Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

Новгородский государственный университет
имени Ярослава Мудрого

Гурьянов С.А.

СТРУКТУРА МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

Новгород
2012

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

Новгородский государственный университет
имени Ярослава Мудрого
Институт электронных и информационных систем

Кафедра радиосистем

Гурьянов С.А.

СТРУКТУРА МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

Курс лекций по дисциплине
"ЦИФРОВЫЕ УСТРОЙСТВА И МИКРОПРОЦЕССОРЫ"
для направления 210300.62 "Радиотехника"

Новгород
2012

УДК 621.38
ББК

Гурьянов С.А. Структура микропроцессорной системы: Курс лекций / ФГБОУ «Новгородский государственный университет им. Ярослава Мудрого», Великий Новгород, 2012 г. - 51 с.

В учебном пособии рассмотрены вопросы: представление данных в цифровых системах, структура микропроцессорной системы, структура микропроцессора, система команд, примеры программ.

Учебное пособие отвечает новым образовательным стандартам и предназначено для подготовки бакалавров по направлению 210300.62 "Радиотехника".

СОДЕРЖАНИЕ

стр.

ВВЕДЕНИЕ

Применение интегральных микросхем в радиотехнических системах позволило существенно улучшить их характеристики: энергопотребление, габариты, масса, стоимость, а также октрыло широкие возможности для реализации сложных алгоритмов обработки сигналов. С появлением микропроцессоров появилась возможность реализации алгоритмов обработки сигналов близких к оптимальным с минимальными затратами.

Микропроцессор (МП) представляет собой цифровое устройство в виде одной или нескольких больших интегральных схем, выполняющее различные операции по обработке данных в соответствии с заданной программой.

Замечательным свойством микропроцессорных систем является их высокая гибкость, возможность быстрой перенастройки при необходимости даже значительных изменений алгоритмов управления. Как правило, перенастройка осуществляется программным путём без существенных производственных затрат. Кроме того, микропроцессоры позволяют легко реализовать принципы открытых систем, функциональные возможности которых могут наращиваться по мере необходимости или по мере появления новых технических средств. Тем самым обеспечивается соответствие технического уровня микропроцессорных систем управления самым современным требованиям в течение длительного времени.

Характерной особенностью развития средств вычислительной техники является широкое применение трех принципов “М” - модульность, магистральность, микропрограммируемость.

Принцип модульной организации предполагает построение вычислительной системы на основе набора конструктивно, функционально и электрически законченных модулей, которые позволяют самостоятельно решать некоторый круг вычислительных задач или задач управления, а также взаимодействовать с другими модулями.

Среди способов организации связи элементов внутри модулей и между модулями в системе можно выделить два основных: с помощью произвольных связей, реализующий принцип “каждый с каждым”, и с помощью упорядоченных связей (магистральный), позволяющий минимизировать число связей. Магистральный способ обеспечивает обмен информацией

между функциональными и конструктивными модулями различного уровня с помощью магистралей, объединяющих входные и выходные шины.

Различают одно-, двух-, трех- и многомагистральные связи. В микропроцессорных системах обычно используется трехмагистальный метод обмена информацией, при этом выделяются следующие магистрали:

- шина адреса;
- шина данных;
- шина управления.

Использование магистральной организации обмена информацией является одним из способов обеспечения регулярности связей между конструктивными модулями микропроцессорных систем. Магистральный способ обмена позволяет минимизировать количество связей между блоками, повысить регулярность операционного устройства и устройства управления, обеспечить стандартизацию интерфейсов, сократить число выводов БИС. Необходимо отметить взаимосвязь схемотехнических и структурных решений, которая проявляется при реализации данного способа обмена в виде создания специальных двунаправленных буферных каскадов с тремя устойчивыми состояниями и при использовании временного мультиплексирования каналов обмена.

Микропрограммное управление обеспечивает наибольшую гибкость при организации многофункциональных микропроцессорных модулей и позволяет осуществить проблемную ориентацию микропроцессорной системы.

Микропрограммное управление за счёт возможности смены микропрограмм повышает гибкость устройства, увеличивает их регулярность, за счёт рассредоточенности управления и распределённости памяти обеспечивает параллелизм решения задач, за счёт применения серийно освоенных БИС повышает надёжность системы, и за счёт регулярности структуры упрощает контроль функционирования устройства. Передача управляющих слов в виде зашифрованных кодовых последовательностей соответствует условиям минимизации числа выводов БИС и снижению числа межсоединений в модулях.

Универсальность микропроцессорных систем обеспечивает их доступность широкому кругу потребителей. Наряду с потенциально низкой стоимостью это определяется и возможностью централизации производства универсальных микропроцессорных систем, специализация которых под

конкретные задачи (т.е. программирование) осуществляется самим потребителем.

1. ФОРМАТ ПРЕДСТАВЛЕНИЯ ДАННЫХ В МИКРОПРОЦЕССОРНЫХ СИСТЕМАХ

Для представления данных в цифровой технике используют двоичную систему счисления, поскольку в цифровых микросхемах сигнал может иметь только два значения: 0 или 1, представляемые, соответственно, низким L-уровнем и высоким H-уровнем напряжения. Один разряд двоичного кода называется битом.

Данные в микропроцессорных системах представляются в виде 8-разрядных кодов. Восемь двоичных разрядов называется байтом. Для идентификации отдельных разрядов в байте они нумеруются от 0 до 7 и располагаются справа налево. При этом нулевой бит 0 соответствует младшему разряду, а 7 - старшему разряду байта. Каждый разряд имеет свой "вес", т.е. каждому разряду соответствует двоичное число равное числу 2, возведенному в степень номера разряда. Два байта объединяются в слово (16 двоичных разрядов).

В микропроцессорных системах используется два типа представления чисел: без знака и со знаком. Для представления чисел без знака используется прямой код числа. Восемь двоичных разрядов позволяют представлять числа без знака в диапазоне от 0 до 255 (00000000 – “0” и 11111111 – “255”).

Для представления чисел со знаком используется старший разряд числа. Единица в старшем разряде числа соответвует отрицательному числу, а ноль – положительному. В этом случае отрицательные числа представляются в дополнительном коде. Инверсия всех двоичных разрядов данных прямого кода позволяет получить обратный код числа. При прибавлении 1 к обратному коду получается дополнительный код числа. Восемь двоичных разрядов позволяют представлять числа со знаком в диапазоне от –128 до 127 (1000000 – “-128” и 01111111 – “127”). В таблице 1 приведено представление двоичных чисел в прямом, обратном и дополнительном коде, а также их десятичное представление как чисел без знака, так и со знаком.

