое ячейки памяти или регистра,

4.1.6.Команды пересылки данных микроконтроллера 8051.

Эта группа представлена 28 командами, их краткое описание приведено в таблице, где также указаны тип команды (Т) в соответствии с таблицей, ее длина в байтах (Б) и время выполнения в машинных циклах (Ц).

Таблица.7.*Команды передачи данных*

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Пересылка в аккумулятор из регистра (n=0÷7)	MOV A, Rn	11101rrr	1	1	1	(A) ← (Rn)
Пересылка в аккумулятор прямоадресуемого байта	MOV A, ad	11100101	3	2	1	(A) ← (ad)
Пересылка в аккумулятор байта из РПД (i=0,1)	MOV A, @Ri	1110011i	1	1	1	(A) ← ((Ri))
Загрузка в аккумулятор константы	MOV A, #d	01110100	2	2	1	(A) ← #d
Пересылка в регистр из аккумулятора	MOV Rn, A	11111rrr	1	1	1	(Rn) ← (A)
Пересылка в регистр прямоадресуемого байта	MOV Rn, ad	10101rrr	3	2	2	(Rn) ← (ad)
Загрузка в регистр константы	MOV Rn, #d	01111rrr	2	2	1	(Rn) ← #d
Пересылка по прямому адресу аккумулятора	MOV ad, A	11110101	3	2	1	(ad) ← (A)
Пересылка по прямому адресу регистра	MOV ad, Rn	10001rrr	3	2	2	(ad) ← (Rn)
Пересылка прямоадресуемого байта по прямому адресу	MOV add, ads	10000101	9	3	2	(add) ← (ads)
Пересылка байта из РПД по прямому адресу	MOV ad, @Ri	1000011i	3	2	2	(ad) ← ((Ri))
Пересылка по прямому адресу константы	MOV ad, #d	01110101	7	3	2	(ad) ← #d
Пересылка в РПД из аккумулятора	MOV @Ri, A	1111011i	1	1	1	((Ri)) ← (A)
Пересылка в РПД прямоадресуемого байта	MOV @Ri, ad	0110011i	3	2	2	((Ri)) ← (ad)
Пересылка в РПД константы	MOV @Ri, #d	0111011i	2	2	1	((Ri)) ← #d
Загрузка указателя данных	MOV DPTR, #d16	10010000	13	3	2	(DPTR) ← #d16
Пересылка в аккумулятор байта из ПП	MOVC A, @A+DPTR	10010011	1	1	2	← ((A) +(DPTR))
Пересылка в аккумулятор байта из ПП	MOVC A, @A+PC	10000011	1	1	2	(PC) ← (PC)+1, (A) ← ((A)+(PC))

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Пересылка в аккумулятор байта из ВПД	MOVX A, @Ri	1110001i	1	1	2	(A) ← ((Ri))
Пересылка в аккумулятор байта из расширенной ВПД	MOVX A,@DPTR	11100000	1	1	2	(A) ← ((DPTR))
Пересылка в ВПД из аккумулятора	MOVX @Ri, A	1111001i	1	1	2	((Ri)) ← (A)
Пересылка в расширенную ВПД из аккумулятора	MOVX @DPTR,A	11110000	1	1	2	((DPTR)) ← (A)
Загрузка в стек	PUSH ad	11000000	3	2	2	(SP) ← (SP) + 1, ((SP)) ← (ad)
Извлечение из стека	POP ad	11010000	3	2	2	(ad) ← (SP), (SP) ← (SP) - 1
Обмен аккумулятора с регистром	XCH A, Rn	11001rrr	1	1	1	(A) ↔ (Rn)
Обмен аккумулятора с прямоадресуемым байтом	XCH A, ad	11000101	3	2	1	(A) ↔ (ad)
Обмен аккумулятора с байтом из РПД	XCH A, @Ri	1100011i	1	1	1	(A) ↔ ((Ri))
Обмен младших тетрад аккумулятора и байта РПД	XCHD A, @Ri	1101011i	1	1	1	$(A_{0...3}) \leftrightarrow ((Ri)_{0...3})$

По команде MOV выполняется пересылка данных из второго операнда в первый. Эта команда не имеет доступа ни к внешней памяти данных, ни к памяти программ. Для этих целей предназначены команды M0VX и MOVC соответственно. Первая из них обеспечивает чтение/запись байт из внешней памяти данных, вторая – чтение байт из памяти программ.

