Тема 9. Команди передавання даних.

Состав команд операций передачи данных

Команды передачи данных выполняют передачу данных из регистра в регистр, размещение данных в памяти, размещение извлеченных из памяти данных в УВВ и установление индикатора переноса (табл. 9.1). Почти все команды передачи не влияют на индикаторы МП. Эта группа содержит множество команд, данные могут быть переданы из любой ячейки памяти в любой регистр или наоборот. Каждая команда передачи содержит адреса источника и назначения данных. Способы адресации ориентированы на то, где и как осуществляется поиск данных.

Рассмотрим первую команду передачи, записанную в табл. 9.1: необходимо их передать из регистра *L*в регистр*A* (аккумулятор). Команда ассемблера (MOV *A, L)*означает перемещение. Следующая за мнемоникой буква (A - в нашем примере) указывает назначение, тогда как последняя (L) идентифицирует источник данных. Это покажется немного несовременно, но эта условность принята фирмой Intel для команд микропроцессоров Intel 8080/ /8085.

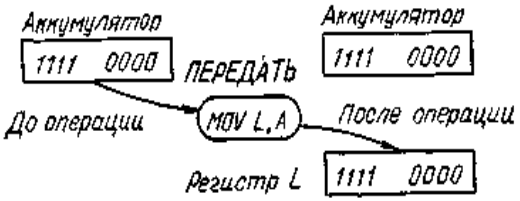


Рис. 9.1. Команда передачи данных из *A* в *L*

Обратимся к примеру на рис.9.1. Источником данных является регистр*A* (аккумулятор), приемником — регистр *L*.Выполненная один раз команда не меняет содержимое аккумулятора.

На рис. 9.2 приведен другой пример. Здесь источником данных является пара регистров *HL*,приемником — 16- разрядный указатель стека. Если рассмотреть внимательно

Таблица 9.1. *Команды передачи данных типового МП*

Операция	Адресация	Мнемоника	КОП	Байты	Формат команд	Символика
Передать L в A	Регистровая	MOV A, L	7D	1	коп	(A) ←(L)
Передать H в A	»	MOV A, H	7C	1	коп	(A) ← (H)
Передать A в L	»	MOV L, A	6F	1	коп	(L) ← (A)
Передать A в H	»	MOV H, A	67	1	коп	(H) ← (A)
Передать HL в PC	»	PCHL	E9	1	коп	(PC) ←(HL)
Передать HL в SP	»	SPHL	F9	1	коп	(SP) ←(HL)
Загрузить A данными	Непосредственная	MVI A	3E	2	КОП данные	(A) ← (байт 2)
Загрузить L данными	»	MVI L	2E	2	КОП данные	(L) ← (байт 2)
Загрузить H данными	»	MVI H	26	2	КОП данные	(H) ← (байт 2)
Загрузить LOC (HL) в A	Косвенная регистровая	MOV A, M	7E	1	коп	(A) ← ((H)(L))
Загрузить HL данными	Непосредственная	LXI H	21	3	КОП мл.байт ст. байт	(L) ← (байт 2) (H) ← (байт 3)
Загрузить SP данными	»	LXI P	31	3	КОП мл.байтст. байт	(SP) ← (байт 2 + 3)
“ Загрузить HL из LOC *	Прямая	LHLD	2A	3	КОП мл.адрес ст. адрес	(L) ← (байт 2 +3) (H) ← ((байт 2 + 3) + 1)
Загрузить A из LOC	»	LDA	3A	3	КОП мл.адрес ст. адрес	(A) ← ((байт 2 +3))
Поместить A в LOCaa	»	STA	32	3	КОП мл.адрес ст. адрес	(адрес) ← (A)
Поместить HL в LOC	»	SHLD	22	3	КОП мл.адрес ст. адрес	(адрес) ←(L) (адрес + 1) ← (H)
Поместить A в LOCaa(HL)	Косвенная регистровая	MOVM, A	77	1	коп	((H)(L)) ← (A)
Поместить L в LOC (HL)	То же	MOV M, L	75	1	коп	((H)(L)) ← (L)
Поместить H в LOC (HL)	»	MOV M, L	74	1	коп	((H)(L)) ← (H)
Ввести в A из порта в LOCa	Прямая	IN	DB	2	КОП адр. порта	(A) ← (адрес порта)
Вывести A в порт в LOCa	»	OUT	D3	2	КОП адр. порта	(адрес порта) ← (A)
Установить индикатор переноса	Неявная	STC	37	1	коп	(CY) ← 1