Таблица 1

Прямой двоичный код числа	Обратный код числа	Дополни-тельный код числа	Десятичное число без знака	Десятичное число со знаком
00000000	11111111	00000000	0	0
00000001	11111110	11111111	1	1
00000010	11111101	11111110	2	2
. . .				
01111110	10000001	10000010	126	126
01111111	10000000	10000001	127	127
10000000	01111111	10000000	128	-128
10000001	01111110	01111111	129	-127
. . .				
11111101	00000010	00000011	253	-3
11111110	00000001	00000010	254	-2
11111111	00000000	00000001	255	-1

Для удобства оперирования двоичными данными используют их шестнадцатеричное представление: байт делится пополам на две группы (тетрады), по четыре разряда в каждой. Каждая тетрада представляется одним шестнадцатеричным числом (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F). При этом, "вес" тетрады равен числу 16, возведенному в степень номера тетрады. Младшей тетраде в байте соответствует вес 16^0=1, старшей тетраде – 16^1=16.

Таблица 2

Двоичный код тетрады	Шестандцатеричное представление числа	Двоичный код тетрады	Шестандцатеричное представление числа
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

Кодирование шестандцатеричных чисел представлено в таблице 2.

Для различения систем счисления в конце числа пишется индекс соответствующей системы счисления:

двоичная система - b,

десятичная система - d,

шестнадцатеричная система - h.

При записи числа в десятичной системе счисления индекс может отсутствовать. При записи чисел в шестнадцатеричной системе счисления необходимо добавлять число 0 перед числами, начинающимися с буквы: A,B,C,D,E и F. Пример записи чисел в различных системах счисления представлен в таблице 3.

Таблица 3

Десятичная	Двоичная	Шестнадцатеричная
10	00001010b	0Ah
16	00010000b	10h
36	00100100b	24h
100	01100100b	64h
172	10101100b	0ACh
255	11111111b	0FFh

2. СТРУКТУРА МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

Обобщенная структура микропроцессорной системы представлена на рис.1. Микропроцессорная система включает в себя

- микропроцессор – основной блок, определяющий параметры микропроцессорной системы,
- память программ, реализованную на постоянном запоминающем устройстве (ПЗУ),
- память данных, использующую оперативное запоминающее устройство (ОЗУ) для хранения информации,
- различные устройства ввода-вывода, определяющие взаимосвязь микропроцессорной системы с внешними блоками всей радиотехнической системы.

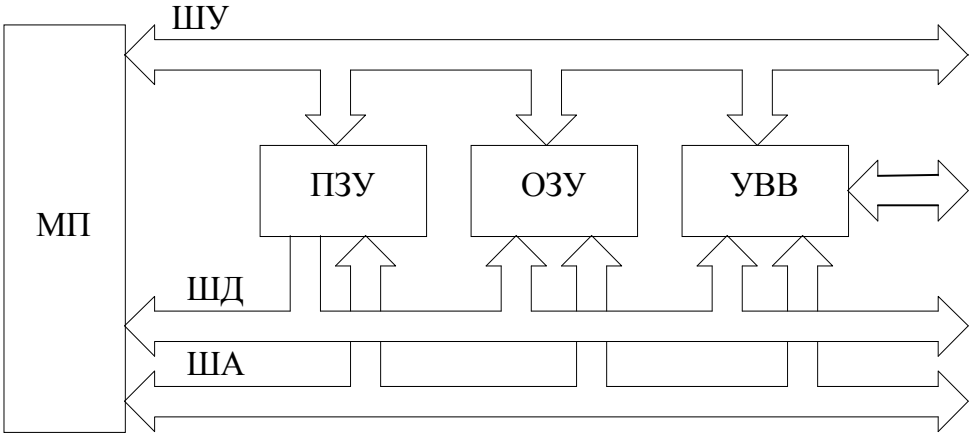


Рис. 1. Стуктура микропроцессорной системы

При работе микропроцессорной системы основной обмен информацией производится между микропроцессором, постоянными и оперативными запоминающими устройствами (ЗУ) и устройствами ввода-вывода (УВВ).

Для достижения высокой скорости передачи данных в микропроцессорных системах используется шинная организация системы. Обычно используется трехшинный метод обмена информацией, при этом выделяются следующие шины:

- шина адреса;
- шина данных;
- шина управления.

Основными сигналами управления в микропроцессорной системе являются сигналы

- чтение из памяти (RDM),
- запись в память (WRM),
- чтение из устройства ввода-вывода (RDIO),
- запись в устройство ввода-вывода (WRIO).

Раздельные шины данных и адреса (рис. 1) характерны для большинства микропроцессорных систем. Существуют системы и с совмещенной шиной адреса-данных (ШАД). Выделение отдельно шин для всех управляющих сигналов (ШУ), адресной информации (ША) и данных (ШД) упрощает организацию обмена информацией между отдельными блоками и уменьшает время выполнения команд в микропроцессорной системе.

Микропроцессор выполняет определенный набор простых операций, называемых системой команд. Конкретная задача решается путем выполнения заданной последовательности команд, называемой программой функционирования системы. Программа размещается в памяти команд (ПЗУ или ОЗУ). Микропроцессор в заданной последовательности выбирает команды из памяти и выполняет их. Для выбора команд из памяти микропроцессор использует адресную шину для адрессации ячеек памяти и шину управления для формирования сигналов сопровождения.
Код команды, выбранной из заданной ячейки памяти, вводится в микропроцессор по шине данных и вызывает вполнение соотвествующей операции. После дешифрации команды микропроцессор переходит к выполнению заданной команды и формированию необходимых управляющих сигналов на шине управления.

При этом микропроцессор использует один и тот же набор сигналов управления как для связи с ячейками запоминающих устройств, так и для связи с периферийными устройствами. Каждой ячейке памяти ОЗУ, ПЗУ, регистрам процессора и регистрам периферийных устройств аппаратным образом присваивается определённый адрес в системе. Благодаря такой структуре все команды для данных, хранящихся в оперативной памяти, в равной мере могут использоваться и для данных в регистрах периферийных устройств.

Эффективность решения задач в микропроцессорной системе в значительной степени определяется структурой связи между микропроцессором, памятью и УВВ, а также организацией обмена между ними. Система шин, вспомогательной аппаратуры и алгоритмов, предназначенная для

организации обмена между микропроцессором, памятью и УВВ, называется интерфейсом. В функции интерфейса входит дешифрация адреса устройств, синхронизация обмена информацией, согласование форматов слов, дешифрация кода команды, связанной с обращением к памяти или УВВ, электрическое согласование сигналов и некоторые другие операции.

Сложность задач, возлагаемых на интерфейс, а также недостаточная мощность буферных схем, входящих в состав БИС микропроцессора, привели к распределению средств интерфейса между различными устройствами:

• устройством управления памятью и вводом-выводом, входящим в состав микропроцессора;

• непосредственно интерфейсным устройством, являющимся промежуточным звеном между микропроцессором, с одной стороны, и памятью и УВВ, с другой;

• специализированными устройствами управления (контроллерами) УВВ, предназначенными для реализации алгоритмов управления специфических для различных УВВ.

Организация обмена между микропроцессором и памятью или УВВ в простейших случаях возможна на основе средств, содержащихся только в микропроцессоре. Недостающие функции в таких случаях реализуются программно.

Более сложные ЗУ и УВВ соединяются с микропроцессором обязательно через дополнительные интерфейсные устройства выполненные в виде специальных БИС.

