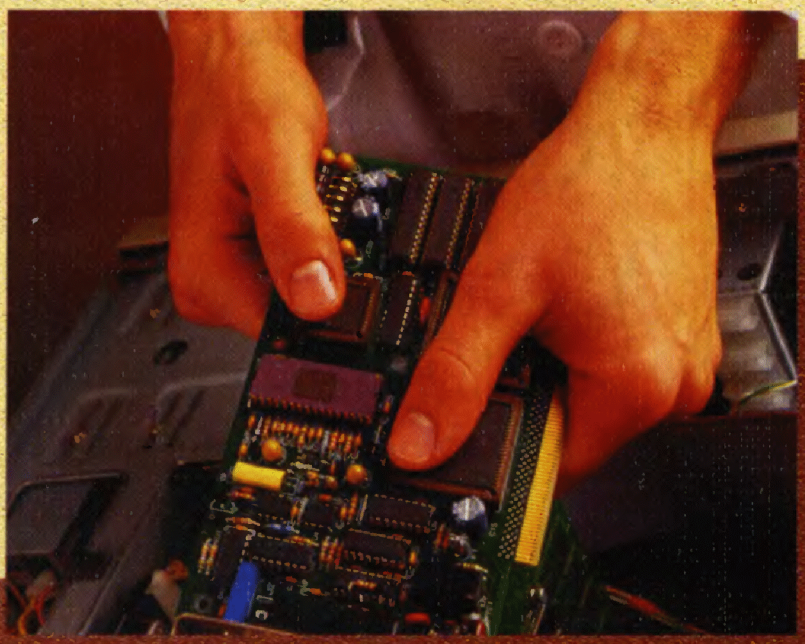
32.973.2
П 14

К.А. Палагута

МИКРОПРОЦЕССОРЫ INTEL 8080, 8085 (КР580ВМ80А, КР1821ВМ85А) и их программирование



ИЗДАТЕЛЬСТВО МГИУ

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ИНДУСТРИАЛЬНЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

К.А. Палагута

Микропроцессоры INTEL 8080, 8085 (КР580ВМ80А, КР1821ВМ85А) и их программирование

Учебное пособие

Москва 2007

УДК 004.42
ББК 32.973.26-04
П 14

Рецензенты:
Мартяков А.И., к.т.н., доцент МГИУ
Воронов Ю.А., к.т.н., доцент МИФИ

Палагута К.А.
Микропроцессоры INTEL 8080, 8085 (КР580ВМ80А, КР1821ВМ85А) и их программирование. – М.: МГИУ, 2007. -- 104 с.

П14

ISBN 978-5-276-01040-3

Рассмотрены архитектура, регистровая модель, временные диаграммы работы микропроцессора. Особое внимание уделяется описанию языка ассемблера микропроцессора КР580ВМ80А, рассматриваются формат различных команд и методы адресации, приводится большое количество примеров.

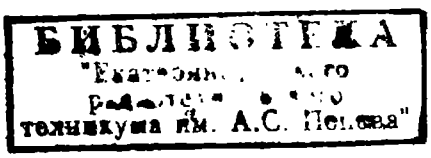
Предназначено для студентов, обучающихся по специальности 220301 (210200) «Автоматизация технологических процессов и производств в машиностроении» специализации 46 «Автоматические и электронные системы транспортных средств», и может быть использовано в курсах «Микропроцессоры и интерфейсные средства транспортных средств», «Микропроцессорные системы управления транспортных средств».

14876

УДК 004.42
ББК 32.973.26-04

ISBN 978-5-276-01040-3

© К.А. Палагута, 2007
© МГИУ, 2007
© ИДО МГИУ, 2007



Оглавление

Введение

Стремительное развитие микропроцессорной техники в конце XX и начале XXI веков привело к тому, что характеристики современных универсальных микропроцессоров в плане быстродействия, объема оперативной памяти и разрядности намного превосходят характеристики микропроцессора (МП) Intel 8080 (КР580ВМ80А).

В то же время изучение идей, заложенных в CISC микроконтроллерах, по мнению автора, целесообразно начинать с тех простых устройств, которые, во-первых, получили развитие в последующих поколениях не только универсальных микропроцессоров, но и микроконтроллеров; во-вторых, подкреплены серийно выпускаемыми лабораторными стендами.

Изучение принципов работы и программирования микропроцессора на примере гипотетического устройства требует некоторого переучивания при переходе к конкретному устройству, что приводит к необоснованной затрате времени и сил.

По этим причинам целесообразно использовать как отправную точку при освоении микропроцессорной техники, которой автор занимается несколько десятков лет, микропроцессор Intel 8080 (КР580ВМ80А), для которого существуют лабораторные стенды, серийно выпускаемые заводом «Протон» (г. Зеленоград).

Поскольку аппаратная часть микропроцессоров шагнула далеко вперед, то в учебном пособии основное внимание уделя-

ется изучению ассемблера МП КР580ВМ80А, что может быть полезно при изучении ассемблеров современных микроконтроллеров, которые рассматриваются в других разделах курса «Микропроцессоры и интерфейсные средства транспортных средств» для студентов специальности 220301(21020). Аппаратная часть рассматривается в главе 1 в той мере, в какой это необходимо для понимания работы данного микропроцессора и освоения его ассемблера.

Автор считает своим долгом выразить благодарность инж. Лукьяновой О.Ю. за помощь в подготовке материала учебного пособия.

Глава 1. Структура и функционирование МП КР580ВМ80А

1.1. Структурная схема микропроцессора К580ВМ80

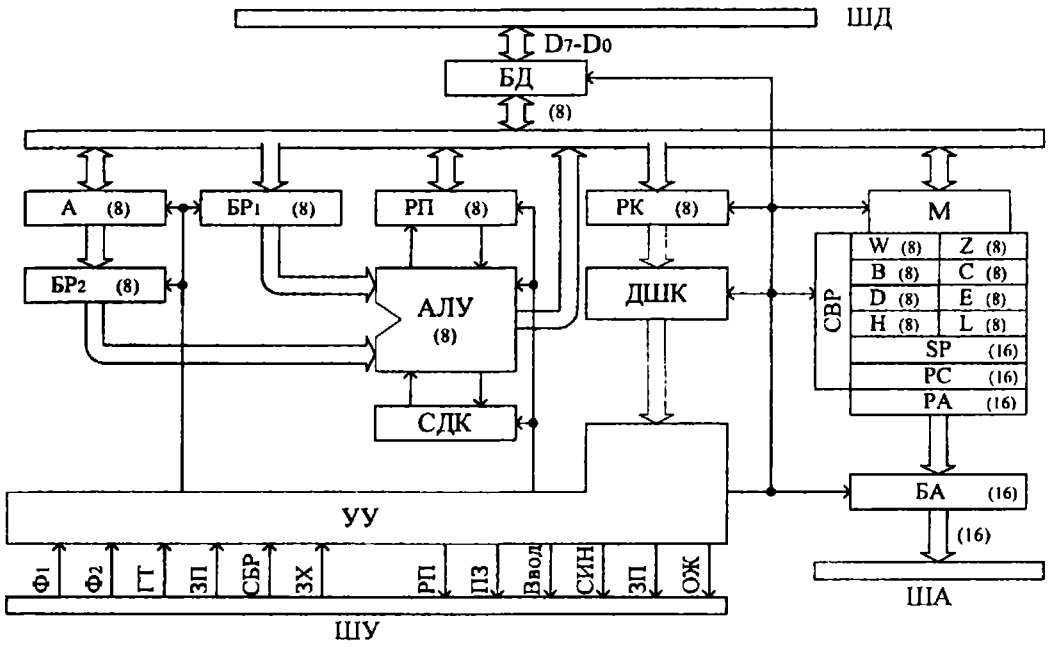


Рис. 1.1. Структурная схема микропроцессора

Структурная схема МП К580ВМ80А приведена на рис. 1.1, где используются обозначения на русском языке:

А – аккумулятор;

БД, БА – буфер шины данных, адреса;

БР1, БР2 – буферные регистры;

АЛУ – комбинационное арифметико-логическое устройство;

РП – регистр признаков;

СДК – схема десятичной коррекции результата;

РК – регистр команд;

ДШК - дешифратор команд;

УУ – устройство управления;

М – мультиплексор;

СВР – схема выборки регистров;

W, Z – программно-недоступные регистры;

B, C, D, E, H, L – регистры общего назначения;

SP – указатель стека (Stack Pointer);

PC – счетчик команд (Programm Connter);

ШД, ША, ШУ – шины данных, адреса и управления соответственно.

Рассмотрим входные выходные сигналы в русской и английской версиях (в английской версии обозначения сигнала присутствуют только заглавные буквы).

Входные сигналы:

- Ф1, Ф2 (F1, F2) – две перекрывающие последовательности импульсов синхронизации;
- ГТ (READY) – сигнал готовности внешнего устройства или памяти к обмену; используется при работе с медленными по отношению к МП устройствами;
- ЗП (INTerrupt) – сигнал запроса прерывания;
- СБР (RESET) – сигнал начальной установки (сброса);
- ЗХ (HOLD) – сигнал запроса прямого доступа к памяти (ПДП), в иных терминах – захвата магистрали.

Выходные сигналы:

- РП (INTerrupt Enable) – сигнал разрешения прерывания;
- ПЗ (Hold Acknowledge) – сигнал подтверждения прямого доступа к памяти или подтверждения захвата магистрали;
- Ввод (Data Bus IN) – сигнал чтения, его высокий (H) уровень свидетельствует о том, что двунаправленная шина данных находится в режиме приема информации;
- СИН (SYNChronization) – сигнал синхронизации, его высокий уровень свидетельствует о том, что по шине данных передается байт состояния, который используется для формирования некоторых управляющих сигналов;
- ЗП ($\overline{\text{WR}}$ite) – сигнал записи, его низкий (L) уровень означает, что двунаправленная шина данных находится в режиме выдачи информации;
- ОЖ (WAIT) – сигнал ожидания, его активный уровень свидетельствует о том, что процессор перешел в режим ожидания и выполняет такты ожидания.

1.2. Основные технические характеристики МП КР580ВМ80А

Универсальный микропроцессор с трехшинной организацией.

Разрядность шины данных – 8 бит.

Разрядность шины адреса – 16 бит.

Способность осуществлять обмен данными под управлением процессора, работать в режимах прямого доступа к памяти (ПДП) и прерываний.

Структура прерываний – радиальная, до 64 устройств.

Структура прямого доступа к памяти – радиальная, до 4 устройств.

Адресное пространство памяти и внешних устройств разделены (64 Кбайт памяти и 256 внешних устройств).

Форма представления чисел – дополнительный код с фиксированной запятой.

Имеет возможность работать с двоично-десятичными числами (BCD код).

Максимальная тактовая частота – 2,5 МГц.

Напряжения питания, В – +12; +5; –5.

Потребляема мощность – ≤ 1,25 Вт.

Технология изготовления – n-МОП.

Тип корпуса – 40-выводный DIP (2123.40-2).

Количество регистров общего назначения (РОН) – 7.

Количество команд – 244.

Количество методов адресации – 4.

1.3. Регистры МП КР580ВМ80А

Регистры являются важной составной частью любого микропроцессора. Они участвуют в реализации основных логических функций микропроцессора. Каждый регистр микропроцес-

сора может использоваться для временного хранения одного слова данных. Некоторые регистры имеют специальное назначение, другие многоцелевое. Регистры последнего типа называются *регистрами общего назначения* и могут использоваться программистом по его усмотрению.

В процессе ознакомления с каждым из основных регистров следует обращать внимание на то, какое влияние оказывает именно этот регистр на данные, проходящие «сквозь» микропроцессор.

С точки зрения пользователя и разработчика МПС важнейшими являются сведения о системе команд микропроцессора. В этом смысле МП представляет собой массив программно-доступных регистров, над содержимым которых выполняются указанные в командах операции.

Программно-доступными являются:

8-битовые регистры A, B, C, D, E, H, L;

16-битовые регистры слова состояния PSW (регистры A и F), пары регистров BC, DE, HL, указатель стека SP, программный счетчик PC;

отдельные признаки регистра флагов FL;

триггер разрешения прерывания INTE

A – аккумулятор – главный регистр микропроцессора при различных манипуляциях с данными. Большинство арифметических и логических операций осуществляется путем использования АЛУ и аккумулятора.

PC – счетчик команд – это один из наиболее важных реги-

стров микропроцессора. Как известно, программа – это последовательность команд, хранимых в памяти микроЭВМ и предназначенных для того, чтобы инструктировать машину, как решать поставленную задачу. Для корректного выполнения задачи команды должны поступать в строго определенном порядке. На счетчике команд лежит ответственность следить за тем, какая команда выполняется, а какая подлежит выполнению следующей. Счетчик команд имеет больше разрядов, чем длина слова даны микропроцессора. В любой из 65 536 областей памяти микроЭВМ может находиться информация о том или ином шаге программы, т.е. в пределах диапазона значений адресов от 0 до 65 535 программа может начаться и закончиться в любом месте. Чтобы обратиться по любому из этих адресов, счетчик команд должен располагать 16 двоичными разрядами.

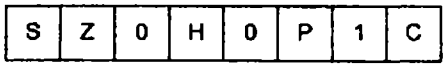
Перед выполнением программы счетчик команд необходимо загрузить числом-адресом области памяти, содержащей первую команду программы.

После извлечения команды из памяти микропроцессор автоматически дает приращение содержимому счетчика команд. Это приращение счетчик команд получает как раз в тот момент, когда микропроцессор начинает выполнять команду, только что извлеченную из памяти. Следовательно, начиная с этого момента, счетчик команд «указывает», какой будет следующая команда. *Счетчик команд содержит адрес следующей команды на протяжении всего времени выполнения текущей команды.* Об этом важно помнить, потому что, программируя работу

микропроцессора, вы можете столкнуться с необходимостью использования текущего значения счетчика команд.

FL – регистр флагов. Этот регистр предназначен для хранения результатов некоторых проверок, осуществляемых в процессе выполнения программы. Запоминание результатов упомянутых проверок позволяет использовать программы, содержащие переходы.

В микропроцессоре К580ВМ80 регистр флагов имеет 8 разрядов, 5 из которых и хранят результат проверки.



7 0

Разряд Ts =1, если результат отрицательный (первый бит результата = 1).

Разряд Tz =1, если результат = 0.

Разряд Th =1, если был перенос из старшей тетрады в младшую.

Разряд Tp =1, если число единиц в результате четно.

Разряд Tc =1, если был перенос или заем.

Пример

Результат получился отрицательный. Следовательно, Ts =1;

– не равен 0, следовательно, Tz = 0;

– переноса из старшей тетрады в младшую не было, сле-

довательно, Th = 0;

– число единиц в результате данного примера = 6, т.е. четно, следовательно, Tp = 1;

– перенос был, следовательно, Tc = 1.