По команде XCH выполняется обмен байтами между аккумулятором и ячейкой РПД, а по команде XCHD – обмен младшими тетрадами (битами 0 – 3).

Команды PUSH и POP предназначены соответственно для записи данных в стек и их чтения из стека. Размер стека ограничен лишь размером резидентной памяти данных. В процессе инициализации микро-ЭВМ после сигнала сброса или при включении питающего напряжения в SP заносится код 07H. Это означает, что первый элемент стека будет располагаться в ячейке памяти с адресом 08H.

Группа команд пересылок микроконтроллера имеет следующую особенность - в ней нет специальных команд для работы со специальными регистрами: PSW, таймером, портами ввода-вывода. Доступ к ним, как и к другим регистрам специальных функций, осуществляется заданием соответствующего прямого адреса, т.е. это команды обычных пересылок, в которых вместо адреса можно ставить название соответствующего регистра. Например, чтение PSW в аккумулятор может быть выполнено командой

MOV A, PSW

которая преобразуется Ассемблером к виду

MOV A, 0D0h (E5 D0),

где E5 – код операции, а D0 – операнд (адрес PSW).

Кроме того, следует отметить, что в микро-ЭВМ аккумулятор имеет два различных имени в зависимости от способа адресации: A – при неявной адресации (например, MOV A, R0) и ACC – при использовании прямого адреса. Первый способ предпочтительнее, однако, не всегда применим.

4.1.7.Команды арифметических операций 8051.

В данную группу входят 24 команды, краткое описание которых приведено в таблице. Из нее следует, что микроЭВМ выполняет достаточно широкий набор команд для организации обработки целочисленных данных, включая команды умножения и деления.

В таблице также указаны тип команды (Т) в соответствии с таблицей, ее длина в байтах (Б) и время выполнения в машинных циклах (Ц).

Таблица.8.*Арифметические операции.*

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Сложение аккумулятора с регистром (n=0÷7)	ADD A, Rn	00l01rrr	1	1	1	(A) ← (A) + (Rn)
Сложение аккумулятора с прямоадресуемым байтом	ADD A, ad	00100101	3	2	1	(A) ← (A) + (ad)
Сложение аккумулятора с байтом из РПД (i = 0,1)	ADD A, @Ri	0010011i	1	1	1	(A) ← (A) + ((Ri))
Сложение аккумулятора с константой	ADD A, #d	00100100	2	2	1	(A) ← (A) + #d
Сложение аккумулятора с регистром и переносом	ADDC A, Rn	00111rrr	1	1	1	(A) ← (A) + (Rn) + (C)
Сложение аккумулятора с прямоадресуемым байтом и переносом	ADDC A, ad	00110101	3	2	1	(A) ← (A) + (ad) + (C)
Сложение аккумулятора с байтом из РПД и переносом	ADDC A, @Ri	0011011i	1	1	1	(A) ← (A) + ((Ri)) + (C)
Сложение аккумулятора с константой и переносом	ADDC A, #d	00110100	2	2	1	(A) ← (A) + # d + (C)
Десятичная коррекция аккумулятора	DA A	11010100	1	1	1	Если $(A_{0...3})$>9 или ((AC)=1),то$(A_{0...3})\leftarrow (A_{0...3})$ + 6, затем если $(A_{4...7})$ >9 или ((C)=1), то $(A_{4...7}) \leftarrow (A_{4...7})$ + 6
Вычитание из аккумулятора регистра и заёма	SUBB A, Rn	10011rrr	1	1	1	(A) ← (A) - (C) - (Rn)
Вычитание из аккумулятора прямоадресуемого байта и заема	SUBB A, ad	10010101	3	2	1	(A) ← (A) - (C) - ((ad))
Вычитание из аккумулятора байта РПД и заема	SUBB A, @Ri	1001011i	1	1	1	(A) ← (A) - (C) - ((Ri))
Вычитание из аккумулятора константы и заема	SUBB A, d	10010100	2	2	1	(A) ← (A) - (C) - #d
Инкремент аккумулятора	INC A	00000100	1	1	1	(A) ← (A) + 1