Наконец, существуют сложные ЗУ и УВВ со специфическими алгоритмами управления (магнитные диски, электронно-лучевые трубки и т.д.), реализация которых возможна лишь специальными контроллерами.

Различают три способа организации связи между МП и УВВ:

• программно – управляемая передача данных;

• использование прерываний;

• прямой доступ к памяти (ПДП).

3. СТРУКТУРА 8-РАЗРЯДНОГО МИКРОПРОЦЕССОРА

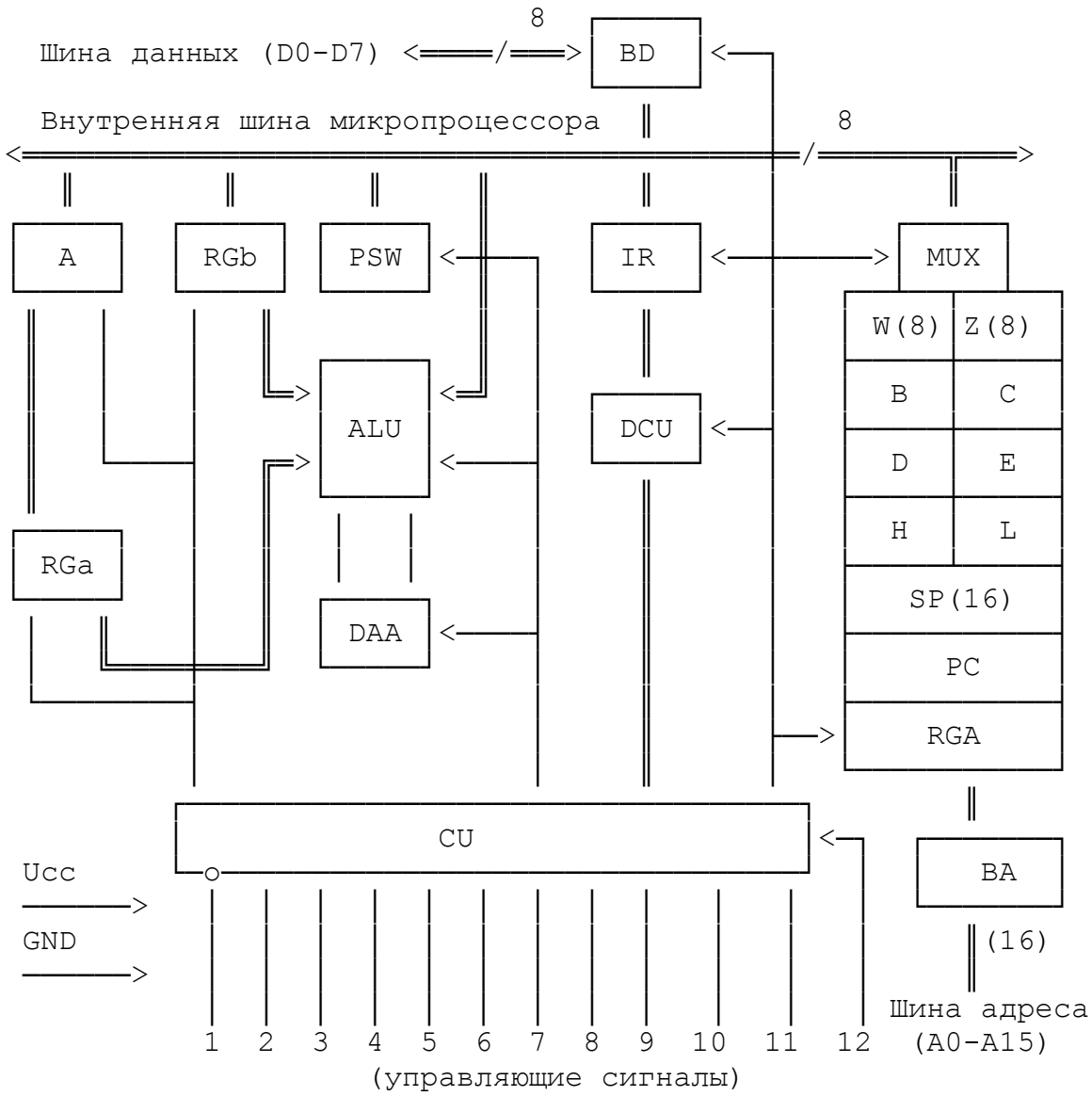


Рис.2. Структура микропроцессора I8080

Примеров 8-разрядного микропроцессора является микропроцессор фирмы INTEL I8080. Даный микропроцессор является функционально законченным однокристальным параллельным восьмиразрядным микропроцессором с фиксированной системой команд. Этот микропроцессор является типичным предствителем семейства восьмиразрядных процессоров. К этому семейству относятся микропроцессоры I8080, I8085, Z80, КМ580ВМ80, К1821ВМ85. Конструктивно микропроцессор выполнен в виде одной большой интегральной микросхемы в 40-выводном корпусе.

Микропроцессор имеет 16-разрядную шину адреса (ША), 8-разрядную двунаправленную шину данных (ШД), четыре входных и шесть выходных выводов сигналов управления. В микропроцессоре предусмотрены два

входа сигналов синхронизации, на которые необходимо подавать две тактовые последовательности, сдвинутые по фазе относительно друг друга. Внутренняя структура микропроцессора представлена на рис.2.

На рис.2 представлены следующие сигналы системной шины микропроцессора

выходные сигналы:

1 - WR, 2 - DBIN, 3 - INTE, 5 - HLDA, 7 - WAIT, 9 - SYNC.

входные сигналы:

4 - INT, 6 - HOLD, 8 - READY, 10 - CLK1, 11 - CLK2, 12 - RESET.

Микропроцессор I8080 включает в себя из следующие блоки:

- ALU - восьмиразрядное арифметико-логическое устройство.
- RS - регистр признаков, фиксирующий признаки, вырабатываемые ALU в процессе выполнения команд (C,AC,Z,S,P).
- A - аккумулятор.
- RGa – буферный регистр аккумулятора.
- RGb - регистр временного хранения операндов.
- DAA - десятичный корректор, выполняющий перевод информации из двоичной в двоично-десятичную форму.
- IR - регистр команд, предназначенный для хранения первого байта команды, содержащей код операции.
- DCU - дешифратор команд.
- CU - схема управления и синхронизации, формирующая последовательности управляющих сигналов для работы ALU и блока регистров.
- MUX - двунаправленный мультиплексор для обмена операндами и результатами операций между ALU и блоком регистров по внутренней шине данных.
- RGA - регистр адреса.
- PC - программный счетчик, предназначенный для хранения текущего адреса выполняемой команды, который автоматически увеличивается в процессе выполнения команды на 1, 2 или 3 в зависимости от формата выполняемой команды.
- SP - указатель стека, в который вводится адрес вершины стека. При этом стек может использовать любую зону ОЗУ объемом до 64 Кбайт. Содержимое указателя стека уменьшается на 2 при загрузке данных в стек, и увеличивается на 2 при чтении данных из стека.