Таким образом, флаговый регистр = 10000111 = 87h

Регистр	*флагов*							
A	**S**	**Z**	**0**	**H**	**0**	**P**	**1**	**C**
B	**C**							
D	**E**							
H	**L**							
SP								
PC								

Рис. 1.2. Регистровая модель микропроцессора КР580ВМ80А

Команда МП – это такое двоичное слово, которое, будучи прочитано микропроцессором, заставляет его выполнить определенные действия. Другие, отличные от команды двоичные слова подобные действия в МП вызывать не могут.

Набор команд МП – это все его команды.

Команда содержит информацию двух видов:

– она должна сообщать МП, что делать (сложить, переслать и т.п.), т.е. должна содержать код операции (коп);

– должна указывать адрес, т.е. местоположение обрабатываемых данных.

Для облегчения запоминания различных команд применяется их мнемоническое обозначение – сокращенная запись названия команды. Это обычно три буквы названия операции. Сочетание сокращенного буквенного обозначения кода операции с

числовой формой записи адреса является одной из наиболее удобных форм записи команды. Мнемоническое обозначение команды является составной частью при использовании языка ассемблера.

1.4. Обозначения, используемые в системе команд МП К580ВМ80

Таблица 1.1

<table>
<tr><th>Обозначение</th><th colspan="2">Содержание</th></tr>
<tr><td rowspan="11">< Bi >

R</td><td colspan="2">i-ый байт команды</td></tr>
<tr><td colspan="2">Обозначение одного из регистров</td></tr>
<tr><td>Код регистра</td><td>Наименование регистра r</td></tr>
<tr><td>000</td><td>B</td></tr>
<tr><td>001</td><td>C</td></tr>
<tr><td>010</td><td>D</td></tr>
<tr><td>011</td><td>E</td></tr>
<tr><td>100</td><td>H</td></tr>
<tr><td>101</td><td>L</td></tr>
<tr><td>110</td><td>M (память) (HL)</td></tr>
<tr><td>111</td><td>A (аккумулятор)</td></tr>
<tr><td colspan="3">Если r = M, то это значит, что источником или приемником информации является ячейка оперативной памяти, адрес которой хранится в паре регистров (HL)</td></tr>
<tr><td rowspan="6">F (FL)</td><td colspan="2">Регистр признаков (флагов), состоящий из триггеров:</td></tr>
<tr><td>Триггеры регистра</td><td>Условие истинности</td></tr>
<tr><td>Tc (перенос)</td><td>Наличие переноса или заема</td></tr>
<tr><td>Tz (нуль)</td><td>Результат равен нулю</td></tr>
<tr><td>Ts (знак)</td><td>Старший разряд результата равен 1</td></tr>
<tr><td>Tp (четность)</td><td>Число единиц в результате четно</td></tr>
<tr><td>Th (полуперенос)</td><td>Наличие переноса из старшей тетрады в младшую</td></tr>
<tr><td>(r)</td><td colspan="2">Содержимое регистра r</td></tr>
<tr><td>[(r)]</td><td colspan="2">Ячейка памяти, адрес которой (r)</td></tr>
</table>

Продолжение табл. 1.1

Обозначение	Содержание
∧	Логическая операция «И»
∀	Исключающее «ИЛИ»
∨	Логическая операция «ИЛИ»
r_m	m-ый бит регистра r
SP	Указатель стека
PC	Счетчик команд
←	Пересылка

1.5. Выполнение команд в МП КР580ВМ80А (i8080)

Каждая команда в МП выполняется на протяжении командного цикла. Командный цикл состоит из цикла выборки команды и цикла выполнения команды (рис. 1.3).

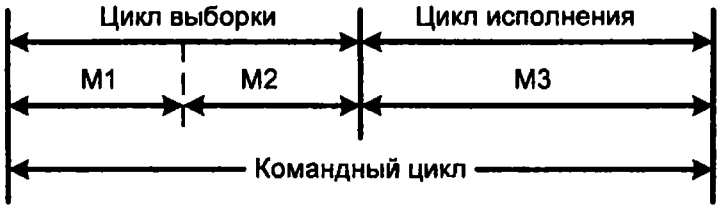


Рис. 1.3. Пример командного цикла 8-разрядного МП

Продолжительность цикла выборки команды зависит от формата команды (количества байт в машинном коде команды). Команды занимают от одного до трех байт в программной памяти. Многобайтные команды хранятся в соседних ячейках памяти. Для выборки однобайтной команды (например, сложения содержимого аккумулятора A и регистра B-ADD B) необходимо

одно обращение к памяти, для выборки трехбайтной команды (например, вызова подпрограммы по адресу ADDR – CALL ADDR) – три обращения. Продолжительность цикла исполнения команды зависит от способа адресации операндов. Так, при выполнении команд с регистровой адресацией не нужно дополнительное обращение к памяти для чтения операнда, в командах с косвенной адресацией такое обращение необходимо. В связи с этим продолжительность командного цикла в МП i8080 различна для различных команд и определяется количеством обращений к памяти или к внешнему устройству.

Интервал, на протяжении которого осуществляется одно обращение процессора к памяти или к внешнему устройству, определяется как ***машинный цикл М.*** Итак, командный цикл процессора состоит из некоторого количества машинных циклов (в зависимости от типа команды). В приведенном на рис. 1.3 примере цикл выборки состоит из двух машинных циклов (М1 и М2), а цикл исполнения – из одного машинного цикла (М3). В команде может быть от одного (для однобайтных команд с регистровой адресацией) до пяти (для трехбайтных сложных команд) машинных циклов.

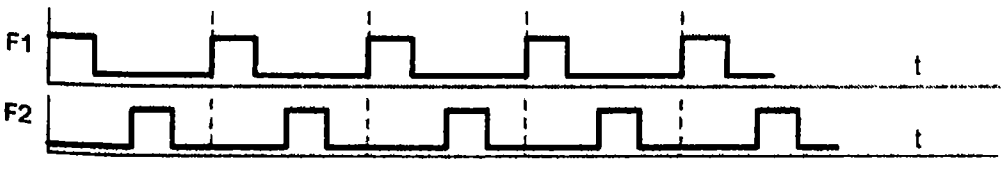


Рис. 1.4. Последовательности импульсов

БИБЛИОТЕКА
"Радиотехнический
техникума им. А.С. Попова"

Машинный цикл, в свою очередь, разбивается на некоторое количество машинных тактов *T*, на протяжении каждого из которых выполняется элементарное действие (микрооперация) в процессоре. Количество тактов в цикле определяется кодом команды и лежит в пределах от 3 до 5. Продолжительность такта задается периодом импульсов синхронизации и определяется как интервал времени между фронтами двух соседних импульсов последовательности *F1* (рис. 1.4). Таким образом, командный цикл МП i8080 состоит из некоторого количества машинных циклов, а каждый машинный цикл – из определенного количества тактов, на протяжении которых выполняются те или иные элементарные действия в процессоре.

В зависимости от действий, выполняемых МП, различают следующие типы машинных циклов:

- ВЫБОРКА (чтение первого байта команды);
- ЧТЕНИЕ ПАМЯТИ (чтение второго и третьего байтов команды, чтение операнда);
- ЗАПИСЬ В ПАМЯТЬ;
- ЧТЕНИЕ СТЕКА;
- ЗАПИСЬ В СТЕК;
- ВВОД данных из внешнего устройства;
- ВЫВОД данных на внешнее устройство;
- ПРЕРЫВАНИЕ;
- ОСТАНОВ;
- ПРЕРЫВАНИЕ ПРИ ОСТАНОВЕ.

Каждый машинный цикл процессора идентифицируется байтом, называемым ***байтом состояния.*** Байт состояния выдается на шину данных в начале каждого машинного цикла и сопровождается одновременной выдачей сигнала SYNC на одноименный контакт БИС МП. Байт состояния несет информацию о последующих действиях процессора. Его можно запомнить по сигналу SYNC и сформировать такие управляющие сигналы, которые не удалось вывести в явном виде на контакты БИС МП из-за ограниченного количества контактов микросхемы.

Временная диаграмма машинного цикла ВЫБОРКА (ЧТЕНИЕ ПАМЯТИ) для процессора i8080 показана на рис. 1.5, а цикл ЗАПИСЬ В ПАМЯТЬ – на рис. 1.6. Такты отсчитываются по передним фронтам последовательности *F1*, а микрооперации в каждом такте определяются передним фронтом последовательности *F2*. При тактовой частоте 2 МГц продолжительность такта равна 0,5 мкс.

Сигналы на линиях шин A15–A0 (D7–D0) изображены на рис. 1.5 на одной временной диаграмме в виде линий – L- и H-уровень одновременно. Таким образом, на диаграмме указан лишь тип информации на шине, например присутствие на ней адреса или данных. Пунктирной линией обозначено высокоимпедансное состояние линий шин (z-состояние).

В первом машинном такте *T1* на шину адреса (линии *A15*–A0) выдается адрес – содержимое указателя команд IP, если выполняется цикл ВЫБОРКА, или содержимое регистра-указателя адреса – если выполняется цикл ЧТЕНИЕ ПАМЯТИ. Вместе с тем на

шину данных (линии D7–D0) выдается байт состояния, а также формируется сигнал SYNC на одноименном выводе БИС МП.

Во втором такте *T2* заканчивается поступление байта состояния и сигнала SYNC, продолжительность которых равняется одному такту. В машинном цикле ВЫБОРКА содержимое IP увеличивается для адресации следующего байта команды или следующей команды. В этом же такте устройство управления МП производит анализ сигналов на входах READY и HOLD, а также контроль выполнения команды останова HLT. Если память или внешнее устройство не готовы к обмену (READY=0) вследствие того, что поступил запрос ПДП (HOLD=1) или выполняется команда останова HLT, то обмен данными осуществляться не может, и процессор переходит в один из режимов – *ожидание, захват шин* или *останов*. В этих режимах осуществляется ожидание сигнала на протяжении нескольких тактов ожидания T_u, количество которых определяется внешними сигналами. На рис. 1.5 в такте *T2* сигнал READY равняется логическому нулю, а в такте T_u – логической единице.

В такте *T3* в зависимости от типа машинного цикла осуществляется обращение к памяти, стеку или внешнему устройству. В результате в МП вводится (см. рис. 1.5) или из него выводится (см. рис. 1.6) байт команды, адреса или данных. В зависимости от типа команды машинный цикл может содержать такты *T4* и *T5* (например, если для выполнения команды нужна обработка операндов). В последнем такте команды (*T3*, *T4* или *T5*) анализируется наличие сигнала запроса прерывания INT.

Если прерывание разрешено, то процессор переходит к машинному циклу ПРЕРЫВАНИЕ.

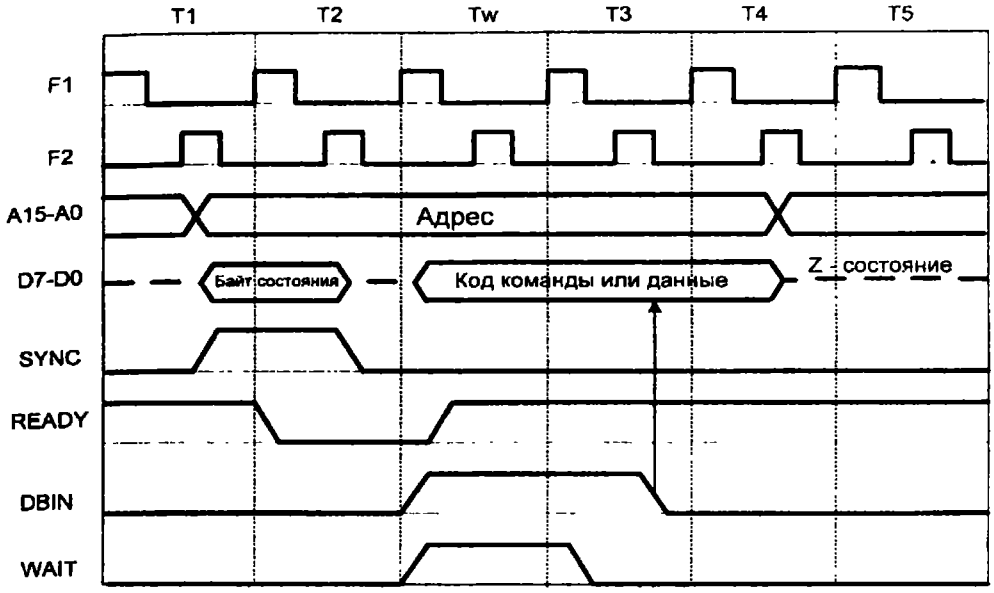


Рис. 1.5. Цикл ВЫБОРКА (ЧТЕНИЕ ПАМЯТИ)