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Инкремент регистра	INC Rn	00001rrr	1	1	1	(Rn) ← (Rn) + 1
Инкремент прямоадресуемого байта	INC ad	00000101	3	2	1	(ad) ← (ad) + 1
Инкремент байта в РПД	INC @Ri	0000011i	1	1	1	((Ri)) ← ((Ri)) + 1
Инкремент указателя данных	INC DPTR	10100011	1	1	2	(DPTR) ← (DPTR) + 1
Декремент аккумулятора	DEC A	00010100	1	1	1	(A) ← (A) – 1
Декремент регистра	DEC Rn	00011rrr	1	1	1	(Rn) ← (Rn) – 1
Декремент прямоадресуемого байта	DEC ad	00010101	3	2	1	(ad) ← (ad) – 1
Декремент байта в РПД	DEC @Ri	0001011i	1	1	1	((Ri)) ← ((Ri)) - 1
Умножение аккумулятора на регистр В	MUL AB	10100100	1	1	4	(B)(A) ← (A)*(B)
Деление аккумулятора на регистр В	DIV AB	10000100	1	1	4	(B).(A) ← (A)/(B)

По результату выполнения команд ADD, ADDC, SUBB, MUL и DIV устанавливаются флаги PSW, структура которых приведена в таблице.

Флаг C устанавливается при переносе из разряда D7, т. е. в случае, если результат не помещается в восемь разрядов; флаг AC устанавливается при переносе из разряда D3 в командах сложения и вычитания и служит для реализации десятичной арифметики. Этот признак используется командой DAA.

Флаг OV устанавливается при переносе из разряда D6, т. е. в случае, если результат не помещается в семь разрядов и восьмой не может быть интерпретирован как знаковый. Этот признак служит для организации обработки чисел со знаком.

Наконец, флаг P устанавливается и сбрасывается аппаратно. Если число единичных бит в аккумуляторе нечетно, то P = 1, в противном случае P = 0.

4.1.8.Команды логических операций микроконтроллера 8051.

В этой группе 25 команд, их краткое описание приведено в таблице. Нетрудно видеть, что эти команды позволяют выполнять операции над байтами: логическое И (/\ ), логическое ИЛИ (\/), исключающее ИЛИ ((+)), инверсию (NOT), сброс в нулевое значение и сдвиг. В таблице также указаны тип команды (Т) в соответствии с таблицей, ее длина в байтах (Б) и время выполнения в машинных циклах (Ц).