- B, C, D, E, H, L - шесть регистров общего назначения, которые при программировании могут быть объединены в три регистровые пары BC, DE и HL.
- W,Z - вспомогательные регистры, являющиеся служебными и предназначаются для приема второго и третьего байта команд.
- BD - восьмиразрядный буферный регистр данных.
- BA - шестнадцатиразрядный буферный регистр адреса.

Перечень входных и выходных сигналов микропроцессора I8080 с указанием их назначения приведены в таблице 4. В таблице приняты следующие обозначения:

L-уровень – низкий (нулевой) уровень сигнала,

H-уровень – высокий (единичный) уровень сигнала.

Таблица 4

Обозначение выводов	Номера контактов	Назначение выводов
A15-A0	25,26,27,29,30,31,32,33 34,35,1,40,37,38,39,36	Трехстабильная 16-разрядная шина адреса (ША)
D7-D0	10,9,8,7,3,4,5,6	Двунаправленная трехстабильная 8-разрядная шина данных (ШД)
WR	18	Выход сигнала “выдача”: L-уровень указывает на выдачу байта информации на шину данных для записи в ЗУ и УВВ
DBIN	17	Выход сигнала “прием”: H-уровень указывает на прием с шины данных байта информации выданного ЗУ или УВВ
INTE	16	Выход сигнала “разрешение прерывания”: H-уровень разрешает работу внешнего прерывания
INT	14	Вход сигнала “запрос на прерывание”: H-уровень указывает на запрос на прерывание от внешнего устройства
HLDA	21	Выход сигнала “подтверждение захвата”: H-уровень указывает

		на перевод шины адреса и шины данных в высокоимпедансное состояние при работе в режиме прямого доступа к памяти (ПДП)
HOLD	13	Вход сигнала “захват”: Н-уровень сигнала указывает на запрос другими устройствами системы на управление шинами в режиме ПДП
WAIT	24	Выход сигнала “ожидание”: Н-уровень сигнала указывает на состояние ожидания микропроцессора
READY	23	Вход сигнала “готовность”: напряжение Н-уровня указывает на готовность данных на шине данных к вводу в микропроцессор или на готовность внешних устройств к приему информации (служит для синхронизации микропроцессора с ЗУ или УВВ)
SYNC	19	Выход сигнала “синхро”: сигнал Н-уровня идентифицирует начало каждого машинного цикла
CLK1	22	Вход тактовой последовательности фазы 1
CLK2	15	Вход тактовой последовательности фазы 2
RESET	12	Вход начальной установки: L-уровень сигнала устанавливает счетчик команд в нулевое состояние
Ucc1	28	Напряжение питания (+12В)
Ucc2	20	Напряжение питания (+5В)
Ucc3	11	Напряжение питания (-5В)
GND	2	Общий

4. СИСТЕМА КОМАНД 8-РАЗРЯДНОГО МИКРОПРОЦЕССОРА

Микропроцессор является цифровой схемой, выполняющей различные операции по обработке данных в соответствии с заданной в памяти программой. Программа функционирования микропроцессорной системы представляется в виде зашифрованных кодовых последовательностей.

Язык программы, "понятный" микропроцессору, называется машинным языком программирования. При этом для записи алгоритма используется набор инструкций, представленных в двоичном коде. Для упрощения записи команды чаще используется шестнадцатеричная форма. Для наглядности программирования двоичная запись в машинных кодах заменяется на мнемонику. В этом случае шестнадцатеричная форма записи заменяется короткими мнемоническими словами, образованными в результате аббревиатурных сокращений полных названий или фраз, указывающих смысл команды. Заменив каждый код машинного языка мнемоникой, можно в наглядном виде написать программу функционирования микропроцессора. Мнемоническая запись разработанной программы может быть преобразована специальной программой непосредственно в двоичные машинные коды. Программы, написанные с использованием мнемоники, называются программами на языке Ассемблер, а программа, преобразующая мнемонические команды непосредственно в двоичные коды, называется программой Ассемблер.

Программа работы микропроцессора, представленная в машинных кодах, представляет собой список команд, который храниться в памяти программ микропроцессорной системы. Микропроцессор начинает работу, считывая первый код программы из памяти, расшифровывает команду и выполняет указанную в ней операцию. Затем процессор считывает команду из следующей ячейки памяти и снова выполняет соответствующую операцию. Длина команды может изменяться от 1 до 3 байт в зависимости от ее назначения. Определенные команды позволяют микропроцессору перейти на другой адрес памяти программ, минуя установленный порядок очередности, для выполнения текущей команды. Встречающиеся в теле программы, повторяющиеся последовательности команд могут быть выделены в самостоятельные образования, именуемые подпрограммами, к которым основная программа может обращаться на любом шаге своего выполнения. Подпрограммы, в свою очередь, тоже могут включать в себя

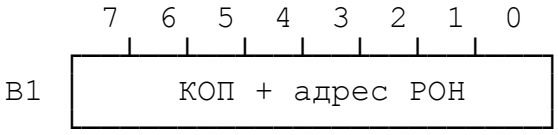
новые подпрограммы. Многократность вложений подпрограмм характеризуется такой величиной, как глубина обращений к подпрограммам. Глубина обращений к подпрограммам определяется при разработке микропроцессорной системы глубиной области памяти, отведенной под стек.

Каждая команда представляет собой последовательность микрокоманд, названных машинными циклами, которые выполняются за несколько периодов тактового генератора. Каждый машинный цикл представляет собой обращение (чтение или запись) к внутренним регистрам микропроцессора, запоминающим устройствам или устройствам ввода-вывода системы. Поэтому время выполнения конкретной команды определяется числом машинных циклов, равным суммарному числу обращений ЗУ или УВВ, необходимых для выборки и выполнения этой команды.

4.1.Формат команд микропроцессроа и методы адресации

Коды команд микропроцессора организованы в виде 8-разрядных слов и по формату подразделяются на одно-, двух- или трехбайтные:

- однобайтные



- двухбайтные

- трехбайтные

При размещении в памяти программ байты команд располагаются последовательно по возрастанию адресов и побайтно вводятся в микропроцессор. Многобайтная команда размещается в последовательно расположенных ячейках памяти. В первом байте команды B1 всегда указывается код операции (КОП), определяющий также и ее формат. Поэтому после ввода B1 в регистр команд и его дешифрации микропроцессор или продолжает ввод последующих байтов команды (для двух- или трехбайтных команд) или переходит к выполнению операции (при однобайтной команде).

B1 указывает код операции, метод адресации, а также в однобайтных командах адреса источников операндов.

B2 в двухбайтных командах задает адрес УВВ в командах ввода/вывода или является операндом - числом, участвующим в операции.

B2,B3 в трехбайтных командах задают адрес ячейки ОЗУ при прямой адресации или адрес управляющей памяти в командах управления, когда необходимо перейти на новую последовательность команд, а также B2,B3 участвуют как 16 - разрядная константа, загружаемая в регистровую пару или указатель стека.