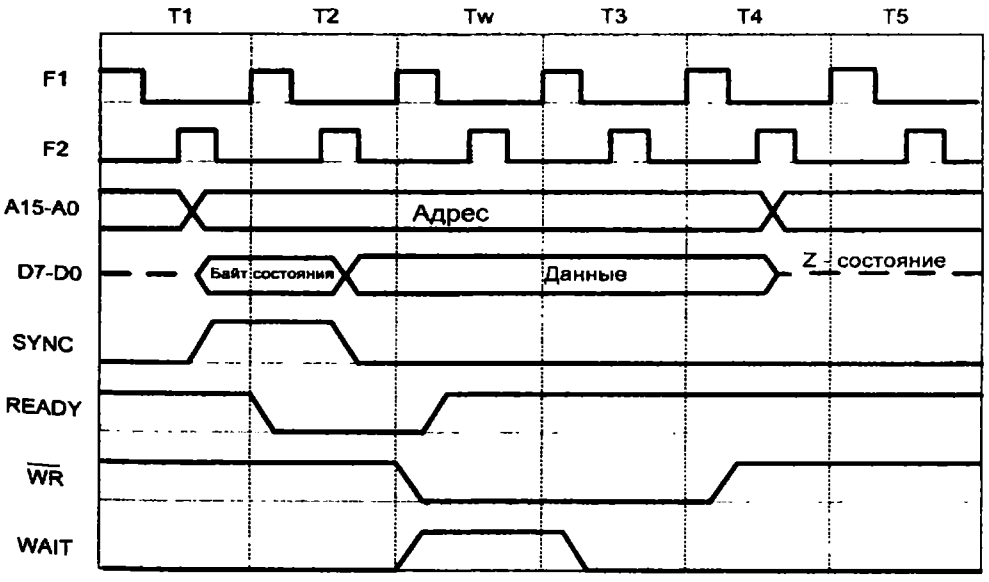


Рис. 1.6. Цикл ЗАПИСЬ В ПАМЯТЬ

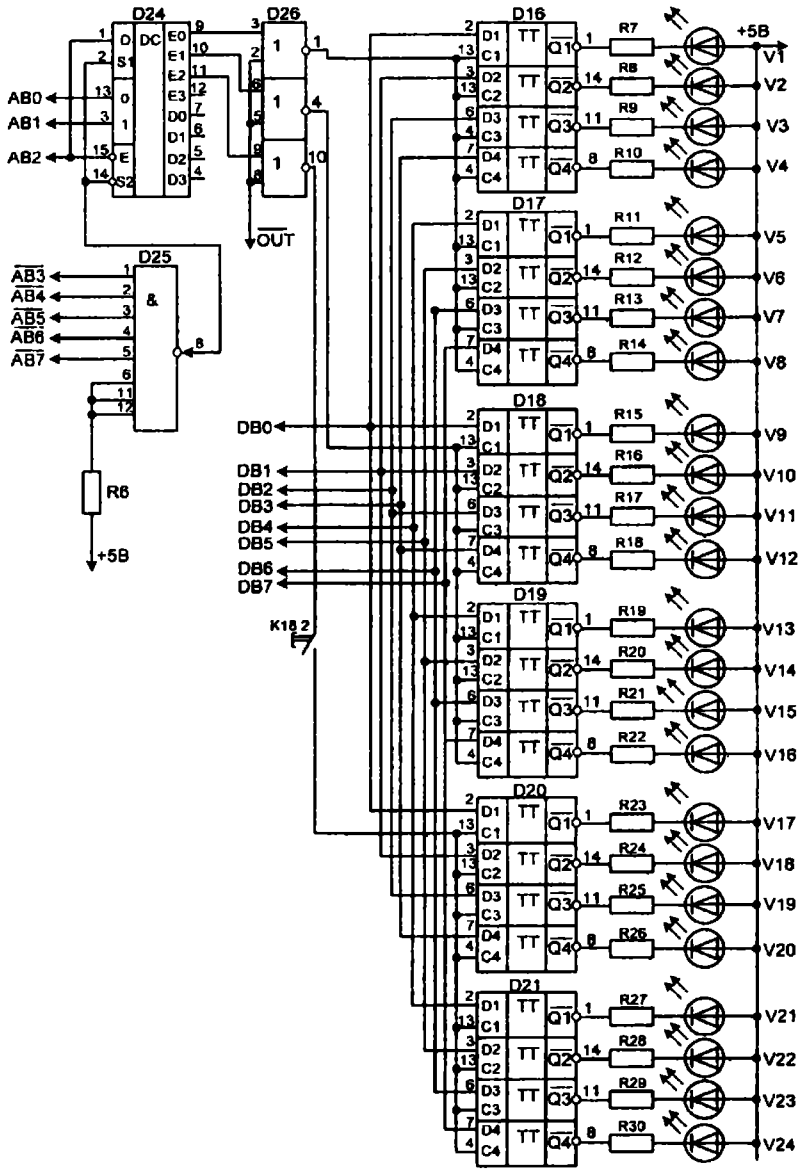


Рис. 1.7. Схема индикации

На рис. 1.7 приведена схема индикации, являющаяся портом ввода, который загружается данными из аккумулятора по команде OUT.

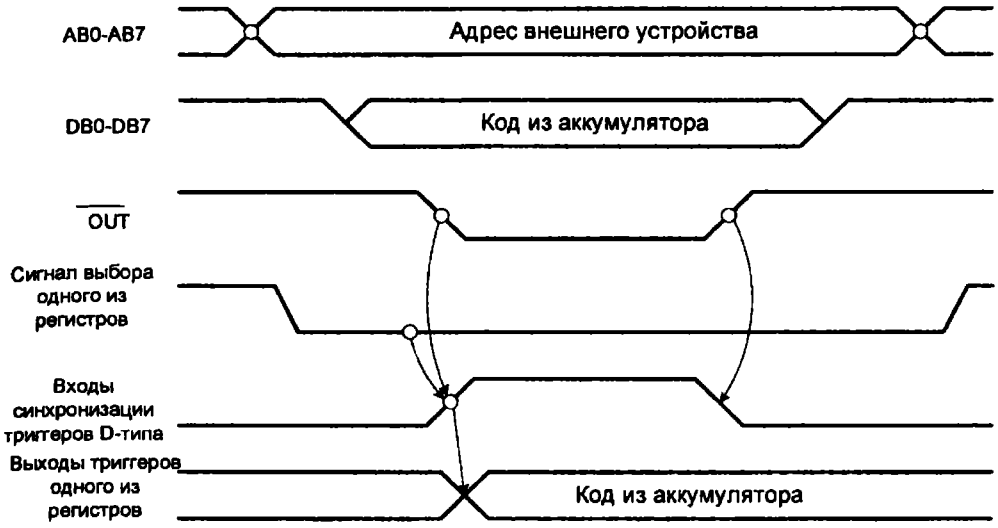


Рис. 1.8. Временная диаграмма работы порта вывода

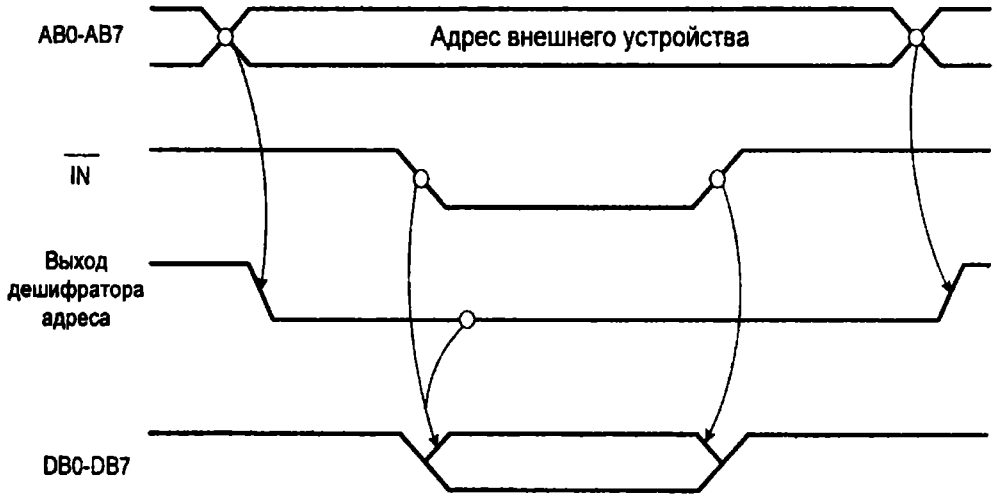


Рис. 1.9. Временная диаграмма работы порта ввода

Схема индикации микроЭВМ подключается к линиям DB0–DB7 шины данных, линиям AB0–AB7 шины адреса и линиям IN и OUT шины управления. Схема индикации выполнена на базе трех 8-разрядных регистров (микросхемы D16–D21). Каждый из этих регистров выполнен из двух микросхем К155ТМ7. Эта микросхема представляет собой четыре

D-триггера с прямыми и инверсными выходами. Входы триггеров подключены к линиям шины данных. Светодиоды подключены к инверсным выходам триггеров, поэтому они зажигаются, если в соответствующий триггер записывается 1, и гаснут, если записывается 0.

Микросхема D24 вместе с микросхемой D25 составляют схему дешифрации адреса для выбора внешних устройств ввода и вывода.

При выполнении одной из команд OUT во время ее третьего цикла на шинах данных, адреса и управления вырабатываются сигналы, изображенные на рис. 1.8. При этом вырабатывается сигнал на соответствующем выходе дешифратора D24. Сигнал с выхода дешифратора подается вместе с сигналом OUT на входы микросхемы D26 и формирует на ее выходе положительный импульс. Этот импульс, поступая на входы синхронизации одного из регистров, своим фронтом (переход из 0 в 1) переписывает код с шины данных в триггеры, а своим срезом фиксирует его там. Таким образом, 8 бит кода, содержащегося в аккумуляторе перед выполнением команды OUT, высвечиваются на индикаторах одного из регистров.

Временная диаграмма работы порта ввода приведена на рис. 1.9.

1.6. Классификация команд МП КР580ВМ80А

Ассемблер команд МП КР580ВМ80А содержит 244 команды, которые классифицируются по трем основным признакам:

- длина команды;
- функциональное назначение;
- метод адресации.

По длине команды делятся на однобайтовые, двухбайтовые, трехбайтовые (рис. 1.10).

При этом первый байт команды всегда содержит код операции (операционный код), а второй и третий байты отводятся под данные или адрес.

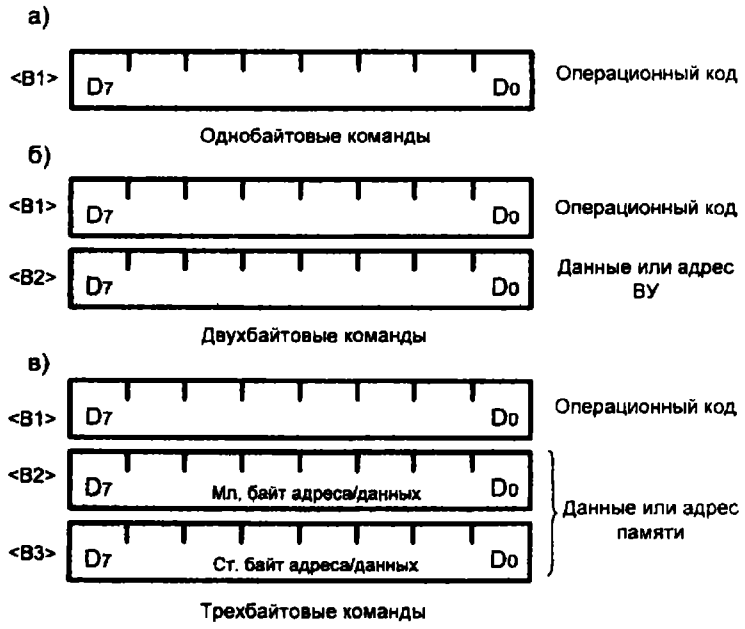


Рис. 1.10. Классификация команд МП КР580ВМ80А по длине

Метод адресации – это метод определения данных, участвующих в операциях, или иначе говоря, способ определения операндов.

Для составления команды важно знать особенности процедур, позволяющих преобразовать информацию об адресах команд и данных в физические адреса участков памяти машины.

Для КР580ВМ80А существуют следующие методы адресации:

- непосредственная;
- прямая;
- регистровая;
- косвенная.

Непосредственная адресация является наиболее экономичным методом хранения и поиска информации, поскольку необходимые данные содержит сама команда.

Эти данные содержатся во втором и третьем байтах трехбайтовой команды или во втором байте двухбайтовой команды. В случае трехбайтовой команды младшие разряды 16-битового числа содержатся во втором байте команды, а старшие – в третьем (рис. 1.11).

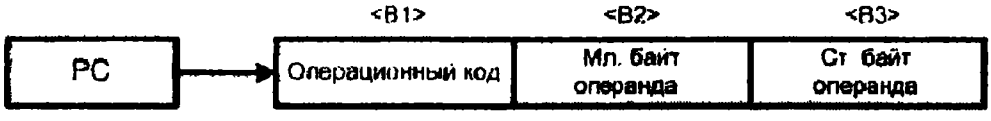


Рис. 1.11. Непосредственная адресация

Прямая адресация является менее экономичной. В этом случае во втором и третьем байтах команды содержится полный 16-разрядный адрес памяти.

Младшим байтом адреса является <B2>, старшим <B3> (рис. 1.12). Таким образом, можно адресоваться к любой ячейке адресного пространства памяти.

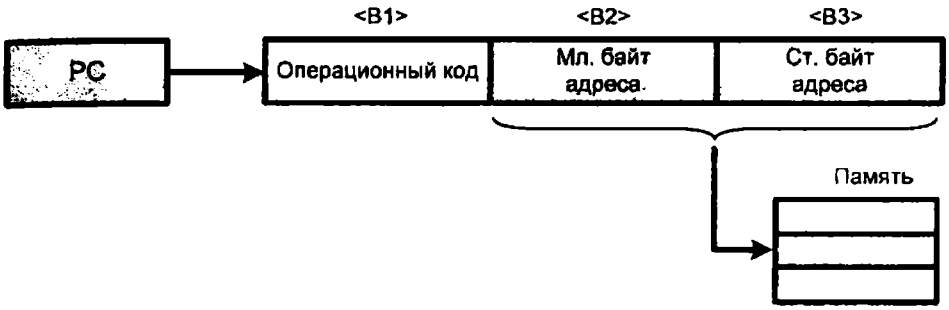


Рис. 1.12. Прямая адресация

При *регистровой адресации* код команды содержит указание на регистр или пару регистров, в которых содержатся данные (операнды). Используемые в регистровой адресации команды являются однобайтовыми (рис. 1.13). Возможность указания пары регистров в однобайтовой команде реализуется за счет того, что адреса регистров являются трехразрядными двоичными кодами.

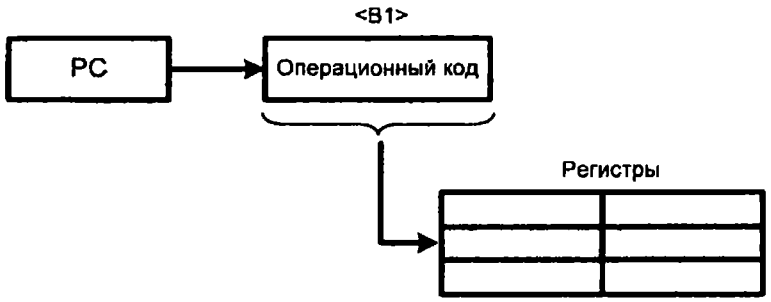


Рис. 1.13. Регистровая адресация

Косвенная адресация отличается от регистровой тем, что в регистровой паре, определяемой кодом команды, содержатся не данные, а полный 16-разрядный адрес ячейки памяти, в которой находятся эти данные.

Старший байт адреса записывается в первом регистре пары, а младший байт – во втором. Обычно указателем адреса при косвенной адресации являются пара регистров HL, но иногда используются пары BC и DE.

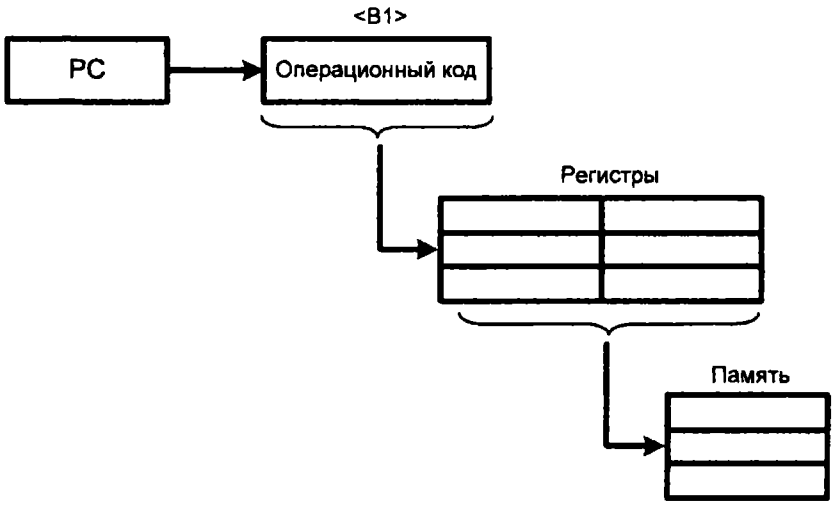


Рис. 1.14. Косвенная адресация

По функциональному назначению команды делятся на следующие основные группы:

- команды пересылки;
- арифметические и логические команды;
- команды переходов;
- команды работы с подпрограммами;
- команды управления.