Таблица.9*Логические операции*

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Логическое И аккумулятора и регистра	ANL A, Rn	01011rrr	1	1	1	(A) ← (A) AND (Rn)
Логическое И аккумулятора и прямоадресуемого байта	ANL A, ad	01010101	3	2	1	(A) ← (A) AND (ad)
Логическое И аккумулятора и байта из РПД	ANL A, @Ri	0101011i	1	1	1	(A) ← (A) AND ((Ri))
Логическое И аккумулятора и константы	ANL A, #d	01010100	2	2	1	(A) ← (A) AND #d
Логическое И прямоадресуемого байта и аккумулятора	ANL ad, A	01010010	3	2	1	(ad) ← (ad) AND (A)
Логическое И прямоадресуемого байта и константы	ANL ad, #d	01010011	7	3	2	(ad) ← (ad) AND #d
Логическое ИЛИ аккумулятора и регистра	ORL A, Rn	01001rrr	1	1	1	(A) ← (A) OR (Rn)
Логическое ИЛИ аккумулятора и прямоадресуемого байта	ORL A, ad	01000101	3	2	1	(A) ← (A) OR (ad)
Логическое ИЛИ аккумулятора и байта из РПД	ORL A, @Ri	0100011i	1	1	1	(A) ← (A) OR ((Ri))
Логическое ИЛИ аккумулятора и константы	ORL A, #d	01000100	2	2	1	(A) ← (A) OR #d
Логическое ИЛИ прямоадресуемого байта и аккумулятора	ORL ad, A	01000010	3	2	1	(ad) ← (ad) OR (A)
Логическое ИЛИ прямоадресуемого байта и константы	ORL ad, #d	01000011	7	3	2	(ad) ← (ad) OR #d
Исключающее ИЛИ аккумулятора и регистра	XRL A, Rn	01101rrr	1	1	1	(A) ← (A) XOR (Rn)
Исключающее ИЛИ аккумулятора и прямоадресуемого байта	XRL A, ad	01100101	3	2	1	(A) ← (A) XOR (ad)
Исключающее ИЛИ аккумулятора и байта из РПД	XRL A, @Ri	0110011i	1	1	1	(A) ← (A) XOR ((Ri))
Исключающее ИЛИ аккумулятора и константы	XRL A, #d	01100100	2	2	1	(A) ← (A) XOR #d
Исключающее ИЛИ прямоадресуемого байта и аккумулятора	XRL ad, A	01100010	3	2	1	(ad) ← (ad) XOR (A)
Исключающее ИЛИ прямоадресуемого байта и константы	XRL ad, #d	01100011	7	3	2	(ad) ← (ad) XOR #d
Сброс аккумулятора	CLR A	11100100	1	1	1	(A) ← 0
Инверсия аккумулятора	CPL A	11110100	1	1	1	(A) ← NOT(A)
Сдвиг аккумулятора влево циклический	RL A	00100011	1	1	1	$(A_{n+1}) \leftarrow (A_n)$, n=0÷6, $(A_0) \leftarrow (A_7)$
Сдвиг аккумулятора влево через перенос	RLC A	00110011	1	1	1	$(A_{n+1}) \leftarrow (A_n)$, n=0÷6 $(A_0) \leftarrow$ (C), (C) $\leftarrow (A_7)$
Сдвиг аккумулятора вправо циклический	RR A	00000011	1	1	1	$(A_n) \leftarrow (A_{n+1})$, n=0÷6, $(A_7) \leftarrow (A_0)$
Сдвиг аккумулятора вправо через перенос	RRC A	00010011	1	1	1	$(A_n) \leftarrow (A_{n+1})$, n=0÷6 $(A_7) \leftarrow$ (C), (C) $\leftarrow (A_0)$
Обмен местами тетрад в аккумуляторе	SWAP A	11000100	1	1	1	$(A_{0...3}) \leftrightarrow (A_{4...7})$

4.1.9.Команды операций над битами микроконтроллера 8051.

Группа состоит из 12 команд, краткое описание которых приведено в таблице. Эти команды позволяют выполнять операции над отдельными битами: сброс, установку, инверсию бита, а также логические И (/\) и ИЛИ (\/). В качестве "логического" аккумулятора, участвующего во всех операциях с двумя операндами, выступает признак переноса C (разряд D7 PSW), в качестве операндов могут использоваться 128 бит из резидентной памяти данных и регистры специальных функций, допускающие адресацию отдельных бит.

В таблице также указаны тип команды (Т) в соответствии с таблицей, ее длина в байтах (Б) и время выполнения в машинных циклах (Ц).

Таблица.10.*Операции с битами*

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

4.1.10.Команды передачи управления микроконтроллера 8051.

Группа представлена командами безусловного и условного переходов, командами вызова подпрограмм и командами возврата из подпрограмм.

В таблице также указаны тип команды (Т) в соответствии с таблицей, ее длина в байтах (Б) и время выполнения в машинных циклах (Ц).

