3. Система команд 8-разрядных микропроцессоров и особенности программирования микропроцессоров на языке ассемблера

3.1. Классификация команд

Команды восьмиразрядного процессора можно классифицировать по нескольким признакам. По виду выполняемых операций все команды МП можно разделить на следующие группы:

1) Передачи данных;
2) Арифметических операций;
3) Логических операций
4) Сдвига;
5) Регистровых операций;
6) Передачи управления;
7) Работа со стеком;
8) Ввода/вывода;
9) Управление процессором.

Эти команды занимают в памяти 1 байт, а при использовании непосредственного операнда - 2 байта. Команды можно классифицировать в соответствии с *адресом*, содержащимся в команде на следующие:

1) *Команды обращения к памяти.* Операция, указанная в команде, относится к содержимому, хранящемуся в памяти ЗУ по определенному адресу, т.е. команда задает адреса ячейки памяти ЗУ. Например, команда ADD 200 означает: выбрать число в качестве второго операнда для сложения с числом, хранимым в аккумуляторе и являющимся первым операндом.

2) *Команды обращения к регистру*. Для выполняемой операции не требуется адресация оперативной памяти. Операция выполняется, как правило, над одним операндом, хранящимся в аккумуляторе. Например, CLEAR (Очистить) - означает обнулить аккумулятор.

3) *Команды обращения к устройствам ввода-вывода*. Эти команды обеспечивают передачу данных между МП и периферийным оборудованием.

Кроме этого команды классифицируют на группы по типу операций, которые должны выполняться.

3.2. Команды передачи данных

С помощью таких команд можно осуществить передачу данных от одного из регистров A, B, C, D, E, H и L к одному из регистров A, B, C, D, E, H и L. Кроме того, можно выбрать ячейку памяти M (1байт) микропроцессора, адресуясь к ней, как к одному из регистров. При этом для адресации памяти используется содержимое регистровой пары HL.

Есть еще одна возможность задать содержимое регистров: загрузить непосредственный операнд в выбранный регистр. Эти команды, однако, не позволяют выполнять запоминание в обратном направлении (не существует команд, по которым содержимое регистра могло бы быть помещено в поле операндов команды).

Формат команд:

(Метка:) MOV r_1, r_2 ; Комментарий r_1 = A,B,C,D,E,H,L,M r_2 = A,B,C,D,E,H,L,M.	Move data – передать данные

При этом r_1 определяет регистр или ячейку памяти, в которую производится запись, а r_2 – регистр или ячейку памяти, из которых извлекается число или непосредственный операнд.

Если $r_1=r_2$ то команда является пустой. Вместо этой команды можно использовать команду NOP. Пустая команда ни при каких обстоятельствах не изменяет содержимого регистров, памяти и состояния МП.

(Метка:) MVI r,число ; Комментари) r = A,B,C,D,E,H,L,M	Move immediate data - Передать непосредственный операнд.

По этой команде число, расположенное в поле операндов, заносится в регистр r.

Примеры:

M1: MOV A,B; <B>→<A>

M2: MOV M,A; <A>→<M> или <A>→<<HL>>

MOV B,M; <M>→<B> или <<HL>>→< B >.

MVI M, 12Q; 12Q→<M> или 12Q→ <<HL>>.

Q – обозначает, что число в восьмеричной системе.

Пример: Передать содержимое ячейки 9B 73H в регистр A; содержимое регистров H и L не определено.

Выполнение:

```
MVI     H,9BH     ; <9B>→ H
MVI     L,73H     ; <73>→L
MOV     A, M      ; Выборка из памяти <HL>.
```

3.3. Команды регистровых операций

Команды регистровых операций занимают в памяти один байт. Они используются для того, чтобы содержимое того или иного регистра увеличить или уменьшить на единицу. Это относится к регистрам:

r = B, C, D, E, H, L.

Данные команды изменяют Z-, P- и S–биты состояния, хотя регистр A не участвует в операциях. Благодаря этому можно использовать названные

регистры для организации программных циклов. Формат команд имеет следующий вид:

(Метка:) INR r ; Увеличить содержимое регистра на единицу
; Increment register

Пример:

INR C ; <C> + 1 → <C>
DCR r ; Уменьшить содержимое регистра на 1
; Decrement register

Для загрузки аккумулятора А содержимым ячейки памяти используются команды LDAX B и LDAX D.