Глава 2. Команды пересылки (перемещения) данных

Это команды передачи данных в МПС из одного места в другое. К числу областей хранения информации относятся как ячейки памяти, так и регистры. В зависимости от того, какие устройства микроЭВМ участвуют в пересылке данных, различают следующие команды: загрузка, пересылка и запись в память. Команды пересылки данных могут быть одно-, двух- или трехбайтовыми. Все команды пересылки, за исключением команды POP PSW, не изменяют содержимого регистра признаков. Приращение программного счетчика (PC) равно числу байтов в команде.

2.1. Пересылка из регистра в регистр

MOV ri, rj	**ri ← rj**	**<B1>**	01	ri	rj
			7		0

По этой команде в регистр *ri* загружается копия данных, содержащихся в регистре *rj*. В качестве регистров могут выступать: аккумулятор (A), РОН B, C, D, E, H, L, имеющие свой трехразрядный двоичный код.

Команда выполняется за один цикл, содержащий 5 тактов. В качестве одного из регистров можно выбрать память (M). В этом случае команда выполняется за два цикла (7 тактов), при этом данные пересылаются в регистр из ячейки ЗУ с адресом, хранящимся в паре регистров HL, или в обратном направлении.

Примеры

015A) MOV B, A

По этой команде данные, находящиеся в аккумуляторе/регистр, будут переданы в регистр *B*. После выполнения команды в аккумуляторе и в регистре *B* содержатся одни и те же данные.

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
015A	47	47
A	DB	DB
B	AF	DB
PC	015A	015B

После выполнения команды содержимое регистра признаков останется без изменения.

016A) MOV M, A

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
016A	77	77
H	0E	0E
L	AD	AD
A	5B	5B
0EAD	12	5B
PC	016A	016B

2.2. Непосредственная пересылка

MVI ri, Данные **ri ← <B2>** **<B1>**

00	ri	110

<B2>

Данные

7 0

По этой команде в регистр *ri* загружаются данные, которые находятся во втором байте команды. Команда двухбайтовая, выполняется за два машинных цикла (7 тактов). Если в качестве регистра используется память (М), т.е. *ri =M*, то данные пересылаются в ячейку памяти [(H)(L)]. В этом случае команда выполняется за три цикла (10 тактов). При выполнении команды содержимое триггеров признаков не меняется.

Пример

0148) MVI B, 25

По этой команде данные, находящиеся во втором байте команды (25 H), загружаются в регистр *B*, имеющий двоичный код 000.

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0148	06	06
0149	25	25
B	B5	25
PC	0148	014A

2.3. Непосредственная загрузка пары регистров

LXI ri, Данные **ri ← <B3> <B1>**
ri+1 ← <B2> <B2>
<B3>

00	ri	001
Мл. байт данных (7–0)		
Ст. байт данных (15–8)		

7 0

ri – код старшего регистра пары (B, D, H).

При *ri* = 110: SP ← <B3><B2>, т. е. загружается указатель стека.

По этой команде в 16-разрядную пару регистров BC, DE или HL заносятся данные, содержащиеся во втором и третьем байтах команды, причем данные заносятся соответственно в младший и старший регистры пары. Это 3-байтовая команда, выполняемая за три цикла (10 тактов). При выполнении команды содержимое триггеров признаков не меняется.

Пример

0124) LXI H, 48A7

По этой команде данные, находящиеся во втором байте команды (число A7 H), будут переданы в регистр L, а число, записанное в третьем байте (48 H), – в регистр H. Содержимое ячеек памяти при этом не меняется, как и регистр признаков.

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0124	21	21
0125	A7	A7
0126	48	48
H	DE	48
L	AD	A7
PC	0124	0127

2.4. Запоминание / загрузка аккумулятора и пары HL

а) **STAX B** **[(B)(C)] ← (A)** **<B1>** 00 | 000 | 010

б) **STAX D** **[(D)(E)] ← (A)** **<B1>** 00 | 010 | 010

в) **LDAX B** **(A) ← [(B)(C)]** **<B1>** 00 | 001 | 010

г) **LDAX D** **(A) ← [(D)(E)]** **<B1>** 00 | 011 | 010

7 0

По командам группы а) и б) содержимое аккумулятора запоминается в оперативной памяти по адресу, хранимому в паре регистров BC (а) или DE (б). По командам группы в) и г) содержимое оперативной памяти из ячейки, адресом которой является содержимое пары регистров BC (в) или DE (г), загружается в аккумулятор.

Команды группы а–г являются однобайтовыми и выполняются за два машинных цикла (7 тактов). Для того чтобы можно было воспользоваться этими командами, в соответствующую пару регистров необходимо предварительно загрузить нужный адрес.

Примеры

00B1) STAX D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
00B1	12	12
A	49	49
D	66	66
E	99	99
6699	E3	49
PC	00B1	00B2

По этой команде содержимое аккумулятора (A) = 49 H будет скопировано в ячейку памяти с адресом 6699 H, хранимым в паре регистров DE.

0800) STAX B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	02	02
B	0A	0A
C	11	11
A	82	82
0A11	BE	82
FL	02	02
PC	0800	0801

0801) LDAX B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0801	0A	0A
B	0A	0A
C	11	11
A	82	BF
0A11	BE	BE
FL	02	02
PC	0801	0802

0802) LDAX D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0802	1A	1A
D	66	66
E	99	99
6699	E3	E3
A	49	E3
FL	02	02
PC	0802	0803

д)

STA Адрес [<B3><B2>] ← (A)

<B1>	00	110	010
<B2>	Мл. байт адреса		
<B3>	Ст. байт адреса		

е)

LDA Адрес (A) ← [<B3><B2>]

<B1>	00	111	010
<B2>	Мл. байт адреса		
<B3>	Ст. байт адреса		

7 0

По командам д) и е) содержимое аккумулятора запоминается в ячейке памяти, адрес которой приводится во втором и

третьем байтах команды (д), или содержимое ячейки памяти, адрес которой приводится во втором и третьем байтах команды, передается в аккумулятор (е).

Команды этой группы трехбайтовые, выполняются за четыре цикла (13 тактов).

1F00) STA 0F1C

По этой команде содержимое аккумулятора скопируется в ячейке памяти с адресом OF1C.

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
1F00	32	32
1F01	1C	1C
1F02	0F	OF
0F1C	2A	05
A	05	05
PC	1F00	1F03

После выполнения операции и в ячейке OF1C, и в аккумуляторе будет записано одно и то же число.

0803) LDA 0F1C

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0803	32	32
0804	1C	1C
0805	0F	0F
0F1C	2A	2A
A	05	2A
FL	0803	0806
PC	02	02

ж)

SHLD Адрес	[<B3><B2>] ← (L)	<B1>	00	100	010
	[<B3><B2>+1] ← (H)	<B2>	Мл. байт адреса		
		<B3>	Ст. байт адреса		

з)

LHLD Адрес	(L) ← [<B3><B2>]	<B1>	00	101	010
	(H) ← [<B3><B2>+1]	<B2>	Мл. байт адреса		
		<B3>	Ст. байт адреса		
			7		0

По командам этой группы происходит передача данных из пары регистров HL в две соседние ячейки оперативной памяти (ж) или в обратном направлении (з). 16-разрядный адрес первой ячейки приводится во втором и третьем байтах команды; адрес второй ячейки вычисляется в МП с помощью инкремента (увеличения на единицу) содержимого второго и третьего байтов команды.

Команды этой группы трехбайтовые, выполняются за пять циклов (16 тактов). При выполнении команд параграфа 2.4 содержимое триггеров признаков не изменяется.

0F41) SHLD 32AF

При выполнении этой команды содержимое пары регистров HL будет скопировано в соседних ячейках памяти 32AF и 32BO. При этом содержимое (HL) не изменяется.

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0F41	22	22
0F42	AF	AF
0F43	32	32
32AF	4B	5D
32B0	8D	3F
H	3F	3F
L	5D	5D
PC	0F41	0F44

0806) LHLD 6AFF

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0806	2A	2A
0807	FF	FF
0808	6A	6A
H	07	FD
L	AF	33
6AFF	33	33
6B00	FD	FD
FL	02	02
PC	0806	0809

2.5. Ввод из пары регистров в стек

PUSH ri	**<B1>**	11	**ri**	101
		7		0

ri – код старшего регистра пары (B, D, H).

[SP-1] ←(ri); [SP-2] ← (ri+1); SP ← (SP)-2.

По этой команде содержимое указателя стека (SP) автоматически уменьшается на 1 и в ячейку памяти, адрес которой равен [SP-1] запишется содержимое старшего регистра пары РОН.

Затем содержимое указателя стека еще раз уменьшится на 1 и в соседнюю ячейку памяти с адресом [SP-2] запишется содержимое младшего регистра выбранной пары РОН. После второго уменьшения содержимое указателя стека останется без изменения до следующего обращения к нему. Для пар регистров BC, DE, HL старшими являются регистры B, D, H. Содержимое регистра признаков не меняется. Выполняется за три цикла (11 тактов).

Пример

01BE) PUSH D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
01BE	D5	D5
SP	0BB0	0BAE
0BAE	00	12
0BAF	00	12
D	12	6E
E	6E	6E
PC	01BE	01BF

080A) PUSH H

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
080A	E5	E5
SP	0BAE	0BAC
0BAC	62	A5
0BAD	01	FC
H	FC	FC
L	A5	A5
FL	02	02
PC	080A	080B

2.6. Ввод A и F в стек

PUSH PSW	<B1>	F5

[SP-1] ←(A); [SP-2] ← (F); SP ← (SP)-2 7 0

По этой команде содержимое указателя стека (SP) автоматически уменьшается на 1 и в ячейку памяти с адресом [SP-1] записывается содержимое аккумулятора (A). Затем содержимое указателя стека вновь уменьшается на 1 и является адресом ячейки памяти [SP-2], куда записывается содержимое регистра признаков (F). После этого содержимое указателя стека не меняется до очередного обращения к нему.

Команды 2.5 и 2.6 используются при входе в подпрограмму, когда необходимо сохранить содержимое некоторых регистров.

Команда выполняется за три цикла (11 тактов). При этом содержимое регистра признаков не меняется.

Примеры

02A1) PUSH PSW

Адрес	Код до выполнения операции	Код после выполнения операции
02A1	F5	F5
A	21	21
FL	02	02
SP	FF09	FF07
FF07	XX	02
FF08	XX	21
PC	02A1	02A2

080B) PUSH PSW

Адрес	Код до выполнения операции	Код после выполнения операции
080B	F5	F5
A	A6	A6
FL	82	82
SP	0BAC	0BAA
0BAA	03	82
0BAB	01	A6
PC	080B	080C

2.7. Выбор из стека пары регистров

POP ri	**<B1>**	11	**ri**	001
		7		0

ri – код старшего регистра пары (B, D, H)

ri+1 ← [(SP)]; ri ← [(SP)+1]; SP ← (SP)+2

Эта команда по действию обратна команде 2.5. При ее выполнении содержимое ячейки памяти, адрес которой записан в указателе стека (SP), перепишется в младший регистр пары РОН (C, E, L). После этого содержимое указателя стека увеличивается на 1 и из ячейки памяти с полученным адресом переписывается содержимое в старший регистр пары РОН (B, D, H). Затем содержимое указателя стека вновь увеличивается на 1 и остается таким до следующего обращения к нему. Данная команда выполняется за три цикла (10 тактов). При ее выполнении содержимое регистра признаков не меняется.

Примеры

01C9) POP D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
01C9	D1	D1
SP	0BAE	0BB0
0BAE	6E	6E
0BAF	12	12
D	33	12
E	6D	6E
PC	01C9	01CA

080C) POP B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
080C	C1	C1
SP	0BAA	0BAC
0BAA	82	82
0BAB	A6	A6
B	11	A6
C	2A	82
PC	080C	080D
FL	02	02

2.8. Выбор (A) и (F) из стека

POP PSW **<B1>** **F1**

7 0

F← [(SP)]; A ← [(SP)+1]; SP ← (SP)+2

Действие этой команды противоположно действию команды 2.6. Эта команда аналогична команде 2.7 и отличается лишь регистрами. Команда выполняется за три цикла (10 тактов). Содержимое регистра признаков изменяется в соответствии с кодом записанного в стеке слова.

Пример

080D) POP PSW

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
080D	F1	F1
SP	0BAC	0BAE
0BAC	02	02
0BAD	18	18
A	11	18
FL	82	02
PC	080D	080E

2.9. Обмен данными

а) Обмен между DE и HL

XCHG **(H) ↔(D) ; (L) ↔ (E)** **<B1>** **EB**

7 0

б) Обмен вершины стека с HL

XTHL **(L) ↔[(SP)];(H) ↔ [(SP) +1]** **<B1>** **E3**

7 0

По этим командам происходит обмен данными между регистрами D и H, E и L (а) или между регистрами и ячейками памяти (б), адреса которых выбираются из регистра SP (указа-

теля стека). Команды однобайтовые и выполняются за один или пять циклов (4 или 13 тактов) соответственно для команд а) и б). После выполнения команды содержимое регистра признаков не изменяется.

Примеры

00D1) XTHL

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
00D1	E3	E3
L	2A	5A
H	48	0C
SP	12B2	12B2
12B2	5A	2A
12B3	0C	48
PC	00D1	00D2

После выполнения операции содержимое указателя стека остается таким же, как и до выполнения операции.

080E) XCHG

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
080E	EB	EB
D	05	77
E	FA	2B
H	77	05
L	2B	FA
FL	02	02
PC	080E	080F

2.10. Пересылка HL

а) Пересылка в указатель стека

SPHL	(SP)←(H) (L)	<B1>	F9
			7 0

б) Пересылка в счетчик команд

PCHL	(PC)←(H) (L)	<B1>	E9
			7 0

При выполнении этих команд содержимое регистровой пары HL передается в регистр **SP** – указатель стека (а) или в регистр PC – счетчик команд (б). При этом старший байт принимается из регистра H, а младший – из регистра L. Содержимое регистров H и L при выполнении команды не изменяется. Не изменяется также содержимое регистра признаков F. Команда выполняется за один цикл (5 тактов).