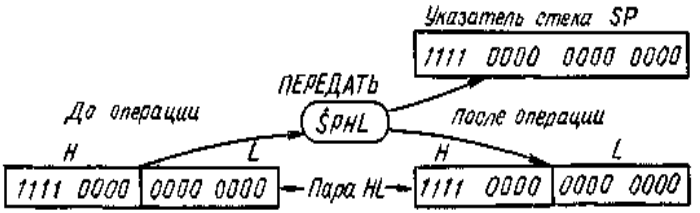


Рис. 9.2. Команда передачи данных из пары регистров *HL*в указатель стека *SP*

мнемонику команды передачи содержимого пары регистров *HL* в указатель стека *SP,* подтвердится, что назначение указывается первым *(SP),* а источник — последним *(HL),*что дает нам мнемонику SPHL.

Существует пять команд непосредственной загрузки данных. Эти команды используются очень часто для помещения начального значения в регистр МП в заданный момент, предшествующий программе. На рис. 9.3 приведен пример такой команды. Пара регистров *HL* емкостью 16 бит должна быть загружена данными, следующими непосредственно за КОП в памяти. Заметим, что согласно табл. 9.1 команда ЗАГРУЗИТЬ *HL* данными является трехбайтовой командой. Первый байт — это КОП (21Н в нашем примере), тогда как два последующих байта являются байтами данных. Второй байт содержит МБ данных(в штриховой рамке), который служит указателем адреса памяти (LOC) и загружается в регистр *L. Содержимое* следующего байта памяти [адрес (LOC + 1) в нашем примере] загружается в регистр *H.*

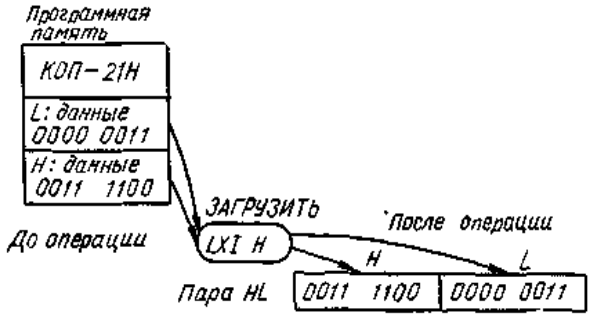


Рис. 9.3 Команда загрузки пары регистров *HL*с непосредственной адресацией

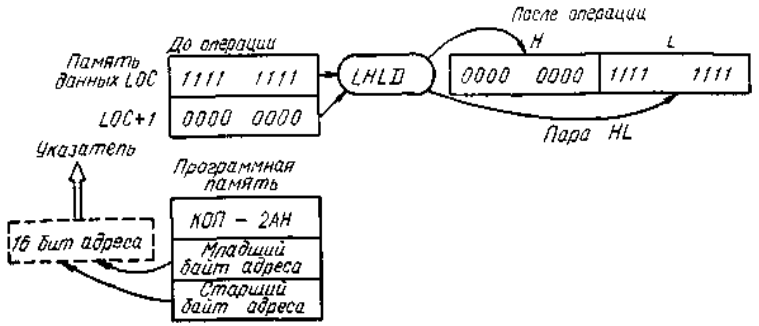


Рис 9.4. Команда загрузки пары регистров *HL*с прямой адресацией

Типовой МП снабжен пятью командами размещения, которые пояснены в табл. 9.1. Рассмотрим пример использования команды прямого размещения содержимого *A,* приведенный на рис. 9.5. Содержимое аккумулятора (регистр*A*) помещается в память (LOC), на которую указывает 16-разрядный адрес, составленный вторым и третьим байтом команды. После выполнения ячейка памяти (LOC) и аккумулятор содержат те же данные, т. е. 1100 0000.