Таблица.11.*Команды передачи управления*

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
Длинный переход в полном объеме ПП	LJMP ad16	00000010	12	3	2	(PC) ← ad16
Абсолютный переход внутри страницы в 2 Кб	AJMP ad11	$a_{10}a_9a_8$00001	6	2	2	(PC) ← (PC) + 2, (PC_{0-10}) ← ad11
Короткий относительный переход внутри страницы в 256 байт	SJMP rel	10000000	5	2	2	(PC) ← (PC) + 2, (PC) ← (PC) +rel
Косвенный относительный переход	JMP @A+DPTR	01110011	1	1	2	(PC) ← (A) + (DPTR)
Переход, если аккумулятор равен нулю	JZ rel	01100000	5	2	2	(PC)←(PC)+2, если (A)=0, то (PC)←(PC)+rel
Переход, если аккумулятор не равен нулю	JNZ rel	01110000	5	2	2	(PC)←(PC)+2, если (A)≠0, то (PC)←(PC)+rel
Переход, если перенос равен единице	JC rel	01000000	5	2	2	(PC)←(PC)+2, если (C)=1, то (PC)←(PC)+rel
Переход, если перенос равен нулю	JNC rel	01010000	5	2	2	(PC)←(PC)+2, если (C)=0, то (PC)←(PC)+rel
Переход, если бит равен единице	JB bit, rel	00100000	11	3	2	(PC)←(PC)+3, если (b)=l, то (PC)←(PC)+rel
Переход, если бит равен нулю	JNB bit, rel	00110000	11	3	2	(PC)←(PC)+3, если (b)=0, то (PC)←(PC)+rel
Переход, если бит установлен, с последующим сбросом бита	JBC bit, rel	00010000	11	3	2	(PC) ← (PC) + 3, если (b)=1, то (b) ← 0 и (PC)← (PC) + rel
Декремент регистра и переход, если не нуль	DJNZ Rn, rel	11011rrr	5	2	2	(PC) ← (PC) + 2, (Rn) ← (Rn) - 1, если (Rn) ≠ 0, то (PC) ← (PC) + rel
Декремент прямоадресуемого байта и переход, если не нуль	DJNZ ad, rel	11010101	8	3	2	(PC) ← (PC) + 2, (ad) ← (ad) - 1, если (ad) ≠ 0, то (PC) ← (PC) + rel
Сравнение аккумулятора с прямоадресуемым байтом и переход, если не равно	CJNE A, ad, rel	10110101	8	3	2	(PC) ← (PC) + 3,если (A) ≠ (ad), то (PC) ← (PC) + rel,если (A) < (ad), то (C) ← 1, иначе (C) ← 0
Сравнение аккумулятора с константой и переход, если не равно	CJNE A, #d, rel	10110100	10	3	2	(PC) ← (PC) + 3,если (A) ≠ #d, то (PC) ← (PC) + rel,если (A) < #d, то (C) ← 1, иначе (C) ← 0
Сравнение регистра с константой и переход, если не равно	CJNE Rn, #d, rel	10111rrr	10	3	2	(PC) ← (PC) + 3,если (Rn) ≠ #d, то (PC) ← (PC) + rel,если (Rn) < #d, то (C) ← 1, иначе (C) ← 0
Сравнение байта в РПД с константой и	CJNE @Ri,#d,re	1011011i	10	3	2	(PC) ← (PC) + 3,если ((Ri)) ≠ #d, то

Название команды	Мнемокод	КОП	Т	Б	Ц	Операция
переход, если не равно						(PC) ← (PC) + rel,если ((Ri)) < #d, то (C) ← 1, иначе (C) ← 0
Длинный вызов подпрограммы	LCALL adl6	00010010	12	3	2	(PC) ← (PC) + 3, (SP) ← (SP) +1, ((SP)) ← ($PC_{0...7}$), (SP) ← (SP) + 1, ((SP)) ← ($PC_{8...15}$), (PC) ← ad16
Абсолютный вызов подпрограммы в пределах страницы в 2 Кб	ACALL ad11	$a_{10}a_9a_8$10001	6	2	2	(PC) ← (PC) + 2, (SP) ← (SP) + 1, ((SP)) ← ($PC_{0...7}$), (SP) ← (SP) + 1, ((SP)) ← ($PC_{8...15}$), (PC_{0-10}) ← ad11
Возврат из подпрограммы	RET	00100010	1	1	2	($PC_{8...15}$) ← ((SP)), (SP) ← (SP) - 1, ($PC_{0...7}$) ← ((SP)), (SP) ← (SP) – 1
Возврат из подпрограммы обработки прерывания	RETI	00110010	1	1	2	($PC_{8...15}$) ← ((SP)), (SP) ← (SP) - 1, ($PC_{0...7}$) ← ((SP)), (SP) ← (SP) – 1
Пустая операция	NOP	00000000	1	1	1	(PC) ← (PC) + 1