B3 является старшими разрядами адреса, B2 - младшими.

Для управления процессом выполнения программы используется слово состояние программы, формат которого показан на рис.2. Старший байт слово состояния представляет содержимое аккумулятора, а младший - содержит флаги условий регистра признаков, определяемые результатом выполнения арифметических и логических операций. В результате выполнения арифметических и логических операций формируются признаки:

- C – признак переноса при сложении или заема при вычитании,
- P – признак четности, определяется по количеству единиц в операнде,
- AC – дополнительный перенос, формирующийся при переносе из 3 разряда операнда в 4,
- Z – признак равенства нулю,
- S – признак знака, формирующийся в соотвествии с 7 разрядом операнда.

Аккумулятор								Регистр признаков							
D7	D6	D5	D4	D3	D2	D1	D0	S	Z	0	AC	0	P	1	C

Рис.3. Формат словосостояния программы

В представленном микропроцессоре используется пять способов адресации:

- регистровая - в команде задается адрес оперативного регистра или пары регистров, где находится, соответственно, 8- или 16-битовый операнд;
- непосредственная - операнд содержится в команде: для двухбайтовых команд - во втором байте, для трехбайтовых команд – во втором

(младшая часть операнда) и в третьем (старшая часть операнда) байтах команды;

- прямая - адрес ячейки памяти, где расположен операнд, указывается во втором (младшая часть адреса) и в третьем (старшая часть адреса) байтах команды;
- регистровая косвенная - адрес ячейки памяти, где расположен операнд, определяется содержимым регистровой пары, явно или неявно указанного в команде; при этом, старший байт адреса находится в первом регистре пары, а младший - во втором;
- стековая - адрес ячейки памяти, содержащей операнд, находится в указателе стека.

Специфический способ адресации памяти используется в однобайтовой команде RST, применяемой при обработке прерывания для вызова одной из восьми подпрограмм обслуживания прерываний. Команды RST различаются по номеру N, задаваемому в трехбайтовом поле кода команды. В результате выполнения команды RST N управление передается по адресу, определяемому восьмикратным увеличением N.

4.2. Состав системы команд

Система команд микропроцессора I8080 содержит 78 команд, включающих 111 операций. Полный перечень команд МП приведен в приложении 1.

По функциональному признаку команды микропроцессора можно разделить на пять групп:

1. команды передачи данных
   - из регистра в регистр,
   - из регистра в память,
   - из памяти в регистр;
2. арифметические команды:
   - сложение,
   - вычитание,
   - инкремент,
   - декремент;
3. логические команды:
   - операция И,
   - операция ИЛИ,
   - операция исключающее ИЛИ,
   - сравнение,
   - сдвиг,
   - инвертирование;
4. команды передачи управления и обработки подпрограмм;
5. команды ввода/вывода и управления состоянием МП.

4.2.1. Команды передачи данных

MOV DD,SS - пересылка данных между регистрами общего назначения A,B,C,D,E,H,L с использованием регистровой адресации или между регистрами и памятью (M) с использованием косвенной адресации через регистровую пару HL. Содержимое регистра источника при этом не изменяется. DD – получатель, SS – источник информации.

MVI D, D8 - непосредственная загрузка регистра или ячейки памяти байтом данных D8, содержащимся во втором байте команды.

LDA ADR - загрузка аккумулятора из памяти по указанному адресу ADR.

STA ADR -запись содержимого аккумулятора по адресу ADR, указанному в адресной части команды.

LDAX RP - загрузка аккумулятора из памяти по адресу, указанному в паре регистров RP (BC или DE).

STAX RP - запись содержимого аккумулятора по адресу, указанному в паре регистров RP (BC или DE).

LHLD ADR - загрузка регистровой пары HL двумя байтами из памяти, с адреса ADR, указанного в команде.

SHLD ADR - запись содержимого регистровой пары HL в памяти по адресу ADR, указанному в команде.

LXI RP,D16 - непосредственная загрузка регистровых пар (BC, DE, HL) или указателя SP константой D16.

PUSH RP - запись в стек содержимого регистровой пары или PSW.

POP RP - извлечение из стека 2 байт данных и помещение их в регистровую пару или в PSW.

XCHG - обмен 2-байтовыми словами между регистровыми парами DE и HL.

XTHL - обмен содержимым между верхушкой стека и регистровой парой HL.

SPHL - пересылка регистровой пары HL в указатель стека (позволяет изменить адрес памяти, отведенной под стек).

PCHL - пересылка регистровой пары HL в программный счетчик (позволяет организовать косвенные переходы).

IN N - прием данных в аккумулятор из порта с номером N.

OUT N - вывод данных из аккумулятора в порт N.

4.2.2. Арифметические и логические команды

ADD SS - сложение содержимого аккумулятора с содержимым регистров общего назначения (РОН) или ячейки памяти (M), адресуемой парой HL. Результат сохраняется в аккумуляторе и устанавливаются признаки.

ADC SS - сложение содержимого аккумулятора с содержимым регистров общего назначения (РОН) или ячейки памяти (M), адресуемой парой HL, с учетом бита переноса CY.

SUB SS - вычитание из аккумулятора содержимого РОН или ячейки памяти (M), адресуемой парой HL.

SBB SS - вычитание из аккумулятора содержимого РОН или ячейки памяти (M), адресуемой парой HL, с учетом заема.

ADD D8, ADC D8, SUB D8, SBB D8 - сложение содержимого аккумулятора

с константой D8, или вычитание из аккумулятора константы D8.

ANA SS, ORA SS, XRA SS – логические операции И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ над содержимым аккумулятора и одним из РОН или ячейки памяти, косвенно адресуемой регистровой парой HL. Результат сохраняется в аккумуляторе, признаки устанавливаются.

ANI D8, ORI D8, XRI D8 – логические операции И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ над содержимым аккумулятора и константой D8.

CMP SS – сравнение содержимого аккумулятора и одного из РОН или ячейки памяти, косвенно адресуемой регистровой парой HL.

CPI D8 – сравнение содержимого аккумулятора и константы D8.

INR SS - инкремент (увеличение на 1) содержимого РОН или или ячейки памяти, косвенно адресуемой регистровой парой HL.

DCR SS - декремент (уменьшение на 1) содержимого РОН или или ячейки памяти, косвенно адресуемой регистровой парой HL.

INX X - инкремент (увеличение на 1) содержимого регистровой пары или указателя стека.

DCX X - декремент (уменьшение на 1) содержимого регистровой пары или указателя стека. Признаки НЕ УСТАНАВЛИВАЮТСЯ!

DAD X - сложение содержимого одной из пар регистров (BC,DE,HL) или указателя стека (SP) с содержимым регистровой пары HL. Результат помещается в HL, устанавливается признак переноса.

RAL – арифметический сдвиг содержимого аккумулятора влево (сдвиг через перенос).

RAR - арифметический сдвиг содержимого аккумулятора вправо (сдвиг через перенос).

RLC - циклический сдвиг содержимого аккумулятора влево.

RRC – циклический сдвиг содержимого аккумулятора вправо.