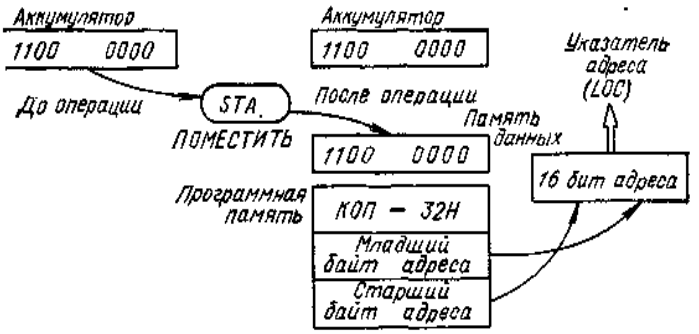


Рис. 9.5. Команда загрузки данных аккумулятора в память с прямой адресацией

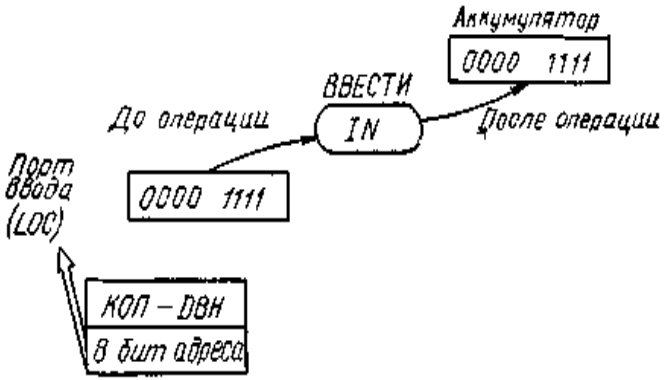


Рис. 9.6. Команда ввода с прямой адресацией

Команда ввода (третья строка снизу в табл. 9.1) идентична команде загрузки. Источник данных передачи является портом ввода, идентифицированным двоичным 8-разрядным числом ($0—255_{10}$), назначение этих данных — аккумулятор МП. Пример выполнения команды представленна рис. 9.6. Данные порта ввода 0000 1111, на

который указывает второй байт команды, передаются в аккумулятор, исходя из порта ввода, идентифицированного LOC.

Таким образом, команды передачи данных предписывают микропроцессору пересылку данных из одного блока в другой, например, из одного регистра в другой. Эти команды всегда определяют источник и приемник данных, причем в команде сначала следует адрес приемника, а потом адрес источника.

Содержимое источника при этом не изменяется.

Команда MOV

MOV A,B; B→A

MOV A,M; M→A,

M адрес ячейки памяти, который предварительно надо поместить в H-пару.

Команда MVI

Команда MVI отличается от команды MOV тем, что в качестве источника данных используется восьмиразрядная константа, которая следует непосредственно за кодом операции. Приемником данных является регистр r или ячейка памяти M.

MV1 C,data8; data8→C

Команда LXI

Команда LXI может быть использована для загрузки регистровых пар (B-пары, D-пары, H-пары) и указателя стека SP шестнадцатиразрядным числом, которое непосредственно следует за кодом операции.

LXI D,data16; data16→(D,E)

Команды LDA, STA

По команде LDA в аккумулятор загружается содержимое ячейки памяти, адрес которой следует за кодом операции. По команде STA содержимое

аккумулятора записывается в ячейку памяти, адрес которой следует за кодом операции.

LDA addr16; addr16 →A

STA addr16; A→addr16

Команды LDA и STA являются трехбайтными. В первом байте содержится код операции, во втором байте < B2 > - младший байт адреса (addr мл), а в третьем байте < B3 > - старший байт адреса (addr ст).

Команда LDAX.

По команде LDAX в аккумулятор загружается содержимое ячейки памяти, адресуемой регистровой парой (B,C) или (D,E).

Команда STAX осуществляет передачу содержимого аккумулятора в ячейку памяти, адресуемой регистровой парой (B,C) или (D,E).

LDAX B; косвенная адресация. Предварительно в B-пару загружается адрес ячейки, содержимое которой загружается в аккумулятор. Операции с портом.

Команды IN и OUT управляют обменом информации между аккумулятором A и портами ввода-вывода. В команде IN (ввод) источником является порт ввода port, а приемником - аккумулятор A. В команде OUT (вывод) источником является аккумулятор A, а приемником порт вывода port. Адрес порта непосредственно следует за кодом операции IN или OUT.