Примеры

080F) SPHL

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
080F	F9	F9
SP	0BB0	0BAC
H	0B	0B
L	AC	AC
FL	02	02
PC	080F	0810

0810) PCHL

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0810	E9	E9
PC	0810	09AE
H	09	09
L	AE	AE
FL	02	02

Контрольные вопросы и задания

1. Назначение команд пересылки.
2. Размер команд пересылки.
3. Какие из следующих команд не относятся к группе команд пересылки:
   - PUSH D;
   - MVI M, A6;
   - LXI H, 012E;
   - CALL 023D;
   - LDAX D;
   - CMA?
4. Как влияют команды пересылки на состояние регистра признаков?
5. Как выполняется команда MOV D, M?
6. Какая или какие из приведенных ниже мнемонических значений ошибочны:
   - MOV D, B4;
   - LHLD 01D6;

– PUSH PSW;

– PCHL;

– POP C;

– LDA B4;

– STAX D?

7. Какая из команд MOV C, M и MOV D, E требует для своего выполнения больше времени и почему?
8. В каком случае после выполнения команды MOV H, L содержимое регистров H и L не изменится?
9. Как выполняется команда SPHL?
10. Особенности выполнения команды POP PSW.

Глава 3. Приращение / отрицательное приращение

3.1. Приращение / отрицательное приращение регистра

а)

INR ri	ri ← (ri) + 1	<B1>	00	ri	100

б)

DCR ri	ri ← (ri) – 1	<B1>	00	ri	101
			7		0

В качестве регистров могут быть использованы A, B, C, D, E, H, L, M. При выполнении этой команды содержимое выбранного регистра увеличивается (а) или уменьшается (б) на единицу. Команда изменяет содержимое всех триггеров при-

знаков, кроме признака переноса, который остается без изменения. Команда выполняется за один цикл (5 тактов). Если в качестве регистра выбрана память M, то число циклов увеличивается до трех (10 тактов).

В команде DCR r_i триггер *Тн* устанавливается так же, как и в команде вычитания.

Примеры

015A) INR L

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
015A	2C	2C
L	DB	DC
FL	02	82
PC	015A	015B

0800) DCR M

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	35	35
H	02	02
L	35	35
0235	EF	EE
FL	02	96
PC	0800	0801

02C9) DCR E

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
02C9	1D	1D
E	A0	9F
FL	02	86
PC	02C9	02CA

012C) INR B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
012C	04	04
B	FF	00
FL	02	56
PC	012C	012D

3.2. Приращение пары регистров

INX ri	**ri, ri+1← (ri), (ri+1) + 1**	**<B1>**	00	**ri**	011

при *ri* =110 – приращение SP

7 0

Пример

2C05) INX B

Адрес	Код до выполнения операции	Код после выполнения операции
2C05	03	03
B	37	38
C	FF	00
FL	02	02
PC	2C05	2C06

3.3. Отрицательное приращение пары регистров

DCX ri	ri, ri+1← (ri), (ri+1) – 1	B1

00	ri+1	011

при *ri* =111 – отрицательное приращение SP 7 0

Действие команд 3.2 и 3.3 аналогично действию команды 3.1, однако рассматривается двухбайтовое слово (16 разрядов). Регистровыми парами могут быть HL, BC, DE. В коде команды 3.2 содержится код старшего регистра пары (B, D, H), а в коде команды 3.3 – код младшего регистра пары (C, E, L). При выполнении этих двух команд содержимое регистра признаков не изменяется. На выполнение команды затрачивается один машинный цикл (5 тактов для команды 3.2 и 6 тактов для команды 3.3) Если код регистра равен 110 или 111, происходит увеличение или уменьшение содержимого регистра SP – указателя стека.

Примеры

0801) DCX D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0801	1B	1B
D	FE	FE
E	00	FF
FL	96	96
PC	0801	0802

0A33) DCX D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0A33	1B	1B
D	07	07
E	2E	2C
FL	13	13
PC	0A33	0A34

Контрольные вопросы и задания

1. Какие команды относятся к группе приращение / отрицательное приращение?
2. Какие признаки (флаги) изменяются при выполнении команды DCR A и как?
3. Чем отличаются по своему выполнению команды INR H и INX H?
4. Как изменится работа команды DCX, если (D) = 0A, (E) = 00?
5. Найдите неверно записанную команду или команды:
   - DCR M;
   - INR L;
   - DCX E;
   - INX D;
   - INX M.

Глава 4. Арифметические и логические операции

При выполнении арифметических и логических операций, когда операнды однобайтовые, один из операндов всегда находится в аккумуляторе, а второй находится в одном из регистров A, B, C, D, E, H, L или M, т. е. в памяти. При непосредственной адресации второй операнд приводится во втором байте команды. Результат выполнения операции записывается в аккумулятор. При этом прежнее содержимое аккумулятора теряется.

Микропроцессор содержит ограниченное число арифметических и логических команд: сложение, вычитание, логическое сложение и логическое умножение, исключающее ИЛИ сравнение и дополнение. Более сложные операции над числовыми данными (умножение, деление и другие) реализуются программно.

4.1. Арифметические операции над (A) и (r)

а) Сложение

ADD ri	**A←(A) + (ri)**	**<B1>**	10000	**ri**

б) Сложение с переносом

ADC ri	**A←(A) + (ri) + (Tc)**	**<B1>**	10001	**ri**

в) Вычитание

SUB ri	A←(A) – (ri)	<B1>	10010	ri

г) Вычитание с переносом

SBB ri	A←(A) – (ri) – (Tc)	<B1>	10011	ri
			7	0

В рассмотренных командах производятся арифметические действия над содержимым двух регистров, один из которых (или оба) – аккумулятор. В результате выполнения операции прежнее содержимое аккумулятора замещается суммой или разностью, старое содержимое аккумулятора теряется.

При выполнении команд сложения все триггеры признаков устанавливаются всегда в соответствии с результатом.

При выполнении команд вычитания 3 флага из 5 устанавливаются всегда одинаковым образом, независимо от того сводится вычитание к сложению или нет. Это *Ts, Tz, Tp;* флаги *Tc* и *Tн* ведут себя иначе. Изменение *Tн* определяется при преобразовании вычитания в сложение (путем смены знака вычитаемого и преобразования значения в дополнительный код). Флаг *Tc* = 1, если код уменьшаемого меньше кода вычитаемого, т. е. имеет место заем.

Применение операций с переносом позволяет обрабатывать не только байтовые числа, но и многобайтовые.

Примеры

015B) ADD M

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
015B	86	86
H	3E	3E
L	23	23
3E23	6C	6C
A	A4	10
FL	02	13
PC	015B	015C

1A20) ADC E

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
1A20	8B	8B
A	16	6A=16+54+0
E	54	54
FL	02; (Tc)=0	06
PC	1A20	1A21

00B3) ADC C

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
00B3	89	89
C	DB	DB
A	AF	8B=DB+AF+1
FL	13; Tc=1	97
PC	00B3	00B4

Рассмотренные команды являются однобайтовыми, выполняются за один цикл (4 такта); если операнд хранится в памяти (M), то операция выполняется за два цикла (7 тактов). При выполнении команд меняется содержимое всех триггеров регистра признаков.

1A30) SUB D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
1A30	92	92
D	45	45
A	FC	B7
FL	02	96
PC	1A30	1A31

1A40) SBB B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
1A40	98	98
B	6A	6A
A	5B	F0 = 5B-6A-1
FL	03; Tc=1	97
PC	1A40	1A41

Рассмотрим несколько особенных примеров.

082A) SBB H

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
082A	9C	9C
A	60	60
H	FF	FF
FL	83, Tc=1	17
PC	082A	082B

Особенностью данной команды является то, что триггеры признаков *Tc* и *Tн* устанавливаются так, как это происходит при выполнении этой команды в два этапа:

1) (A) – Tc = 60 – 1= 5F;

2) ((A) – Tc) – H = 5F – FF = 60;

и признаки переноса и полупереноса (*Tc* и *Tн*) устанавливаются в соответствии со вторым этапом, т.е. *Tc* = 1, *Tн* = 1.

0903) ADD L

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0903	85	85
A	7F	81
L	02	02
FL	13	96
PC	0903	0904

0A09) SUB D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0A09	92	92
A	83	79
D	0A	0A
FL	17	02
PC	0A09	0A0A

Особенностью двух последних команд является то, что их результаты имеют смысл только в тех случаях, когда обрабатываемые коды являются кодами чисел без знака.

4.2. Арифметические операции с непосредственной адресацией

а) Сложение со вторым байтом

ADI Данные	**A←(A) + <B2>**	**<B1>**	**C6**
		<B2>	**Данные**

б) Сложение со вторым байтом и переносом

ACI Данные	**A←(A) + <B2>+(Tc)**	**<B1>**	**CE**
		<B2>	**Данные**

в) Вычитание второго байта

SUI Данные	**A←(A) – <B2>**	**< B1>**	**D6**
		<B2>	**Данные**

г) Вычитание второго байта с переносом

SBI Данные	**A←(A) – <B2>– (Tc)**	**<B1>**	**DE**
		<B2>	**Данные**

7 0

При выполнении этой группы команд второй операнд приводится непосредственно в команде во втором ее байте. Команды являются двухбайтовыми, выполняются за два цикла (7 тактов). Триггеры признаков устанавливаются в соответствии с результатом выполненной операции. В результате выполнения операции изменяется содержимое аккумулятора, старое содержимое аккумулятора теряется.

Примеры

12A6) ADI 47

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
12A6	C6	C6
12A7	47	47
A	24	6B=24+47
FL	97	02
PC	12A6	12A8

0802) ACI A7

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0802	CE	CE
0803	A7	A7
A	53	FB
FL	03	82
PC	0802	0804

0804) SUI FD

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0804	D6	D6
0805	FD	FD
A	FB	FE
FL	03	83
PC	0804	0806

0806) SBI 0A

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0806	DE	DE
0807	0A	0A
A	F8	ED
FL	03	96
PC	0806	0808

4.3. Сложение содержимого пар регистров

DAD ri	**HL ←(HL)+ (ri)(ri+1) <B1>**	00	ri+1	010

При ri+1=111 – HL ←(HL) + (SP)

7 0

При выполнении этой команды содержимое пары регистров HL складывается с содержимым пар BC, DE или с содержимым указателя стека (SP). В коде команд приводится код младшего регистра пары (C или E). В результате выполнения операции устанавливается в соответствующее состояние только триггер переноса, который принимает значение в соответствии

с переносом из старшего бита старшего регистра. Команда выполняется за три цикла (10 тактов).

Пример

1FC1) DAD D

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
1FC1	19	19
H	1A	70
L	10	35
D	56	56
C	25	25
FL	97	96
PC	1FC1	1FC2

4.4. Логические операции над (A) и (r)

а) Логическая операция «И»

ANA ri	**A←(A) ∧ (ri)**	**<B1>**	10100	**ri**
			7	0

Эта операция, как и все остальные логические операции, является побитовой. Можно считать, что для обработки битов каждого разряда используется одна двухвходовая логическая схема, на один вход которой подается значение бита аккумулятора, а на другой – значение соответствующего бита слова, расположенного в памяти (M) или одном из регистров (A, B, C, D, E, H, L). В микропроцессоре предусмотрено восемь таких двухвходовых схем – по одной для каждого разряда микропроцессо-

ра. После выполнения операции результат запишется в аккумулятор, а старое содержимое аккумулятора потеряется. В результате выполнения операции логического умножения все триггеры регистра признаков, кроме триггера переноса (*Тс*), устанавятся в соответствии с полученным результатом; триггер переноса установится в состояние 0. Триггер полупереноса (*Тн*) принимает значение разряда А3 результата. На выполнение операции затрачивается один цикл (4 такта). Если *ri* = *M*, то число циклов равно двум (7 тактов).

Пример

0800) ANA C

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	A1	A1
A	11010111=D7	10000000=80
C	10100000=A0	A0
FL	02	82
PC	0800	0801

б) Операция «исключающее ИЛИ»

XRA ri	**A←(A) ∀ (ri)**	**<B1>**	10101	**ri**
			7	0

Эта операция, как и предыдущая, является побитовой; для ее исполнения используются те же регистры. После выполнения команды результат записывается в аккумулятор на место первого операнда. Триггеры признаков, кроме триггеров переноса и полупереноса, устанавливаются в соответствии с результатом.

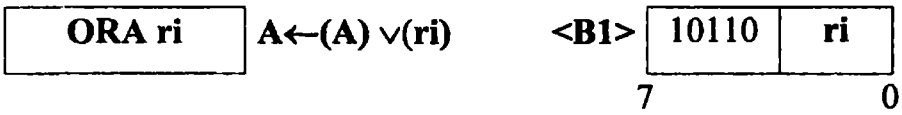
Триггеры *Тс* и *Тн* принимают значение 0. Операция выполняется за 1/2 цикла (4/7) тактов.

Пример

0800) XRA C

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	A9	A9
A	11010111=D7	00000000=00
C	10100000=A0	A0
FL	02	42
PC	0800	0801

в) Логическая операция «ИЛИ»



Все сказанное для команды б) справедливо и для этой команды.

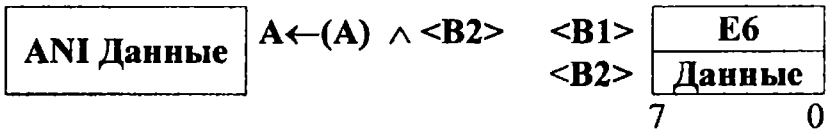
Пример

0800) ORA C

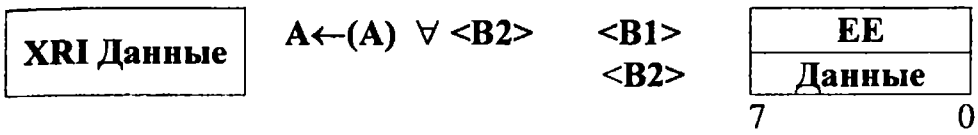
Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	B1	B1
A	11010111=D7	11110111=F7
C	10100000=A0	A0
FL	02	82
PC	0800	0801

4.5. Логические операции с непосредственной адресацией

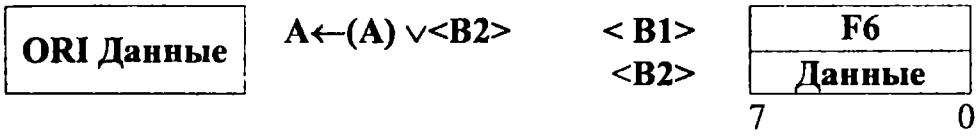
а) Логическая операция «И» со вторым байтом



б) Логическая операция «исключающее ИЛИ» со вторым байтом



в) Логическая операция «ИЛИ» со вторым байтом



Все сказанное для логических операций над *A* и *ri* (п. 4) справедливо и для команд данной группы. Отличие состоит лишь в том, что второй операнд приводится непосредственно в команде во втором ее байте. Каждая операция выполняется за два машинных цикла (7 тактов).