LDAX B ; Загрузить Акк. Содержимым ячейки памяти, адрес
; которой
; находится в регистровой паре B,C
; Load Akkumulator
LDAX D ; <<D,E>> → A. Обе эти команды однобайтные.

Команда LDA адрес загружает в Акк. Содержимое ячейки памяти, адресуемой вторым и третьим байтами команды.

[<B3> <B2>] → A.

Команда запоминания STA адрес производит противоположную передачу:

A → [<B3> <B2>] .

3.4. Команды работы со стеком

PUSH rp (rp = B, C; D, E; H, L, PSW) -*Запоминание значения регистров в стеке.* При выполнении команды содержимое пары регистров запоминается в стеке. Одной из пар регистров является слово состояния процессора *PSW*, которое состоит из содержимого *аккумулятора* (старший байт) и регистра состояния (младший байт). Нет такой команды, при выполнении которой запомнится в стеке только один регистр.

Пример:

PUSH B. При выполнении этой команды <B,C>, запоминается в вершине стека, а указатель стека *уменьшается* на 2. <B> запоминается первым, потом <C> заканчивает стек в его вершине.

POP rp (rp = B, C; D, E; H, L, PSW)- *Загрузка регистров из стека.* По этой команде пара регистров *rp* загружается содержимым вершины стека.

Пример: POP D. Эта команда загружает регистры D и E из вершины стека и *увеличивает* указатель стека на 2. Регистр E загружается первым.

Стек имеет следующие особенности:

- указатель стека содержит адрес ячейки, которая была занята самой последней (младший занятый адрес). Стек может быть расположен в любом месте памяти;
- данные запоминаются в стеке с использованием *предуменьшения,* то есть команды уменьшают указатель стека на 1 перед запоминанием каждого байта;
- данные загружаются из стека с использованием *послеувеличения,* то есть команды увеличивают указатель стека 1 после загрузки каждого байта;
- отсутствуют указатели выхода за границы стека в ту или иную сторону, что типично для *микропроцессоров* всех типов.

SPHL - Загрузка указателя стека (sp) – однобайтная команда.

<H> $\rightarrow$ <<SP> +1>; <L> $\rightarrow$ <<SP>>.

PCHL - Загрузка программного счетчика PC .

<H,L> $\rightarrow$ <PC>.

3.5. Команды арифметических и логических операций

Команды арифметических операций.

Эти команды, в зависимости от типа применяемого операнда занимают в памяти 1, 2 или 3 байта. По командам арифметических операций содержимое одного из регистров B, C D, E, H, L или содержимое ячейки памяти M (адресуемой регистровой парой H) или непосредственный операнд команды прибавляется (вычитается) к (из) содержимому регистра A, причем в операции может быть учтено значение C-бита. Значения битов состояния (C, S, Z, P) устанавливаются в зависимости от результата выполнения операции: C - переполнение; S - знак результата (минус S=1); P=1 – четное число единиц в аккумуляторе.

Примеры команд:

```
ADD  r              ; Сложить содержимое регистра  (ячейки памяти) и аккумулятора
ADD  M          ;
ADC  r (M)          ; Сложить содержимое регистра  (ячейки памяти) и аккумулятора
                    ; с учетом бита переноса
SUB  r (M)
SBB  r (M)
ADI число       ; Сложить непосредственный операнд и аккумулятор
ACI  число
```

Аналогичные команды для вычитания SUI и SBI .

DAD rp - сложить содержимое регистровой пары rp (B или D) и аккумулятора, в роли которого выступает регистровая пара H,L.

Команды логических операций.

В состав команд входят команды выполнения операций И, ИЛИ, исключающее ИЛИ (сумма по модулю 2), а также операция сравнения.

Список команд МП:

ANA r (M) <A> and <r>→ <A>

XRA r (M) <A> xor <r>→ <A>

ORA r (M) <A> or <r>→ <A>

CMP r (M) если <A> < <r>, то <C бит> = 1
если <A> = <r>, то <Z бит> = 1
если <A> > <r>, то <C бит> = 0

Аналогично с командами арифметических операций, существуют команды логических операций, когда вместо r задается число.