IN addrport;

OUT addrport;

Работа со стеком.

Команда LXI

LXI SP,data16; data16 → SP. С этого момента известно, где в памяти находится стек.

Команды пересылок PUSH (Поместить в стек) и POP (Вытолкнуть из стека) всегда оперирует с регистровой парой (B,C), (D,E), (H,L) или парой регистров (A,РП), образующей слово состояния программы PSW. Напомним, что указатель стека SP содержит адрес той ячейки в стеке, в которую в последний раз была записана информация, т.е адрес верха стека. По команде PUSH (Поместить в стек) выполняются действия: а) содержимое SP сначала уменьшается на 1 (декрементируется); б) старший байт загружается в стек; в) содержимое SP вновь декрементируется; г) младший байт загружается в стек. PUSH B; Записать в стек B-пары. По команде POP (Вытолкнуть из стека) выполняются обратные действия. Сначала младший байт выталкивается из стека в МП и содержимое SP увеличивается на 1 (инкрементируется). Затем старший байт выталкивается из стека в МП и содержимое SP инкрементируется. POP B; вытолкнуть из стека в B-пару.

Использование регистра адреса/данных

Использование пары регистров *HL* в качестве указателя адреса является интересным свойством типового МП. Обычно рассматривают ее использование в качестве указателя адреса, когда она временно берет на себя роль основного счетчика команд, указывая адрес ячейки памяти или УВВ. Многие широко распространенные МП (например, Intel 8080/8085, Z 80) содержат регистры такого типа. Регистры адреса/данных в рассматриваемом типовом МП называются также парой *HL*.-регистров, регистром адреса, счетчиком данных или указателем адреса.

Рассмотрим простую задачу сложения содержимого трех последовательных ячеек памяти с размещением суммы в следующей ячейке памяти (выполнение ее показано на рис. 9.7).

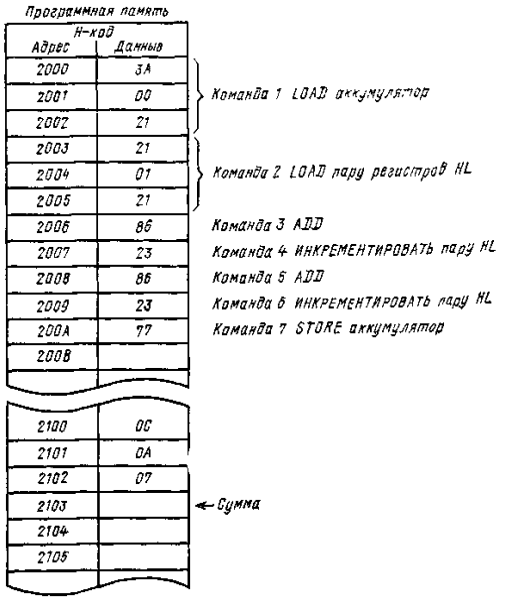


Рис. 9.7. Воображаемая память и команды в примере сложения

Программа загружена в ячейки памяти 2000Н — 200АН, а три слагаемых (0СН + 0АН + 07Н) — в ячейки памяти 2100Н—2102Н. Программа содержит 6 команд, записанных справа на рис. 9.7. Не следует забывать, что текущая сумма будет всегда помещаться в аккумулятор, содержащий вначале первое слагаемое (0СН).

Команда *1* имеет КОП 3АН (рис. 9.7) и приказывает МП ЗАГРУЗИТЬ (LOAD) в аккумулятор содержимое ячейки памяти 2100Н. Выполнение этой команды прямой загрузки аккумулятора показано на рис. 9.8, *а*.

После выполнения команды аккумулятор будет содержать первое слагаемое (0CH).