Примеры

0900) XRI 2B

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0900	EE	EE
0901	00101011=2B	2B
A	10101010=AA	10000001=81
FL	02	86
PC	0900	0902

0800) ANI 98

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	E6	E6
0801	98	98
A	AE	88
FL	03	96
PC	0800	0802

0802) ORI FE

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0802	F6	F6
0803	FE	FE
A	A3	5D
FL	03	02
PC	0802	0804

4.6. Операции сравнения

а) Сравнение (A) с (ri)

CMP ri	**(A) – (ri)**	**<B1>**	10111	**ri**
			7	0

Пример

0100) CMP D

Адрес	Код до выполнения операции	Код после выполнения операции
0100	BA	BA
A	B6	B6
D	CB	CB
FL	02	87
PC	0100	0101

б) Сравнение (A) со вторым байтом

CPI Данные	**(A) – <B2>**	**<B1>**	**FE**
		<B2>	**Данные**
			7 0

Пример

0125) CPI 25

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0125	FE	FE
0126	25	25
A	96	96
FL	02	17
PC	0125	0127

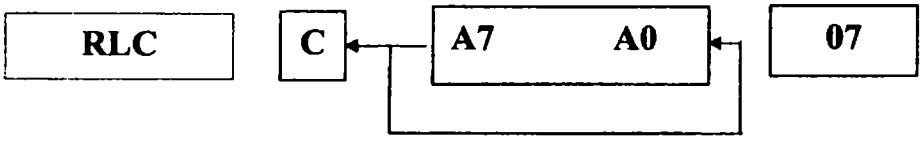
Выполнение команд этой группы аналогично выполнению команды «вычитание», но ее результат не загружается в аккумулятор. Хотя по окончании операции исходные данные в аккумуляторе не изменяются, разрядам Z, S и C регистра признаков присваиваются значения в соответствии с полученным результатом.

В качестве регистров *ri* в команде а) могут быть выбраны A, B, C, D, E, H, L, M. С помощью этой команды можно производить сравнение двух слов на «больше», «меньше», «равно».

На выполнение команды а) затрачивается один машинный цикл (4 такта) при работе с регистрами или два цикла (7 тактов), если данные находятся в памяти (M) по адресу HL. Команда б) выполняется за два цикла (7 тактов).

4.7. Операции циклического сдвига (A)

а) Сдвиг аккумулятора влево без переноса

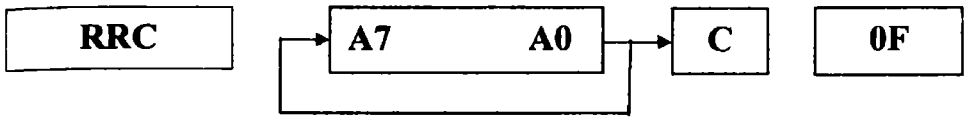


Пример

0800) RLC

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	07	07
A	96	2D
FL	02	03
PC	0800	0801

б) Сдвиг аккумулятора вправо без переноса

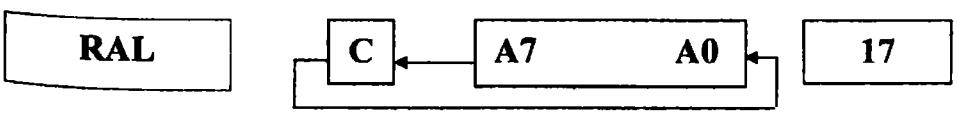


Пример

0801) RRC

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0801	0F	0F
A	96	4B
FL	02	02
PC	0801	0802

в) Сдвиг аккумулятора влево с переносом

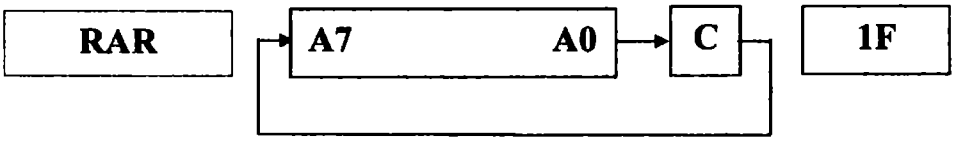


Пример

0802) RAL

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0802	17	17
A	96	2D
FL	02	03
PC	0802	0803

г) Сдвиг аккумулятора вправо с переносом



Пример

0803) RAR

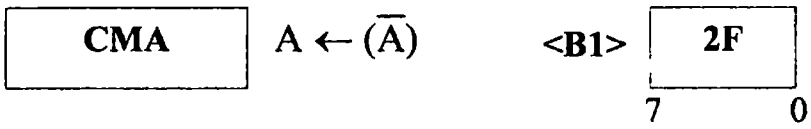
Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0803	1F	1F
A	96	4B
FL	02	02
PC	0803	0804

Командой этой группы все данные, находящиеся в аккумуляторе и триггере переноса, сдвигаются на одну позицию. При этом в командах а) и б) значение выдвигаемого разряда A7 или A0 передается в разряд A0 или A7 соответственно и запоминается в триггере переноса C. В командах в) и г) триггер пе-

реноса используется как дополнительный девятый разряд регистра.

Каждая команда выполняется в течение одного машинного цикла (4 такта). В результате выполнения команды изменяется значение только триггера переноса C, а другие признаки остаются прежними.

4.8. Дополнение аккумулятора



С помощью этой команды производится инвертирование всех разрядов аккумулятора. Команда выполняется в течение одного цикла (4 такта). Значение триггеров признаков при этом не изменяется.

Пример

0823) CMA

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0823	2F	2F
A	F2	0D
FL	02	02
PC	0823	0824

Контрольные вопросы и задания

1. Перечислите логические команды МП КР580ВМ80А.
2. Перечислите арифметические команды МП КР580ВМ80А.
3. Как изменяются признаки при выполнении команды ADD?
4. Как изменяются признаки при выполнении команды SUB?
5. Как изменяются признаки при выполнении команды SBB?
6. Как изменяются признаки при выполнении команды ANA?
7. Как изменяются признаки при выполнении команд ORA и XRA?
8. В каких случаях может возникнуть переполнение разрядной сетки в командах сложения и вычитания?
9. Назовите признаки переполнения разрядной сетки.
10. Какая или какие из перечисленных ниже команд не относятся к группе арифметических и логических команд:
    - ANI 4D;
    - SBB M;
    - MVI A9;
    - XRA C;
    - ADC H;
    - CPI D4;
    - CMP E;
    - LXI B;
    - SUI 91?
11. Чем отличается выполнение команд сравнения и вычитания?

12. Куда всегда помещается результат выполнения логических команд, команд сложения и вычитания?
13. Определите результаты работы следующей команды: ADD E при (A) = 7E, (E) = F2.
14. Определите результаты работы следующей команды: SUB L при (A) = 83, (L) = 5A.
15. Определите результаты работы следующих команд: ANA H, ORA H, XRA H при одних и тех же операндах.

Глава 5. Команды перехода и работы с подпрограммами

Команды этой группы позволяют изменять последовательность выполнения команд программы. Существуют два способа изменения этой последовательности. Первый из них называется безусловным. Согласно этому способу последовательность выполнения программы подвергается изменению всякий раз, когда реализуется определенная команда. В соответствие со вторым способом последовательность выполнения программы определяется некоторыми условиями, т.е. изменяется только в том случае, когда значение указанного условия совпадает с заданным.

Команды перехода и вызова подпрограмм являются одной из составных частей процесса принятия решений. В результате выполнения команд арифметической и логической обработки данных вырабатываются значения признаков в регистре состоя-

ния. Команды перехода и вызова подпрограмм проверяют значения разрядов регистра признаков (состояния) и определяют последующий ход выполнения программ в зависимости от результата проверки.

Команды перехода, называемые также командами ветвления, позволяют организовать в программах циклы и разветвления.

Команды вызова подпрограмм дают возможность сократить объем разрабатываемых программ за счет повторного использования подпрограмм. Наличие в МП стека с указателем стека делает возможным возврат в главную программу после выполнения подпрограммы.

Ни одна из команд перехода и вызова подпрограммы не изменяет состояние регистра признаков.

5.1. Команды переходов

Эти команды занимают в оперативной памяти 3 байта и выполняются за три машинных цикла (10 тактов). Во всех этих командах применяется прямая адресация. Команды делятся на 2 группы:

1) 1 команда безусловного перехода;

2) 8 команд с условным переходом.

При выполнении команды безусловного перехода изменяется содержимое счетчика команд PC. Содержимое второго и третьего байтов команды перехода пересылается автоматически в счетчик команд во время фазы выполнения. Тогда при оче-

редном цикле выборки МП извлекает команду из области памяти, на которую указывает 2-й и 3-й байты команды перехода. Таким образом, происходит переход в другую точку программы. Теперь выполняются одна за другой команды новой последовательности. Это продолжается до тех пор, пока не будет опять выполнена команда перехода.

а) Переход безусловный

JMP Адрес	**(PC) ← <B3><B2>**	**<B1>**	**C3**
		<B2>	Мл. байт адреса
		<B3>	Ст. байт адреса
			7 0

После этой команды выполнение программы продолжается с новой исходной точки, т.е. с адреса, содержащегося в байтах B2 и B3 команды перехода.

б) Переход условный

(PC) ← <B3><B2>, если проверяемое условие выполняется; в противном случае **PC ← (PC)+3**, т.е. выполняется следующая команда после команды условного перехода.

<B1>	11	CCC	010	
<B2>	Мл. байт адреса			
<B3>	Ст. байт адреса			
	7			0

Значение разрядов CCC команда выбирает в зависимости от проверяемого условия. В МП К580 имеется возможность делать переход в зависимости от результата проверки следующих условий:

Команда перехода	Проверяемое условие	Значение CCC
JNZ Адрес	(Tz) = 0	000
JZ Адрес	(Tz) = 1	001
JNC Адрес	(Tc) = 0	010
JC Адрес	(Tc) = 1	011
JPO Адрес	(Tp) = 0	100
JPE Адрес	(Tp) = 1	101
JP Адрес	(Ts) = 0	110
JM Адрес	(Ts) = 1	111

Примеры

0806) **JMP beep**

beep: 0A76

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0806	C3	C3
0807	76	76
0808	0A	0A
PC	0806	0A76
FL	02	02

0A58) **JMP beep**

beep: 0B43

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0A58	C3	C3
0A59	43	43
0A5A	0B	0B
PC	0A58	0B43
FL	02	02

0921) **JNZ L0**

L0: 0A42

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0921	C2	C2
0922	42	42
0923	0A	0A
PC	0921	0A42
FL	02 (T_Z=0)	02

0800) **JNZ L0**

L0: 09FE

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	C2	C2
0801	FE	FE
0802	09	09
PC	0800	0803
FL	42 (T_Z=1)	42

0921) **JC L0**

L0: 0A42

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0921	DA	DA
0922	42	42
0923	0A	0A
PC	0921	0A42
FL	03 (T_C=1)	03

0921) **JC L0**

L0: 0A42

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0921	DA	DA
0922	42	42
0923	0A	0A
PC	0921	0924
FL	02 (T_C=0)	02

5.2. Команды вызова подпрограмм и возврата из подпрограмм

С помощью команд переходов последовательность выполнения команд программы может быть изменена путем перехода к новой последовательности. Однако команда перехода сама по себе не позволяет вернуться в то место главной программы, откуда был осуществлен переход. Это можно легко сделать с помощью команды вызова подпрограммы. Подпрограмма пред-

ставляет собой фрагмент программы, обращение к которой может иметь место в любой точке главной программы. Когда происходит вызов подпрограммы, то в начале своего выполнения она реализует действия по запоминанию текущего содержимого счетчика команд (точка возврата). Когда выполнение подпрограммы заканчивается, то с помощью команды возврата микропроцессору указывается, что исходное содержимое счетчика команд должно быть извлечено из памяти. Этой информации микропроцессору достаточно, чтобы осуществить возврат в прерванную последовательность команд главной программы. Для запоминания точки возврата используется стек, куда записывается адрес команды, следующей за командой вызова подпрограммы.

Команды данной группы не изменяют содержимого регистра признаков.

а) Вызов подпрограммы безусловный

CALL Адрес	**[SP-1] [SP-2] ← (PC)**	**< B1>**	**CD**
	SP ← (SP) -2	**<B2>**	Мл. байт адреса
	PC ← <B3><B2>	**<B3>**	Ст. байт адреса
			7 0

При выполнении этой команды никаких проверок не производится, т.е. вызов подпрограммы производится в любом случае. Команда занимает три байта и выполняется за пять машинных циклов (17 тактов).

б) Условный вызов подпрограммы

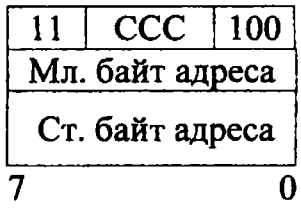
Переход к подпрограмме происходит только в том случае, если выполняется проверяемое условие. При этом

[SP-1] [SP-2] ← (PC); SP ← (SP) –2; PC ← <B3><B2>.

В этом случае команда выполняется за пять циклов (17 тактов).

Если проверяемое условие не выполняется, то переход к подпрограмме не происходит и выполняется следующая команда программы с адресом PC = (PC) +3. В этом случае команда выполняется за три цикла (11 тактов).

Формат команды условного вызова



Значение разрядов CCC выбирается в зависимости от проверяемого условия:

Команда вызова	Проверяемое условие	Значение CCC
CNZ Адрес	(Tz) = 0	000
CZ Адрес	(Tz) = 1	001
CNC Адрес	(Tc) = 0	010
CC Адрес	(Tc) = 1	011
CPO Адрес	(Tp) = 0	100
CPE Адрес	(Tp) = 1	101
CP Адрес	(Ts) = 0	110
CM Адрес	(Ts) = 1	111

в) Возврат из подпрограммы безусловный

RET	PC ← [SP] [SP+1]; SP← (SP) + 2	B1	C9

7 0

Команда выполняется за три машинных цикла (10 тактов).

г) Возврат из подпрограммы условный

Команда условного возврата из подпрограммы <B1>

11	CCC	000

7 0

Если проверяемое условие выполняется, то PC ← [SP] [SP-1]; SP← (SP) + 2 и происходит выход из подпрограммы в точку, адрес которой записан в стеке. В этом случае команда выполняется за три машинных цикла (11 тактов). Если это условие не выполняется, то возврата не происходит и выполняется следующая команда подпрограммы, т.е. PC = (PC) + 1.

В этом случае команда выполняется в течение одного машинного цикла (5 тактов).