CMA - инверсия содержимого аккумулятора.

CMC - инверсия признака переноса.

STC - установка признака переноса в 1.

DAA - десятичная коррекция аккумулятора.

4.2.3. Команды передачи управления

Для изменения последовательного характера выполнения программы предусмотрены команды безусловного и условного переходов.

JMP ADR - безусловный переход на адрес ADR, указанный в команде.

Команды условных переходов представлены в таблице 5.

Таблица 5

Наименование команды	Условие перехода	Признак
JNZ ADR	Неравенство нулю	Z=0
JZ ADR	Равенство нулю	Z=1
JNC ADR	Отсутствие переноса	C=0
JC ADR	Наличие переноса	C=1
JPO ADR	Нечетность	P=0
JPE ADR	Четность	P=1
JP ADR	Положительный результат	S=0
JM ADR	Отрицательный результат	S=1

Команды переходов являются трехбайтными. Второй и третий байты команды задают полный 16-разрядный адрес перехода. При выполнении команд условных переходов, если значение признака соответствует значению, заданному кодом команды, т.е. условие выполняется, второй и третий байт загружаются в счетчик команд. При невыполнении условия содержимое счетчика команд не изменяется и продолжается последовательное выполнение программы.

Если некоторая последовательность команд используется в программе многократно, то данная последовательность может быть выделена как подпрограмма и занесена в память однократно. Для обращения к подпрограмме в систему команд включены команды вызова подпрограммы и возврата из подпрограмм.

CALL ADR – вызов подпрограммы с заданным начальным адресом ADR. При переходе на подпрограмму содержимое счетчика команд автоматически записывается в стек, а в счетчик команд загружается адрес подпрограммы ADR (второй и третий байт команды).

Для возврата из подпрограммы на очередной шаг основной программы используется команда *RET* – возврат из подпрограммы. По команде возврата происходит чтение данных из стека и запись в счетчик команд, что обеспечивает продолжение основной программы с прерванного адреса.

Аналогично командам условных переходов существуют команды условного вызова и возврата из подпрограммы, представленные в таблице 6.

Таблица 6

Наименование команды		Условие перехода	Признак
Вызов подпрограммы	Возврат из подпрограммы		
CNZ ADR	RNZ	Неравенство нулю	Z=0
CZ ADR	RZ	Равенство нулю	Z=1
CNC ADR	RNC	Отсутствие переноса	C=0
CC ADR	RC	Наличие переноса	C=1
CPO ADR	RPO	Нечетность	P=0
CPE ADR	RPE	Четность	P=1
CP ADR	RP	Положительный результат	S=0
CM ADR	RM	Отрицательный результат	S=1

RST N - переход на программу обслуживания прерывания. Обеспечивает обращение к восьми подпрограммам обслуживания прерывания.

EI,DI - команды установки и сброса триггера разрешения прерывания. Осуществляют программное разрешение или запрет прерываний.

HLT - команда останова.

NOP - холостая команда ("нет операции").

5. ПРИМЕРЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ АССЕМБЛЕР

5.1. Программы умножения

- Объем программы - 30 байт, время выполнения - 679 тактов
- Сдвиг множителя C вправо, сдиг результата HL вправо

```
Параметры:
C  - множимое
E  - множитель
HL - результат
MUL1: xra A
      mov  H,A    ;обнуление результата
      mov  L,A
      mov  D,A
      mov  A,E
      ora C       ;проверка сомножителей на равенство 0
      jz M3          ;выход из программы
      mvi B,8     ;счетчик циклов
M1:   mov A,C
      rrc         ;сдвиг множимого вправо
      mov C,A
      jnc M2      ;проверка разрядов множимого
      dad D       ;сложение DE+HL
M2:   mov A,H     ;сдвиг результата вправо
      rar
      mov H,A
      mov A,L
      rar
      mov L,A
      dcr B
      jnz M1
M3:   ret
```

- Объем программы - 22 байта, время выполнения - 499 тактов
- Сдвиг множителя A вправо, сдиг пром. результата DE влево

```
MUL2: xra A
      mov  H,A
      mov  L,A
      mov  D,A
      add  C
      jz M3
M2:   rar          ;сдвиг множителя
      jnc M1
      dad D        ;результат в HL
M1:   xchg
      dad H         ;сдиг второго сомножителя влево
      xchg
      ora A
      jnz M2
M3:   ret
```

- Объем программы - 22 байта, время выполнения - 423 такта

```
MUL3: xra A
      mov  H,A
      mov  L,A
      mov  D,A
      add  C
      jz M3
      xra A
      add E
      jz M3
      mov H,C
      mvi B,8
M1:   dad H
      jnc M2
      dad D
M2:   dcr B
      jnz M1
M3:   ret
```

- Объем программы - 25 байт, время выполнения - 510 тактов

```
Параметры:
C   - множимое
DE  - множитель
AHL - результат

MUL4: xra A
      mov H,A
      mov L,A
      add C
      jz M3
      xra A
      ora D
      ora E
      jz M3
      mov A,C
      mvi B,8
M1:   dad H
      ral
      jnc M2
      dad D
      aci 0
M2:   dcr B
      jnz M1
M3:   ret
```

5.2. Программы преобразования чисел

5.2.1. Преобразование двоично-десятичного числа в двоичное

Алгоритм преобразования двоично-десятичного числа в двоичное состоит в умножении старшей тетрады двоично-десятичного числа на 10 и сложения полученного результата с младшей тетрадой исходного числа.

Параметры:
A – входное и выходное число

```
        mov B,A
        ani 0Fh       ;маскирование и сохранение младшей тетрады
        mov C,A
        mov A,B
        ani 0F0h      ;умножение старшей тетрады на 8
        rrc
        mov B,A
        rrc
        rrc           ;умножение старшей тетрады на 2
        add B         ;сложение старшей тетрады(8+2)=10
        add C         ;сложение старшей и младшей тетрады
        ret
```

5.2.2. Преобразование двоичного числа в двоично-десятичное

Параметры:

C – входное число

HL – выходное число

Номер шага	CY	Регистр C	Регистр H	Регистр L	Результат
0	0	00001011	00000000	00000000	0
1	0	00010110	00000000	00000000	0
2	0	00101100	00000000	00000000	0
3	0	01011000	00000000	00000000	0
4	0	10110000	00000000	00000000	0
5	1	01100001	00000000	00000001	1
6	0	11000010	00000000	00000010	2
7	1	10000101	00000000	00000101	5
8	1	00001011	00000000	00010001	11

```
        lxi H,0
        mvi B,0
CICLE:  mov A,C       ;сдвиг регистра C влево
        rlc
        mov C,A
        mov A,L       ;вычисление HL*2+CY в двоично-десятичном виде
        adc L
        daa
        mov L,A
        mov A,H
        adc H
        daa
        mov H,A
        dcr B
        jnz CICLE     ;цикл преобразования
        ret
```

5.2.3. Преобразование двухбайтного двоичного числа в двоично-десятичное

Параметры:

BC – входное число

AHL – выходное число

```
        xra A
        mov H,A
        mov L,A
        mov D,B
        mov E,C          ;перенос BC в DE
        mov C,A
        mvi B,16         ;счетчик циклов
CICL:   xchg             ;сдвиг двоичного числа влево
        dad H
        xchg
        mov A,L          ;удвоение суммы с учетом переноса
                         ;в двоично-десятичном виде
        adc L
        daa
        mov L,A
        mov A,H
        adc H
        daa
        mov H,A
        mov A,C
        adc C
        daa
        mov C,A
        dcr B
        jnz CICL
        mov A,C
        ret
```

5.3. Программы работы с массивами

Различают пересылку массивов с начала и с конца в зависимости от их расположения в памяти системы. При перекрытии массивов пересылка с начала осуществляется, если начальный адрес массива источника больше начального адреса массива получателя. В противоположном случае осуществляется пересылка с конца. Если массивы не перекрываются, то возможно применение любого из представленных алгоритмов.