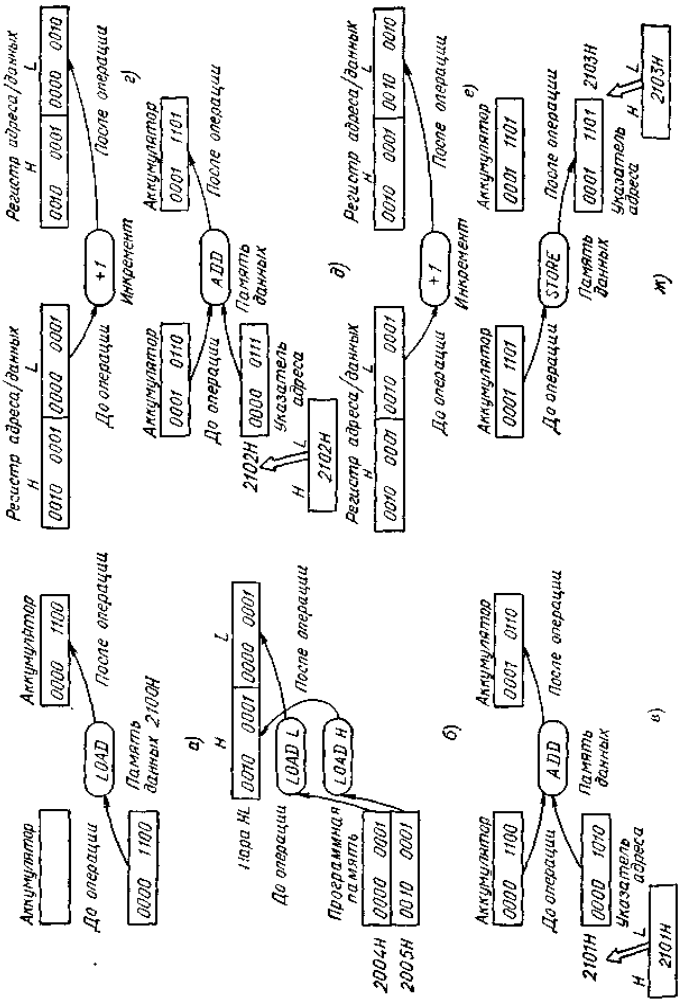


Рис. 9.8. Выполнение операций в соответствии с программой рис. 9.7:

а — команда 1 загрузки аккумулятора; б — команда 2 загрузки пары регистровHL\ в — команда 3 сложения, г — команда 4 инкремента пары регистров; д —команда 5 сложения; е — команда 6 инкремента пары регистров HL\ ж — команда 7 размещения в памяти содержимого аккумулятора

Команда *2* приказывает МП загрузить (LOAD) 2101H в пару регистров *HL*,емкость которых 16 бит. Это число представляет собой адрес памяти данных. Более точно команду *2* можно сформулировать так: загрузить пару регистров *HL*непосредственно следующими за КОП данными, ее выполнение приведено на рис. 9.8, *б.* Заметим, что содержимое первой ячейки памяти (2004H) загружается в младший байт L, следующей за ней — в старший байт *H* пары регистров *HL.*

Команда *3* приказывает МП выполнить операцию сложить (ADD) содержимое аккумулятора с содержимым ячейки памяти, адрес которой содержится в паре регистров *HL.* Ее выполнение приведено на рис. 9.8, *в* (команда ADD). Пара регистров *HL*указывает на ячейку памяти 2101H, и АЛУ складывает свое содержимое ($0000\ 1010_2$) с содержимым аккумулятора ($0000\ 1100_2$), что дает сумму ($0001\ 0110_2$), помещаемую в аккумулятор.

Команда *4* приказывает МП инкрементировать (увеличить на 1) содержимое пары регистров *HL*(см. рис. 9.8, *г).* Заметим, что изменился только младший байт пары регистров *HL.*

Команда *5* снова приказывает МП сложить (ADD) содержимое аккумулятора и ячейки памяти с адресом 2102H, на которую указывает пара регистров *HL*(см. рис. 9.8,*б*) .Оба содержимых складываются, что дает сумму ($0001\ 1101_2$), помещаемую в аккумулятор.

По команде *6* содержимое пары регистров *HL*снова инкрементируется (см. рис. 9.8, *е).*

Команда *7* приказывает МП поместить (STORE) содержимое аккумулятора, т. е. окончательную сумму

($0001\ 1101_2$) в ячейку памяти, на которую указывает пара регистров *HL* (см. рис. 9.8, *ж), т.е.* по адресу 2103Н.

Команды *3, 5, 7,* взаимодействующие с парой регистров *HL*как с указателем адреса, используют косвенно-регистровый способ адресации. Его мы изучим в следующей главе.