Значение разрядов CCC команды определяется проверяемым условием:

Команда возврата	Проверяемое условие	Значение CCC
RNZ	(Tz) = 0	000
RZ	(Tz) = 1	001
RNC	(Tc) = 0	010
RC	(Tc) = 1	011
RPO	(Tp) = 0	100
RPE	(Tp) = 1	101
RP	(Ts) = 0	110
RM	(Ts) = 1	111

При выполнении команд возврата (в и г) МП извлекает из стека значение программного счетчика, при котором произошел переход из программы предыдущего уровня, и загружает это значение в счетчик команд. Команде «возврат из подпрограммы» ничего не сообщается относительно того, какого уровня вложения выполняется подпрограмма и сколько раз происходил вызов этой подпрограммы. Выполняя эту команду, МП просто возвращается к тому значению программного счетчика, которое было последним загружено в стек.

Примеры

0AF3) **CALL BEEP**

BEEP: 0BC0)

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0AF3	CD	CD
0AF4	C0	C0
0AF5	0B	0B
PC	0AF3	0BC0
SP	0BB0	0BAE
0BAE	00	F6
0BAF	00	0A
FL	02	02

0BD4) **RET**

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0BD4	C9	C9
SP	0BAE	0BB0
0BAE	F6	F6
0BAF	0A	0A
PC	0BD4	0AF6
FL	02	02

0A12) **CPO L1**
L1: 0AFE)

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0A12	E4	E4
0A13	FE	FE
0A14	0A	0A
SP	0BAE	0BAC
0BAD	00	15
0BAC	00	0A
PC	0A12	0AFE
FL	02 (T_P=0)	02

0800) **CP L1**

L1: 0C54)

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	F4	F4
0801	54	54
0802	0C	0C
SP	0BAC	0BAC
PC	0800	0803
FL	12 (T_S=1)	12

3456) **CC M0**

M0: 0807)

Адрес/регистр	Код до выполнения операции	Код после выполнения операции
3456	DC	DC
3457	07	07
3458	08	08
SP	0BB0	OBAE
0BAF	00	59
0BAE	00	34
PC	3456	0807
FL	03 (T_C=1)	03

Контрольные вопросы и задания

1. Объясните назначение команд переходов.
2. В чем состоят различия в работе команд переходов и вызова подпрограмм?
3. Какую длину имеют команды вызова и возврата из подпрограмм и почему?
4. На какие группы делятся команды переходов и работы с подпрограммами?
5. Какой или какие флаги не используются в командах переходов и работы с подпрограммами?
6. Как изменяются стек и SP при выполнении команды CALL?
7. Как изменяются стек и SP при выполнении команды RET?

8. Как изменяется PC, если проверяемое условие в команде условного вызова подпрограммы не выполняется?
9. Как изменяется PC, если проверяемое условие в команде условного возврата из подпрограммы не выполняется?
10. Найдите ошибку в данной программе:

```
0800) CALL L0
  RST1
  L0:    0900) CMA
  PUSH PSW
  RET
```

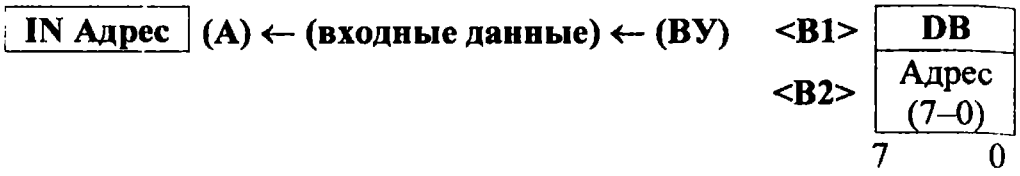
Глава 6. Команды ввода – вывода

Эти команды обеспечивают передачу данных между аккумулятором МП и внешним устройством, 8-ми разрядный адрес которого приводится во втором байте команды. Таким образом, с помощью команд ввода-вывода можно передавать данные к $256 = 2^8$ портам вывода и принимать данные от 256 портов ввода.

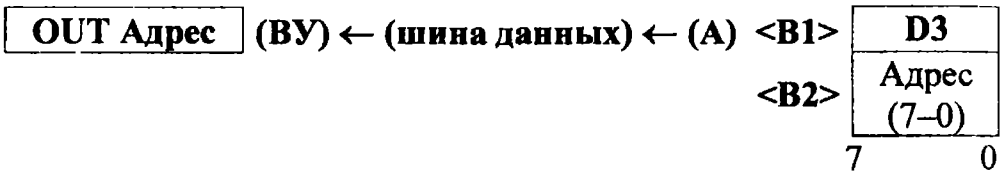
После того как команда будет выбрана полностью, адрес внешнего устройства передается из МП на ША по восьми младшим линиям и таким образом обеспечивает передачу данных между МП и адресуемым внешним устройством. Следует иметь в виду, что 8-разрядный адрес внешнего устройства дублируется и передается также по восьми старшим линиям ША.

Команды выполняются за три машинных цикла (10 тактов). При выполнении команд значения триггеров признаков не изменяются.

6.1. Ввод данных из входного порта



6.2. Вывод данных в выходной порт



Контрольные вопросы и задания

1. В чем состоят особенности команд ввода-вывода?
2. Как влияют команды ввода-вывода на флаговый регистр?
3. Какой дополнительный сигнал формируется на шине управления при выполнении команд ввода-вывода?
4. Сколько байт занимают команды ввода-вывода?
5. Выберите одну или несколько неправильно записанных команд ввода-вывода из предложенного списка:

– IN 05;

– LDA 05;

– OUT 02B7;

– MOV M,05;

– IN B, 0A;

– STA 0B;

– OUT BF.

Глава 7. Команды управления

7.1. Рестарт (повторный запуск)

RST	[SP-1] [SP-2] ← (PC)	<B1>	11	AAA	111
	SP ← (SP) -2		7		0
	PC ← 00000000 00AAA000				

Так же, как и команда CALL, эта команда обеспечивает безусловный переход к подпрограмме, начальный адрес которой задается с помощью разрядов 3–5 команды. Команда рестарт может быть подана внешним прерывающим устройством в ответ на сигнал разрешения прерывания, а также может присутствовать в программе. По этой команде управление передается одной из восьми подпрограмм с векторами (указателями областей памяти): 0000, 0008, 0010, 0018, 0020, 0028, 0030, 0038 H.

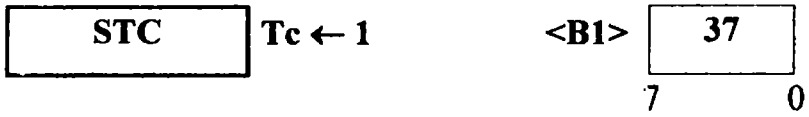
Значения соседних векторов отличаются на 8. Области памяти, указываемые векторами, могут содержать либо короткие 8-байтовые программы обработки прерываний, либо

3-байтовые команды безусловного перехода к некоторой программе обработки прерывания.

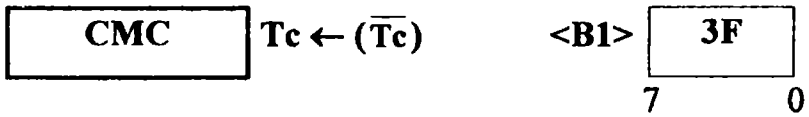
Команда выполняется за три машинных цикла (11тактов). Значения триггеров признаков не изменяются.

7.2. Изменение (Tc)

а) Установка переноса



б) Дополнение переноса



Команды этой группы изменяют значение триггера переноса. При этом остальные триггеры флажков остаются без изменения. Каждая команда выполняется за один машинный цикл (4 такта).

Примеры

0800) STC

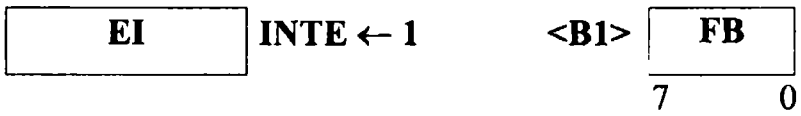
Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	37	37
FL	02(Tc=0)	03 (Tc=1)
PC	0800	0801

0800) CMC

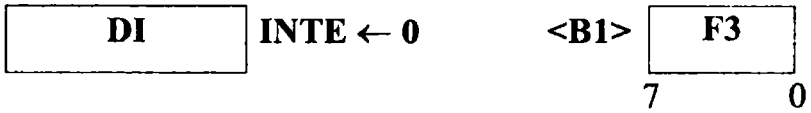
Адрес/регистр	Код до выполнения операции	Код после выполнения операции
0800	3F	3F
FL	13(Tc=1)	12 (Tc=0)
PC	0800	0801

7.3. Управление прерываниями

а) Разрешение прерывания



б) Блокировка прерываний



Команды этой группы позволяют программно защитить от прерываний некоторые участки программы. С помощью этих команд производится установка (а) или сброс (б) триггера прерываний INTE МП. Если прерывания запрещены, то МП не будет реагировать на сигналы запроса прерываний до тех пор, пока программно этот запрет не будет снят с помощью команды EI. Если прерывания разрешены, то с приходом сигнала запроса на прерывание триггер INTE автоматически сбросится при переходе к прерывающей программе и установить его можно лишь командой EI. Это делается обычно на том этапе выполнения подпрограммы обслуживания прерывания, когда уже могут быть разрешены другие прерывания.

Каждая команда выполняется за один машинный цикл (4 такта). Значения триггеров признаков при этом не изменяются.

7.4. Двоично-десятичная коррекция

С помощью этой команды производится коррекция содержимого аккумулятора, которое получено в результате арифметического сложения двоично-десятичных чисел на предыдущем шаге выполнения программы. Коррекция заключается в прибавлении числа 0110 В к каждой двоичной тетраде 8-разрядного числа, записанного в аккумулятор, и производится в том случае, если число в тетраде не меньше 10, или имел место перенос из этой тетрады, или число в старшей тетраде равно 9 и был перенос из младшей тетрады (полуперенос).

Команда выполняется за один цикл (4 такта). Признаки принимают значения в соответствии с полученным после коррекции результатом.

Пример

```
0800) ADD B
      DAA
```

7.5. Пустая операция

NOP	<B1>	00
		7 0

Эта команда ничего не изменяет, кроме того, что содержимое программного счетчика увеличивает на единицу. Команда выполняется за один цикл (4 такта) и используется для организации временных задержек в программе, а также резервирования места, чтобы можно было вносить изменения в отдельные участки программы без изменения адресов остальных команд.

7.6. Останов

HLT	<B1>	76
		7 0

Эта команда вызывает прекращение выполнения программы и переводит МП в состояние останова. При этом ША и ШД переводятся в высокоимпедансное состояние, а на линии ожидания (WAIT) устанавливается высокий уровень. В этом состоянии МП может находиться в течение любого временного интервала. Из состояния останова МП можно вывести двумя способами:

– подачей сигнала сброса на вход RESET, при котором выполнение программы начинается с адреса 0000 H;

– подачей сигнала на вход прерывания INT. При этом триггер разрешения прерывания должен быть установлен; в

противном случае при INTE = 0 единственным способом запуска оказывается сигнал RESET.

Контрольные вопросы и задания

1. Перечислите команды управления, позволяющие изменять состояние отдельных флагов.
2. Какую команду нежелательно помещать в конце программы?
3. Как можно разрешать и запрещать аппаратные прерывания?
4. Как выполняется команда рестарта?
5. Для чего используется команда NOP?
6. Для чего предназначена команда DAA?
7. Какие команды управления из перечисленных ниже не изменяют состояние флагового регистра:
   - STC;
   - NOP;
   - ET;
   - RST1;
   - DAA?
8. Рассмотрите выполнение следующей программы:

   0803) ADD 74

   DAA

   При (A) = 39, (Fl) = 02.

Глава 8. Архитектура МП Intel 8085

8.1. Структура МП Intel 8085

На рис. 8.1 показана архитектура МП Intel 8085. Он имеет 16-разрядный счетчик команд и защелку адреса, которая загружает специализированную адресную (A15–A8) и мультиплексированную шины (AD7–AD0). Параллельные данные входят в МП и покидают его через AD7–AD0. Эта шина передает адрес, когда линия управления ALE получает H-сигнал, и данные, когда L-сигнал.

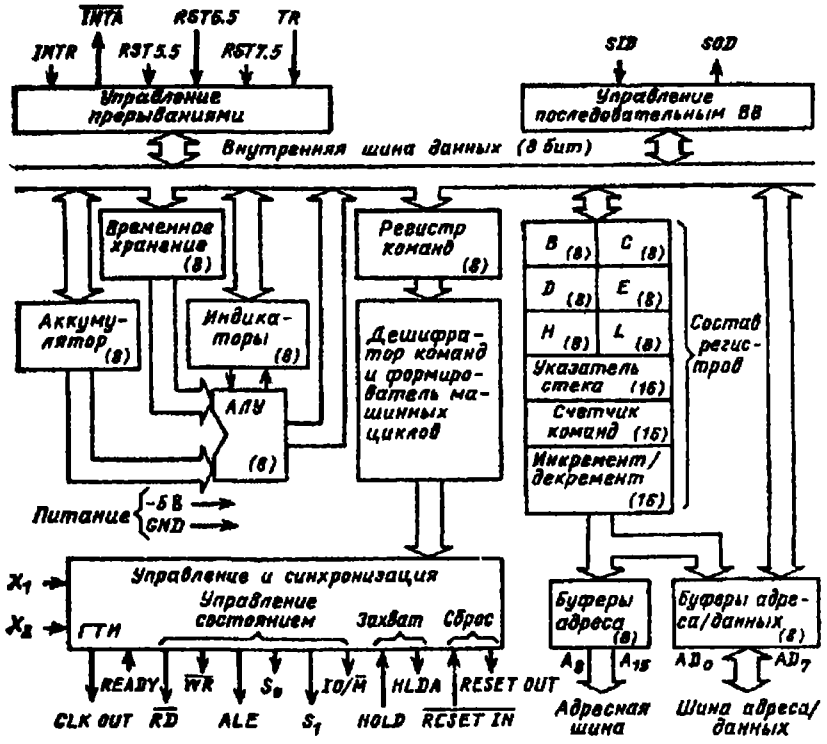


Рис. 8.1. Функциональная схема МП Intel 8085 (архитектура)

По 8-разрядной внутренней шине входящие и выходящие данные вводятся внутрь устройства. Они могут поступать с

внутренней шины данных в 8-разрядный аккумулятор или регистр временного хранения, в индикаторы, регистр команд, устройство управления, в какой-либо из регистров общего назначения (B, C, D, E, H, L), 16-разрядный указатель стека, 16-разрядный счетчик команд или 8-разрядный буфер адреса / данных. Выводы SID и SOD ввода и вывода последовательных данных приведены справа вверху на рис. 8.1, входы прерывания (INTR, RST5.5, RST6.5, RST7.5 и TRAP) — вверху слева вместе с выходом $\overline{\text{INTA}}$ (подтверждение запроса на прерывание). Арифметико-логическое устройство загружается двумя 8-разрядными регистрами (аккумулятором и регистром временного хранения), как и в МП Intel 8080. Регистр состояний содержит пять индикаторов состояния.