Параметры:
BC – адрес приемника
DE – адрес источника
HL – объем массива

5.3.1. Пересылка массива с конца

```
      push H
      dad D
      dcx H    ;определение адреса конца массива источника
      xchg
      pop H
      push H
      dad B
      dcx H    ;определение адреса конца массива приемника
      pop b
;DE – адрес конца массива источника
;HL – адрес конца массива приемника
;BC - объем массивов
  M1: ldax D   ;чтение данных из массива источника
      mov M,A  ;запись данных в массив приемника
      dcx B    ;модификация указателей адреса
      dcx D
      dcx H    ;модификация счетчика объема массивов
      mov A,B  ;проверка объема массива
      ora C
      jnz M1
      ret      ;выход из подпрограммы
```

5.3.2. Пересылка массива с начала

```
M1: ldax D   ;чтение данных из массива источника
    stax B   ;запись данных в массив приемника
    inx B    ;модификация указателей адреса
    inx D
    dcx H    ;модификация счетчика объема массивов
    mov A,H  ;проверка объема массива
    ora L
    jnz M1
    ret
```

5.3.3. Программа сравнения массивов

Параметры:

BC – адрес первого массива

DE – адрес второго массива

HL – объем массивов

При равенстве массивов A=0FFh (A=0)

```
       push H
       push D
       push B
       xchg          ;пересылка значения объема в DE
M1:    mov A,E       ;проверка счетчика циклов на 0
       ora D
       mvi A,0FFh
       jz M2         ;выход из программы
       ldax B        ;(BC)->A чтение элемента первого массива
       cmp M         ; A-M(HL)-/>
                      ; сравнение с элементом второго массива
       mvi A,0        ; xra A
       jnz M2        ;выход из программы при несовпадении

       inx B         ;модификация адресов
       inx H
       dcx D         ;модификация счетчика циклов
       jmp M1
M2:    pop B
       pop D
       pop H
       ret
```

5.3.4. Программа поиска максимума в массиве

```
Параметры входные:
HL – адрес массива
DE – объем массива
Параметы выходные:
C – максимум
        mvi    C,0          ; начальное значение MAX=0
LP_MAX: mov    A,M
        cmp    C            ; сравнение элемента с MAX
        jc     NO_MAX       ; переход, если MAX >= X(I)
        mov    C,A          ; новое значение MAX
NO_MAX: inx    H            ; указатель на новый элемент
        dcx    D            ; изменение счетчика
        mov    A,E
        ora    D
        jnz    LP_MAX
```

5.3.5.Программа поиска кода в массиве

Параметры входные:

C - искомый код

HL – адрес массива

DE – объем массива

Параметы выходные:

HL – адрес кода, если DE<>0

```
M1:    mov A,E     ;проверка счетчика циклов на 0
       ora D
       jz M2
       mov A,M     ;считывание кода из массива
       cmp C       ;сравнение с исходным кодом
       jz  M2      ;выход из подпрограммы, если код найден
       inx H       ;модификация счетчика циклов
       dcx D
       jmp M1
m2:    ret        ;выход из подпрограммы, если код не найден
```

5.4. Программа перекодировки текстовой строки

Код символа во входной строке является индексом для нахождения соотвествующего кода выходной строки в таблице перкодировки.

Параметры:
BC - адрес входной текстовой строки
DE - таблица перекодировки
HL - адрес выходной текстовой строки
A - длина строки

```
BEG:
      push PSW          ;push A
      push H
;формирование индекса по коду символа входной строки в HL
      mvi  H,0
      ldax B          ;(BC)->A чтение кода из входной строки
      mov  L,A
;получение соответствующего кода в выходной строке
      dad  D          ;DE+HL->HL  - адрес кода в таблице
      перекодировки
      mov A,M         ;чтение кода из таблицы перекодировки
      pop  H          ;восстановление адреса выходной строки
      mov  M,A        ;запись кода в выходную строку
      inx  H          ;модификация указателей адреса
      inx  B
      pop  PSW        ;восстановление счетчика
      dcr  A          ;счетчик циклов
      jnz  BEG
      ret
```

СПИСОК ЛИТЕРАТУРЫ

1. Нарышкин А.К. Цифровые устройства и микропроцессоры: Учеб.пособие для вузов. - М. : Академия, 2006. – 317 с.
2. Микушин А.В. Цифровые устройства и микропроцессоры: Учеб.пособие для вузов. - СПб. : БХВ-Петербург, 2010. - XIII, 818с.
3. Китаев Ю.В. Цифровые устройства и микропроцессоры: Учеб.пособие для вузов. http://makc-e.ucoz.ru/load/1-1-0-8
4. Безуглов Д.А. Цифровые устройства и микропроцессоры: Учеб.пособие для вузов. - Ростов н/Д : Феникс, 2006. - 468с.
5. Новиков, Ю. В. Основы микропроцессорной техники : учеб. пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - М.: Интернет-Университет Информационных Технологий : БИНОМ. Лаборатория знаний, 2009. - 357с.
6. Токхайм Р. Микропроцессоры: Курс и упражнения. /Пер. с англ. Под. ред. В.Н. Грасевича. М.: Энергоатомиздат, 1988, - 336с.
7. Гуртовцев А.Л., Гудыменко С.В. Программы для микропроцессоров: Справочное пособие. -Мн.: Высш. шк. ,1989, -352с.
8. Цифровые устройства и микропроцессоры. Рабочая программа. / Сост. Гурьянов С.А. – В.Новгород, 2011 – 12 с.