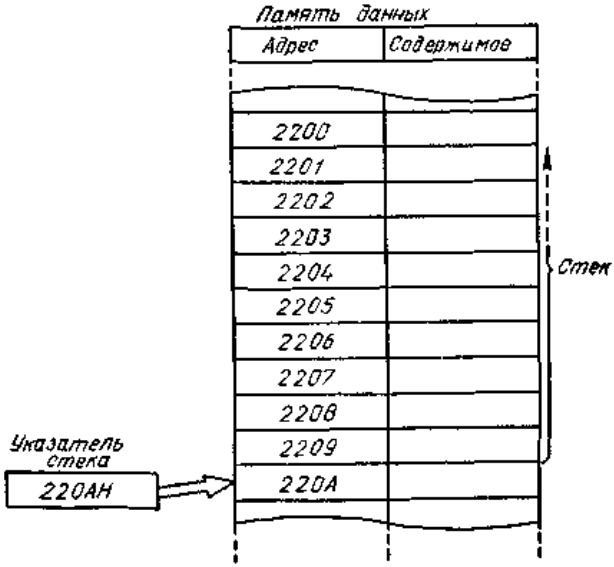
Использование указателя стека

Типовый микропроцессор содержит указатель стека — специализированный 16-разрядный регистр-счетчик, содержимым которого всегда является адрес. Этот адрес принадлежит особой группе ячеек памяти данных, которая называется *стеком.* В некоторых МП стек может быть составлен из группы физически локализованных на кристалле МП ячеек памяти. Когда микропроцессором выполнялась подпрограмма обслуживания прерывания, текущие данные во всех регистрах МП должны были временно сохраняться. Эта сохранность обеспечена стеком. А когда подпрограмма полностью выполнена, содержимое счетчика команд должно быть сохранено таким образом, чтобы МП мог возвратиться в соответствующее место в программной памяти. Зона временной памяти является стеком. Напомним, что подпрограмма является короткой, часто используемой, специализированной программой (например, умножить).

Стек типового микропроцессора будет содержаться в ОЗУ, и его положение определяется программистом. Указатель стека загружается старшим адресом, представляющим собой вершину стека (рис. 9.9). В этом случае указательстека содержит 220АН, что

на единицу старше первой ячейки памяти стека 2209Н.

Данные можно записать в стек, используя команды PUSH (поместить) или CALL (вызвать). Они могут быть считаны из стека по командам POP (извлечь) или RETURN (возврат). Стек функционирует как память с последовательным доступом по типу: данные, поступившие последними, извлекаются первыми (тип LIFO от *Last In — First Out* — последний входит — первый выходит, или FILO от *First In — Last Out* — первый входит — последний выходит).



Рнс. 9.9. Расположение стека в ОЗУ

Команда загрузки в стек (PUSH) приводит к результату, показанному на рис. 9.10, *а.* Содержимое пары регистров *HL*помещается в стек. Отметим, что двухбайтовая пararегистров *HL*должна быть размещена в двух ячейках памяти стека. Последовательность событий может быть описана в соответствии с номерами, показанными в кружках на рис. 9.10.

1. Указатель стека МП декрементируется от 220АН до 2209Н.

2. Указатель стека показывает на ячейку памяти 2209Н по адресной шине и старший байт ($0000\ 0000_2$) помещается в стек.

3. Указатель стека снова декрементируется от 2209Н до 2208Н.

4. Указатель стека указывает на ячейку памяти 2208Н (по адресной шине системы) и младший байт из регистра данных ($0000\ 1111_2$) загружается в стек.

На рис. 9.10, *б* показано выполнение другой операции загрузки. На этот раз в стек загружается содержимое аккумулятора и регистра состояния. Проследим снова за событиями, отмеченными цифрами в кружках.

5. До операции указатель стека указывает на последнюю ячейку стека. Ее называют вершиной стека. Затем указатель стека декрементируется до 2207Н.

6. Указатель стека указывает на ячейку памяти 2207Н, и содержимое аккумулятора ($0101\ 0101_2$) загружается в стек по этому адресу.

7. Указатель стека декрементируется от 2207Н до 2206Н.

8. Указатель стека указывает на ячейку памяти 2206Н. Содержимое регистра состояния ($1111\ 1111_2$) загружается по этому адресу.