Команда безусловного перехода LJMP (L – long – длинный) осуществляет переход по абсолютному 16-битному адресу, указанному в теле команды, т. е. команда обеспечивает переход в любую точку памяти программ.

Действие команды AJMP (A – absolute – абсолютный) аналогично команде LJMP, однако в теле команды указаны лишь 11 младших разрядов адреса. Поэтому переход осуществляется в пределах страницы размером 2 Кбайт, при этом надо иметь в виду, что сначала содержимое счетчика команд увеличивается на 2 и только потом заменяются 11 разрядов адреса.

В отличие от предыдущих команд, в команде SJMP (S – short – короткий) указан не абсолютный, а относительный адрес перехода. Величина смещения rel рассматривается как число со знаком, а, следовательно, переход возможен в пределах – 128...+127 байт относительно адреса команды, следующей за командой SJMP.

Команда косвенного перехода JMP @A+DPTR позволяет вычислять адрес перехода в процессе выполнения самой программы.

Командами условного перехода можно проверять следующие условия:

- JZ — аккумулятор содержит нулевое значение;
- JNZ — аккумулятор содержит не нулевое значение
- JC — бит переноса C установлен;
- JNC — бит переноса C не установлен;
- JB — прямо адресуемый бит равен 1
- JNB — прямо адресуемый бит равен 0;
- JBC — прямо адресуемый бит равен 1 и сбрасывается в нулевое значение при выполнении команды.

Все команды условного перехода рассматриваемых микро-ЭВМ содержат короткий относительный адрес, т. е. переход может осуществляться в пределах—128... +127 байт относительно следующей команды.

Команда DJNZ предназначена для организации программных циклов. Регистр Rn или байт по адресу ad, указанные в теле команды, содержат счетчик повторений цикла, а смещение rel — относительный адрес перехода к началу цикла. При выполнении команды содержимое счетчика уменьшается на 1 и проверяется на 0. Если значение содержимого счетчика не равно 0, то осуществляется переход на начало цикла, в противном случае выполняется следующая команда.

Команда CJN удобна для реализации процедур ожидания внешних событий. В теле команды указаны "координаты" двух байт и относительный адрес перехода rel. В качестве двух байт могут быть использованы, например, значения содержимого аккумулятора и прямо адресуемого байта или косвенно адресуемого байта и константы. При выполнении команды значения указанных двух байт сравниваются и в случае, если они не одинаковы, осуществляется переход. Например, команда

WAIT: CJNE A, P0, WAIT

будет выполняться до тех пор, пока значения на линиях порта P0 не совпадут со значениями содержимого аккумулятора.

Действие команд вызова процедур полностью аналогично действию команд безусловного перехода. Единственное отличие состоит в том, что они сохраняют в стеке адрес возврата.

Команда возврата из подпрограммы RET восстанавливает из стека значение содержимого счетчика команд, а команда возврата из процедуры обработки прерывания RETI, кроме того, разрешает прерывание обслуженного уровня. Команды RET и RETI не различают, какой командой – LCALL или ACALL – была вызвана подпрограмма, так как и в том, и в другом случае в стеке сохраняется полный 16-разрядный адрес возврата.

В заключение следует отметить, что большинство Ассемблеров допускают обобщенную мнемонику JMP – для команд безусловного перехода и CALL – для команд вызова подпрограмм. Конкретный тип команды определяется Ассемблером, исходя из "длины" перехода или вызова.