ANI число;
XRI число;
ORI число;
CPI число;

При выполнении команд сравнения содержимое аккумулятора сравнивается либо с содержимым регистра (CMP r), либо со значением числа (CPI число). При этом содержимое аккумулятора не изменяется. Биты состояния устанавливаются в зависимости от результата вычитания операнда из содержимого аккумулятора.

Команда исключающего ИЛИ

XRA r: (A)⊕(r)→A

производит поразрядное сложение операндов по mod 2. Команда XRA применяется для инвертирования определенных битов слова с помощью слова-маски на основе тождества

$$1 \oplus x = \overline{x}.$$

Другое применение XRA связано со сравнением слов на абсолютное равенство. В единственном случае, когда операнды поразрядно совпадают, результат операции содержит нули во всех разрядах (согласно тождеству $X \oplus X = 0$), о чем сигнализирует флажок (бит состояние) Z=1.

Команды сдвига.

Команды данной группы требуют 1 байт памяти. По этим командам содержимое аккумулятора сдвигается влево или вправо на один разряд. В этой

операции принимает и С-бит. Значение С-бита может в результате выполнения команды изменяться, а значения остальных флагов остается неизменным.

Формат команды имеет вид:

```
(Метка:)  RLC   ; Сдвинуть циклически содержимое аккумулятора влево
                ; Rotate accumulator left
          RRC   ; Сдвинуть циклически содержимое аккумулятора вправо

          RAL   ; Сдвинуть циклически содержимое аккумулятора влево
                ; через бит переноса

          RAR   ; Сдвинуть циклически содержимое аккумулятора вправо
                ; через бит переноса
```

Если обозначить разряды в А от A[0] до A[7], то выполняемую операцию можно описать следующим образом:

RLC: $A[7] \rightarrow < C_б >; A[m] \rightarrow A[m+1], \quad m = 0,...,6; A[7] \rightarrow A[0]$.

RAL: $A[7] \rightarrow < C_б >; A[m] \rightarrow A[m+1], \quad m = 0,...,6; \quad C_б \rightarrow A[0]$.

Схематично четыре операции сдвига можно изобразить следующим образом (рис.4.8).

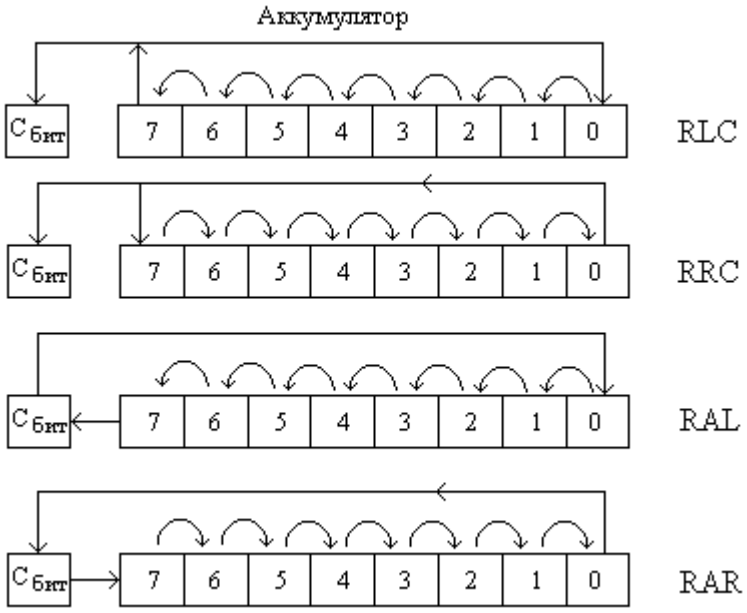

Рисунок 4.8 – Схема осуществления команд сдвига в аккумуляторе

3.6. Команды передачи управления

Эти команды часто называют командами перехода. Позволяют выполнять различные действия в соответствии со значением внешних сигналов или