Стек может продолжать расти, пока длится процесс загрузки в него (на рис. 9.9 показано, что стек растет вверх).Стек не имеет ограничений, за исключением тех, которые обусловлены наличием других программ в ОЗУ.

Обычно каждой команде загрузки в стек (PUSH)

позже будет соответствовать команда извлечения из стека (POP), по которой данные берутся из вершины стека. Поскольку стек является памятью типа LIFO (последний вошел — первый вышел), данные должны извлекаться из стека в порядке, обратном загрузке. На рис. 9.10, *в* и *г* подробно показаны операции извлечения из стека.

Рассмотрим команду POP на рис. 9.10, *в*. Аккумулятор и регистр состояния свободны до операции извлечения из стека. Следуем последовательности, указанной цифрами в кружках.

1. Указатель стека указывает на вершину стека, т. е. на адрес 2206H. Содержимое регистра состояния ($1111\ 1111_2$) извлечено из стека и переслано в АЛУ.
2. Указатель стека инкрементирован с 2206H до 2207H.
3. Указатель стека указывает на адрес 2007H стека. Вершина стека извлекается, и ее содержимое пересылается в аккумулятор АЛУ.
4. Указатель стека инкрементирован до 2208H и указывает теперь на следующий адрес извлечения из стека.

Содержимое аккумулятора и регистра состояния было восстановлено до тех значений, которые были до операции PUSH, показанной на рис. 9.10, *б*.

Затем (на рис. 9.10, *г)* содержимое регистра адреса/данных в свою очередь извлекается из стека. Снова последуем согласно заключенным в кружки цифрам:

5. Указатель стека указывает на вершину стека (адрес 2208H). Содержимое этой ячейки памяти стека извлекается и пересылается в младший байт пары

регистров *HL*.

6. Указатель стека инкрементируется до 2209Н.

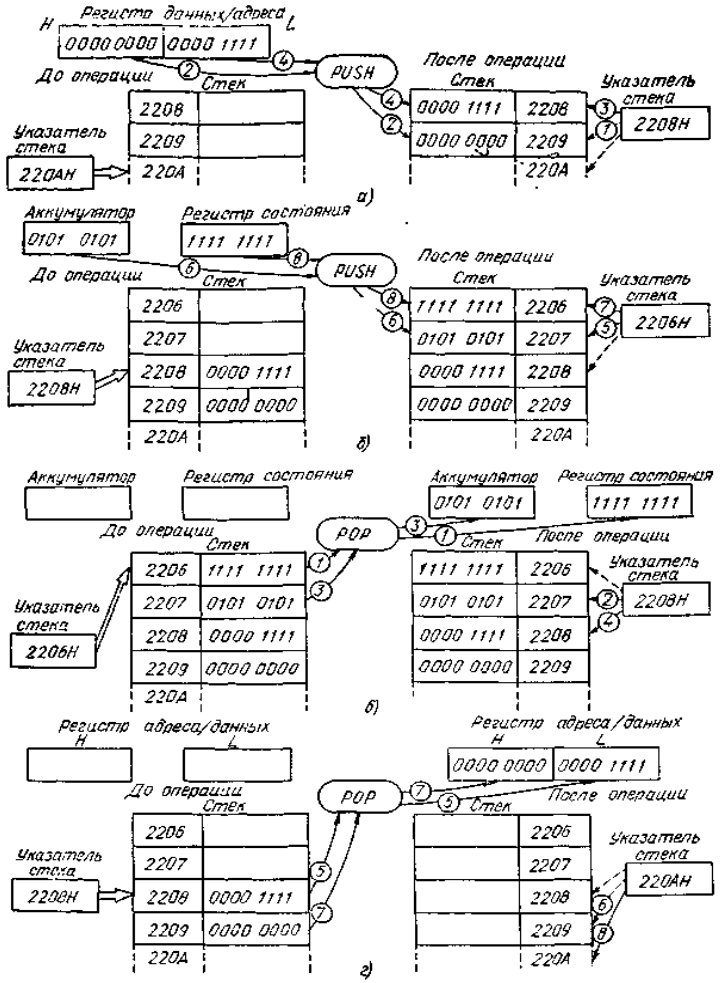


Рис. 9.10. Операции со стеком

а — помещение в стек содержимого регистра адреса/данных; б — помещение в стек содержимого аккумулятора и регистра состояния; в — извлечение из стека содержимого аккумулятора н регистра состояния; г ~ извлечение из стека содержимого регистра адреса/данных

7. Указатель стека показывает на вершину стека, т. е. теперь адрес 2209H, содержимое которого (0000 0000_2), передается в старший байт пары регистров *HL*.