Регистр команд связан с дешифратором. Последний определяет текущую команду, требуемую микропрограмму или следующий машинный цикл. Он информирует схему управления и синхронизации о последовательности действий. Эта схема координирует действия МП и периферии.

8.2. Регистры

Как и в случае МП Intel 8080, в состав МП Intel 8085 входят 8- и 16-разрядные регистры. Адресуемых 8-разрядных регистров здесь восемь, шесть из которых (регистры общего назначения) могут быть использованы или как 8-разрядные, или мо-

гут объединяться в три 16-разрядные пары. Кроме того, МП Intel 8085 содержит два 16-разрядных регистра.

1. Аккумулятор (или регистр А) является ядром все операций МП, к которым относятся арифметические, логические операции, загрузки или размещения данных памяти и ВВ. Это 8-разрядный регистр.

2. Регистры общего назначения BC, DE и HL могут быть использованы как шесть 8-разрядных пар или три 16-разрядные пары регистров в зависимости от текущей выполняемой команды. Как и в МП Intel 8080, пара HL (фирмой Intel названа указателем данных) может быть использована для указания адреса. Несколько команд используют пары BC и DE в качестве указателя адреса, но обычно они являются регистрами хранения данных.

3. Счетчик команд PC всегда указывает на ячейку памяти следующей для выполнения команды.

4. Указатель стека SP является специальным регистром – указателем адреса (или данных), который всегда указывает на вершину стека в ОЗУ. Это 16-разрядный регистр.

5. Регистр состояния (или индикаторов) содержит пять одноразрядных индикаторов, в которых содержится информация, относящаяся к состоянию МП. Эти указатели используются условными ветвлениями программы, вызовами подпрограмм и возвратами из подпрограмм.

8.3. Ввод и вывод последовательных данных

Выводы, предназначенные *для ввода и вывода последовательных* данных в МП Intel 8085, способствуют минимизации числа кристаллов в малой системе, составляя интерфейс последовательного порта. По специальной команде RIM данные передаются с вывода последовательного входа SID в бит 7 (b7) аккумулятора (рис. 8.2,а, где в качестве примера H-сигнал передается по линии SID в наиболее значимый бит аккумулятора).

Отдельный последовательный бит может быть выведен через выход SOD, используя специальную команду SIM (см. рис. 8.2,б, где в качестве примера L-сигнал выводится по линии SOD через защелку последовательного выхода). Заметим, что на рис. 8.2 источником данных является наиболее значимый бит 7 (b7) аккумулятора. Бит 6 (b6) аккумулятора должен быть установлен в положение 1, чтобы мог осуществляться последовательный вывод данных.

Последовательный вход SID может быть использован так же, как универсальный вход TEST, тогда как вывод выхода SOD может служить выходом однобитовой команды.

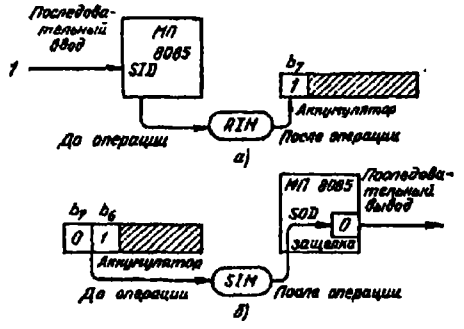


Рис. 8.2. Схемы выполнения команд:
а – последовательного ввода RIM;
б – последовательного вывода SIM

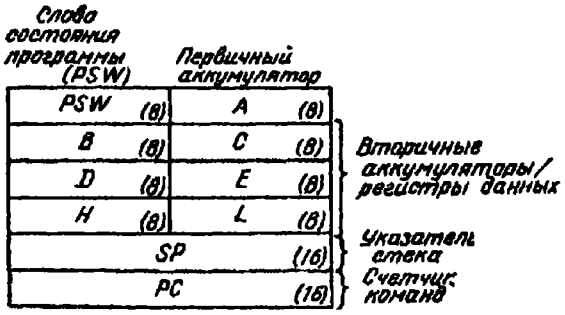


Рис. 8.3. Доступные для программиста регистры МП Intel 8085

Мнемоника RIM означает считывать маску прерывания (Read Interrupt Mask), SIM – установить маску прерывания (Set Interrupt Mask).

На рис. 8.3 представлены программируемые регистры МП Intel 8085. Эти регистры являются для программиста основными, так как они доступны, и этот набор регистров составляет программную модель МП Intel 8085.

Две команды только для INTEL 8085

RIM	**Читать маску прерывания**	**B1**	00100000
			7 0
SIM	**Установить маску прерывания**	**B1**	00110000
			7 0

Для уяснения смысла этих команд необходимо рассмотреть структуру МП INTEL 8085.

Заключение

В учебном пособии рассмотрены вопросы, связанные с архитектурой и программированием на языке ассемблера 8-разрядного микропроцессора КР580ВМ80А (Intel 8080), а также микропроцессора КР1821ВМ85 (Intel 8085).

Архитектура и функционирование МП КР580ВМ80А достаточно подробно излагаются в главе 1. Такой материал впервые встречается в курсе «Микропроцессоры и интерфейсные средства транспортных средств», и без его усвоения невозможно научиться программировать на языке ассемблера.

Знание языка ассемблера конкретного микропроцессора является своего рода фундаментом в базовой подготовке как программистов, так и пользователей микропроцессорной техники, поскольку позволяет разрабатывать наиболее эффективные программы или эффективные ассемблерные фрагменты критических секций программ на языке высокого уровня, а также открывает доступ ко всем ресурсам той конкретной системы, для которой разрабатывается программа.

Выбранный микропроцессор очень удобен в качестве материала для первого знакомства с ассемблером по следующим причинам:

– простота команд и методов адресации;

– наличие серийного выпускаемых учебных стендов УМПК 80, на базе которых разработан соответствующий лабораторный практикум;

– использование изучаемых команд и методов адресации в ассемблерах современных 8-разрядных микроконтроллеров.

Учебное пособие может быть полезно для студентов специальности 220301 (210200) «Автоматизация технологических процессов и производств в машиностроении» специализации 46 «Автоматические и электронные системы транспортных средств», так как направление их подготовки связано с разработкой, наладкой и эксплуатацией современных микропроцессорных и микроконтроллерных систем, архитектура и программирование которых имеют много общего с микроконтроллером Intel 8080 (КР580ВМ80).

Список литературы

1. Схемотехника электронных систем. Микропроцессоры и микроконтроллеры / В.И. Бойко, А.Н. Гуржий, В.Я. Жуйков, А.А. Зори, В.М. Спивак, Т.А. Терещенко, Ю.С. Петергеря. – СПб.: БХВ-Петербург, 2004. – 464 с.
2. Простейшая микро-ЭВМ: Проектирование. Наладка. Использование / Л.Н. Бурев, А.Л. Дудко, В.Н. Захаров. – М.: Энергоатомиздат, 1989. – 216 с.
3. Программирование на языке ассемблера для микропроцессоров 8080 и 8085 / Л. Левенталь, У. Сэйвилл. – М.: Радио и связь, 1987. – 448 с.

Приложение

Коды команд

CE	2	ACI	1100	1110	1B	1	DCX D	50	1	MOV D, B	F1	1	POP PSW
8F	1	ADC A	1000	1111	2B	1	DCX H	51	1	MOV D, C	C5	1	PUSH B
88	1	ADC B	1000	1000	3B	1	DCX SP	52	1	MOV D, D	D5	1	PUSH D
89	1	ADC C	1000	1001	F3	1	DI	53	1	MOV D, E	E5	1	PUSH H
8A	1	ADC D	1000	1010	FB	1	EI	54	1	MOV D, H	F5	1	PUSH PSW
8B	1	ADC E	1000	1011	76	1	HLT	55	1	MOV D, L	17	1	RAL
8C	1	ADC H	1000	1100	DB	2	IN	56	1	MOV D, M	1F	1	RAR
8D	1	ADC L	1000	1101	3C	1	INR A	5F	1	MOV E, A	D8	1	RC
8E	1	ADC M	1000	1110	04	1	INR B	58	1	MOV E, B	C9	1	RET
87	1	ADD A	1000	0111	0C	1	INR C	59	1	MOV E, C	07	1	RLC
80	1	ADD B	1000	0000	14	1	INR D	5A	1	MOV E, D	F8	1	RM
81	1	ADD C	1000	0001	1C	1	INR E	5B	1	MOV E, E	D0	1	RNC
82	1	ADD D	1000	0010	24	1	INR H	5C	1	MOV E, H	C0	1	RNZ
83	1	ADD E	1000	0011	2C	1	INR L	5D	1	MOV E, L	F0	1	RP
84	1	ADD H	1000	0100	34	1	INR M	5E	1	MOV E, M	E8	1	RPE
85	1	ADD L	1000	0101	03	1	INX B	67	1	MOV H, A	E0	1	RPO
86	1	ADD M	1000	0110	13	1	INX D	60	1	MOV H, B	0F	1	RRC
C6	2	ADI	1100	0110	23	1	INX H	61	1	MOV H, C	C7	1	RST 0
A7	1	ANA A	1010	0111	33	1	INX SP	62	1	MOV H, D	CF	1	RST 1
A0	1	ANA B	1010	0000	DA	3	JC	63	1	MOV H, E	D7	1	RST 2
A1	1	ANA C	1010	0001	FA	3	JM	64	1	MOV H, H	DF	1	RST 3
A2	1	ANA D	1010	0010	C3	3	JMP	65	1	MOV H, L	E7	1	RST 4
A3	1	ANA E	1010	0011	D2	3	JNC	66	1	MOV H, M	EF	1	RST 5
A4	1	ANA H	1010	0100	C2	3	JNZ	6F	1	MOV L, A	F7	1	RST 6
A5	1	ANA L	1010	0101	F2	3	JP	68	1	MOV L, B	FF	1	RST 7
A6	1	ANA M	1010	0110	EA	3	JPE	69	1	MOV L, C	C8	1	RZ
E6	2	ANI	1110	0110	E2	3	JPO	6A	1	MOV L, D	9F	1	SBB A
CD	3	CALL	1100	1101	CA	3	JZ	6B	1	MOV L, E	98	1	SBB B
DC	3	CC	1101	1100	3A	3	LDA	6C	1	MOV L, H	99	1	SBB C
FC	3	CM	1111	1100	0A	1	LDAX B	6D	1	MOV L, L	9A	1	SBB D
2F	1	CMA	0010	1111	1A	1	LDAX D	6E	1	MOV L, M	9B	1	SBB E
3F	1	CMC	0011	1111	2A	3	LHLD	77	1	MOV M, A	9C	1	SBB H
BF	1	CMP A	1011	1111	01	3	LXI B	70	1	MOV M, B	9D	1	SBB L
B8	1	CMP B	1011	1000	11	3	LXI D	71	1	MOV M, C	9E	1	SBB M
B9	1	CMP C	1011	1001	21	3	LXI H	72	1	MOV M, D	DE	2	SBI
BA	1	CMP D	1011	1010	31	3	LXI SP	73	1	MOV M, E	22	3	SHLD
BB	1	CMP E	1011	1011	7F	1	MOV A, A	74	1	MOV M, H	F9	1	SPHL
BC	1	CMP H	1011	1100	78	1	MOV A, B	75	1	MOV M, L	32	3	STA
BD	1	CMP L	1011	1101	79	1	MOV A, C	3E	2	MVI A	02	1	STAX B
BE	1	CMP M	1011	1110	7A	1	MOV A, D	06	2	MVI B	12	1	STAX D
D4	3	CNC	1101	0100	7B	1	MOV A, E	0E	2	MVI C	37	1	STC
C4	3	CNZ	1100	0100	7C	1	MOV A, H	16	2	MVI D	97	1	SUB A
F4	3	CP	1111	0100	7D	1	MOV A, L	1E	2	MVI E	90	1	SUB B
EC	3	CPE	1110	1100	7E	1	MOV A, M	26	2	MVI H	91	1	SUB C
FE	3	CPI	1111	1110	47	1	MOV B, A	2E	2	MVI L	92	1	SUB D
E4	3	CPO	1110	0100	40	1	MOV B, B	36	2	MVI M	93	1	SUB E
CC	3	CZ	1100	1100	41	1	MOV B, C	00	1	NOP	94	1	SUB H
27	1	DAA	0010	0111	42	1	MOV B, D	B7	1	ORA A	95	1	SUB L
09	1	DAD B	0000	1001	43	1	MOV B, E	B0	1	ORA B	96	1	SUB M
19	1	DAD D	0001	1001	44	1	MOV B, H	B1	1	ORA C	D6	2	SUI
29	1	DAD H	0010	1001	45	1	MOV B, L	B2	1	ORA D	EB	1	XCHG
39	1	DAD SP	0011	1001	46	1	MOV B, M	B3	1	ORA E	AF	1	XRA A
3D	1	DCR A	0011	1101	4F	1	MOV C, A	B4	1	ORA H	A8	1	XRA B
05	1	DCR B	0000	0101	48	1	MOV C, B	B5	1	ORA L	A9	1	XRA C
0D	1	DCR C	0000	1101	49	1	MOV C, C	B6	1	ORA M	AA	1	XRA D
15	1	DCR D	0001	0101	4A	1	MOV C, D	F6	2	ORI	AB	1	XRA E
1D	1	DCR E	0001	1101	4B	1	MOV C, E	D3	2	OUT	AC	1	XRA H
25	1	DCR H	0010	0101	4C	1	MOV C, H	E9	1	PCHL	AD	1	XRA L
2D	1	DCR L	0010	1101	4D	1	MOV C, L	C1	1	POP B	AE	1	XRA M
35	1	DCR M	0011	0101	4E	1	MOV C, M	D1	1	POP D	EE	2	XRI
0B	1	DCX	0000	1011	57	1	MOV D, A	E1	1	POP H	E3	1	XTHL

Учебное издание

Константин Алексеевич Палагута

Микропроцессоры INTEL 8080, 8085 (КР580ВМ80А, КР1821ВМ85А) и их программирование

Учебное пособие

Редактор *Н.А. Киселева*
Компьютерная верстка *Р.Д. Рахматуллоева*
Оформление обложки *А.М. Гришиной*

Санитарно-эпидемиологическое заключение
№ 77.99.02.953.Д.002624.03.06. от 30.03.2006

Подписано в печать 30.11.2006
Формат бум. 60x84/16 Изд. № 3-45/06
Усл. печ. л. 6,75 Уч.-изд. л. 7,0 Тираж 1000
Заказ № 845

Издательство МГИУ, 115280, Москва, Автозаводская, 16

По вопросам приобретения продукции Издательства МГИУ обращаться по адресу:
115280, Москва, Автозаводская, 16
www.izdat.msiu.ru; E-mail: izdat@msiu.ru; тел. (495) 677–23–15

Отпечатано в типографии издательства МГИУ

ISBN 978-5-276-01040-3