выработанных в процессе выполнения операций условий. Все типы команды делятся на команды безусловного и условного перехода. К безусловным командам относятся:

```
JUMP  адрес      ; Обеспечивается переход в программе по адресу,
                 ; указанному в команде. При выполнении этой
                 команды
                 ; адрес перехода загружается в программный
                 счетчик PC,
                 ; причем текущее значение PC теряется.

SKIP             ; Пропускается следующая команда программы
                 ; Команда вызова подпрограммы. Осуществляется
CALL имя         переход к
                 ; подпрограмме с указанным именем
RET  №           ; Возврат из подпрограммы
                 ; Осуществляется повторный запуск с адреса: 8 × №
```

При выполнении команды вызова CALL временно запоминается текущее содержимое программного счетчика (т.е. адрес команды, следующий за командой CALL и называемый адресом возврата), и загружается адрес перехода команды CALL в PC, а после выполнения программы адрес возврата передается в PC. Последняя функция реализуется специальной однобайтной командой RET. Для запоминания адресов возврата используется стековая память.

Имеется группа команд условного перехода, выполняемых в зависимости от значения одного из четырех флагов состояния (C, Z, S, P). Если условие перехода выполнено, то осуществляется переход по адресу, указанному в команде. В противном случае выполняется следующая команда. Группу команд условного перехода образуют 8 команд.

```
Метка:  JC      Адрес   ; Перейти, если C бит = 1      Jump if carry
        JNC     Адрес   ; Перейти, если C бит = 0      Jump if no carry
        JZ      Адрес   ; Перейти, если Z бит = 1      Jump if zero
        JNZ     Адрес   ; Перейти, если Z бит = 0      Jump if not zero
        JP      Адрес   ; Перейти, если S бит = 0      Jump if positive
        JM      Адрес   ; Перейти, если S бит = 1      Jump if minus
        JPE     Адрес   ; Перейти, если P бит = 1      Jump if parity even
        JPO     Адрес   ; Перейти, если P бит = 0      Jump if parity odd
```

3.7. Команды ввода/вывода

Имеется две команды*: IN* и *OUT* . Эти команды используются для того, чтобы передать (считать) байт данных из канала ввода в аккумулятор, или для того, чтобы содержимое Akk передать (записать) в выбранный канал вывода. Номера канала при этом задаются операндом.

Номера от *0* до *7* соответствуют каналам ввода, а номера от 8 до 31 – каналам вывода. Поэтому данные команды требуют всегда одного операнда. Команды ввода/вывода не влияют на биты состояния.

Формат команды:

(Метка:) *IN канал* ; Комментарий

Пример: *m1: IN 6* *;* считать один байт из 6-го канала
m2: OUT 29 *;* передать 1 байт из Akk в 29-й канал

Команда останова *HLT* . Она занимает в памяти 1 байт и используется для того, чтобы остановить выполнения команд в *микропроцессоре*. Содержимое регистров и памяти остается неизменным. Повторный запуск микропроцессора возможен только по сигналу прерывания.

Пример программы сложения массива двоичных чисел

Процесс сложения можно выполнять по следующему алгоритму:

Шаг 1. COUNT = <41>
SUM = 0
POINTER = 42
Шаг 2. SUM = SUM + <POINTER>
Шаг 3. COUNT = COUNT – 1
POINTER = POINTER + 1
Шаг 4. Если COUNT > 0 Идти к шагу 2
Шаг 5. <40> = SUM

При составлении программы в качестве счетчика используем один из регистров РОН, а в качестве адресного указателя данных – регистровую пару H, L.

```
;Задание начальных значений
 SUB A          ; очистка аккумулятора
 LXI H, 41H     ; установление счетчика длины
                ; массива
 MOV B, M       ; пересылка длины массива в
                ; РОН B
                ;Программный цикл
SUMD:  INX H    ; увеличение адреса H, L на единицу
```

```
ADD M      ; суммирование с <Akk> содержимого ячейки памяти с
             ;адресом H, L
DCR B      ; уменьшение счетчика массива
JNZ  SUMD  ; проверка условия <B> ≠ 0
             ;Завершающий блок
STA 40H    ; запомнить сумму и записать
             ; по адресу 40
HLT
```