8. Указатель стека инкрементируется от 2209H до 220AH для последующей операции загрузки в стек (PUSH) или извлечения из стека (POP).

Извлечение данных из стека и их восстановление в регистре адреса/данных является действием, обратным операции загрузки в стек (PUSH), выполненной на рис. 5.15, *а*. Команды PUSH и POP используются всегда совместно, однако между ними располагаются другие команды, которые меняют данные, содержащиеся в регистрах МП.

Наш типовой МП загружает в стек и извлекает содержимое пары регистров. Некоторые МП осуществляют эти операции только с однобайтовыми регистрами, единственной командой. В других МП указатель стека может указывать на пустую ячейку памяти по ближайшему большему по отношению к вершине стека адресу вместо указания на вершину стека, как в предыдущем случае. При этих условиях имеется возможность сохранить в памяти то, что программист загрузил в начале (адрес 220AH в нашем случае) в указатель стека для определения его адреса

Рассмотрим простейшие примеры программ пересылки данных.

Пример 1. Переслать данные, занимающие 2 байта, из памяти в регистровую пару *DE.*

Программа:

Адрес	Ассемблерный код	Комментарий
00	*LXI H, ADR*	; Установить непосредственный
01		; адрес, по которому хранится
02		; 2-й байт данных, в ; регистровую пару *HL*
03	*MOV E ,M*	; Переслать в *E* содержимое ; ячейки памяти, адресуемой парой *HL*
04	*INX H*	; Увеличить на 1 адрес, находящийся в *HL*
05	*M0V D ,M*	; Переслать в *D* содержимое ячейки памяти, адрес которой находится в *HL*
06	*END*	; Конец.

Пример 2. Переслать данные, занимающие 2 байта, из регистровой пары *DE* в память.

Программа:

Адрес	Ассемблерный код	Комментарий
00	*LKI H, ADR*	; Установить непосредственный
01		;адрес в регистровую
02		; пару *HL*
03	*M0V M, E*	; Переслать содержимое *E* в ячейку памяти с адресом, находящимся в *HL*
04	*INX H*	; Увеличить на 1 адрес в *HL*
05	*MOV M, D*	; Переслать содержимое *D* в ячейку памяти, адресуемую парой *HL*
06	*END*	; Конец.

Пример 3. Переслать данные длиной 2 байта из памяти в пару *HL*.

Программа:

Адрес	Ассемблерный код	Комментарий
00	*LXI H, ADR*	; Установить непосредственный
01		; адрес в регистровую
02		; пару *DE*
03	*LDAX D*	; Загрузить *A* содержимым ячейки памяти с адресом, находящимся в *DE*
04	*MOV L, A*	; Переслать содержимое *A* в *L*
05	*INX D*	; Увеличить на 1 адрес, содержащийся в *DE*
06	*LDAX D*	; Загрузить *A* содержимым ячейки памяти, адресуемой парой *DE*
07	*MOV H, A*	; Переслать содержимое *A* в *H*
08	*E ND*	; Конец.

Пример 4. Переслать данные длиной 2 бай та из пары *HL* в память.

Программа:

Адрес	Ассемблерный код	Комментарий
00	*LXI D, ADR*	; Установить непосредствен-
01		; ный адрес в регистровую
02		; пару *DE*
03	*MOV A, L*	; Переслать содержимое регистра *L* в *A*
04	*STAX D*	; Записать в ячейку памяти, адресуемую парой *DE*, содержимое *A*
05	*INX D*	; Увеличить на 1 адрес в *DE*
06	*MOV A, H*	; Переслать содержимое регистра *H* в *A*
07	*STAX D*	; Записать в ячейку памяти, адресуемую парой *DE*, содержимое *A*
08	*END*	; Конец.