ПРИЛОЖЕНИЕ 1

Таблица команд микропроцессора I8080

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
	КОМАНДЫ ПЕРЕДАЧИ ДАННЫХ Однобайтные передачи данных					
78	MOV A,B	1	1	5	A <-- B	- - - - -
79	MOV A,C	1	1	5	A <-- C	- - - - -
7A	MOV A,D	1	1	5	A <-- D	- - - - -
7B	MOV A,E	1	1	5	A <-- E	- - - - -
7C	MOV A,H	1	1	5	A <-- H	- - - - -
7D	MOV A,L	1	1	5	A <-- L	- - - - -
7E	MOV A,M	1	2	7	A <-- (M)	- - - - -
7F	MOV A,A	1	1	5	A <-- A	- - - - -
40	MOV B,B	1	1	5	B <-- B	- - - - -
41	MOV B,C	1	1	5	B <-- C	- - - - -
42	MOV B,D	1	1	5	B <-- D	- - - - -
43	MOV B,E	1	1	5	B <-- E	- - - - -
44	MOV B,H	1	1	5	B <-- H	- - - - -
45	MOV B,L	1	1	5	B <-- L	- - - - -
46	MOV B,M	1	2	7	B <-- (M)	- - - - -
47	MOV B,A	1	1	5	B <-- A	- - - - -
48	MOV C,B	1	1	5	C <-- B	- - - - -
49	MOV C,C	1	1	5	C <-- C	- - - - -
4A	MOV C,D	1	1	5	C <-- D	- - - - -
4B	MOV C,E	1	1	5	C <-- E	- - - - -
4C	MOV C,H	1	1	5	C <-- H	- - - - -
4D	MOV C,L	1	1	5	C <-- L	- - - - -
4E	MOV C,M	1	2	7	C <-- (M)	- - - - -
4F	MOV C,A	1	1	5	C <-- A	- - - - -
50	MOV D,B	1	1	5	D <-- B	- - - - -
51	MOV D,C	1	1	5	D <-- C	- - - - -
52	MOV D,D	1	1	5	D <-- D	- - - - -
53	MOV D,E	1	1	5	D <-- E	- - - - -
54	MOV D,H	1	1	5	D <-- H	- - - - -
55	MOV D,L	1	1	5	D <-- L	- - - - -
56	MOV D,M	1	2	7	D <-- (M)	- - - - -
57	MOV D,A	1	1	5	D <-- A	- - - - -
58	MOV E,B	1	1	5	E <-- B	- - - - -
59	MOV E,C	1	1	5	E <-- C	- - - - -
5A	MOV E,D	1	1	5	E <-- D	- - - - -
5B	MOV E,E	1	1	5	E <-- E	- - - - -
5C	MOV E,H	1	1	5	E <-- H	- - - - -
5D	MOV E,L	1	1	5	E <-- L	- - - - -

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
5E	MOV E,M	1	2	7	E <-- (M)	- - - - -
5F	MOV E,A	1	1	5	E <-- A	- - - - -
60	MOV H,B	1	1	5	H <-- B	- - - - -
61	MOV H,C	1	1	5	H <-- C	- - - - -
62	MOV H,D	1	1	5	H <-- D	- - - - -
63	MOV H,E	1	1	5	H <-- E	- - - - -
64	MOV H,H	1	1	5	H <-- H	- - - - -
65	MOV H,L	1	1	5	H <-- L	- - - - -
66	MOV H,M	1	2	7	H <-- (M)	- - - - -
67	MOV H,A	1	1	5	H <-- A	- - - - -
6A	MOV L,D	1	1	5	L <-- D	- - - - -
6B	MOV L,E	1	1	5	L <-- E	- - - - -
6C	MOV L,H	1	1	5	L <-- H	- - - - -
6D	MOV L,L	1	1	5	L <-- L	- - - - -
6E	MOV L,M	1	2	7	L <-- (M)	- - - - -
6F	MOV L,A	1	1	5	L <-- A	- - - - -
71	MOV M,B	1	2	7	(M) <-- B	- - - - -
72	MOV M,C	1	2	7	(M) <-- C	- - - - -
73	MOV M,D	1	2	7	(M) <-- D	- - - - -
74	MOV M,E	1	2	7	(M) <-- E	- - - - -
75	MOV M,H	1	2	7	(M) <-- H	- - - - -
76	MOV M,L	1	2	7	(M) <-- L	- - - - -
77	MOV M,A	1	2	7	(M) <-- A	- - - - -
	Загрузка однобайтной константы					
06	MVI B,D8	2	2	7	B <-- D8	- - - - -
0E	MVI C,D8	2	2	7	C <-- D8	- - - - -
16	MVI D,D8	2	2	7	D <-- D8	- - - - -
1E	MVI E,D8	2	2	7	E <-- D8	- - - - -
26	MVI H,D8	2	2	7	H <-- D8	- - - - -
2E	MVI L,D8	2	2	7	L <-- D8	- - - - -
36	MVI M,D8	2	3	10	(M)<-- D8	- - - - -
3E	MVI A,D8	2	2	7	A <-- D8	- - - - -
	Чтение данных из памяти в аккумулятор					
3A	LDA ADR	3	4	13	A <-- (ADR)	- - - - -
0A	LDAX B	1	2	7	A <-- (BC)	- - - - -
1A	LDAX D	1	2	7	A <-- (DE)	- - - - -
	Запись данных из аккумулятора в память					
32	STA ADR	3	4	13	(ADR)<-- A	- - - - -
02	STAX B	1	2	7	(BC) <-- A	- - - - -
12	STAX D	1	2	7	(DE) <-- A	- - - - -

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
			Двухбайтные передачи данных			
E9	PCHL	1	1	5	PC <-- HL	- - - - -
F9	SPHL	1	1	5	SP <-- HL	- - - - -
E3	XTHL	1	5	18	HL <-->(SP+1),(SP)	- - - - -
EB	XCHG	1	1	4	HL <--> DE	- - - - -
			Запись двухбайтной константы			
01	LXI B,D16	3	3	10	BC <-- D16	- - - - -
11	LXI D,D16	3	3	10	DE <-- D16	- - - - -
21	LXI H,D16	3	3	10	HL <-- D16	- - - - -
31	LXI SP,D16	3	3	10	SP <-- D16	- - - - -
			Чтение из памяти двухбайтного числа			
2A	LHLD ADR	3	5	16	HL<--(ADR+1),(ADR)	- - - - -
			Запись двухбайтного числа в память			
22	SHLD ADR	3	5	16	(ADR+1),(ADR)<--HL	- - - - -
			КОМАНДЫ ВВОДА/ВЫВОДА			
DB	IN PORT	2	3	10	A <-- (PORT)	- - - - -
D3	OUT PORT	2	3	10	(PORT) <-- A	- - - - -
			ПРОЧИЕ КОМАНДЫ			
27	DAA	1	1	4	десятичная коррек-ция аккумулятора	
2F	CMA	1	1	4	A <--инверсия A	
37	STC	1	1	4	CY <-- 1	
3F	CMC	1	1	4	CY <--инверсия CY	
76	HLT	1	1	7	останов	
00	NOP	1	1	4	нет операции	
			КОМАНДЫ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ Сложение однобайтных чисел			
80	ADD B	1	1	4	A <-- A + B	+ + + + +
81	ADD C	1	1	4	A <-- A + C	+ + + + +
82	ADD D	1	1	4	A <-- A + D	+ + + + +
83	ADD E	1	1	4	A <-- A + E	+ + + + +
84	ADD H	1	1	4	A <-- A + H	+ + + + +
85	ADD L	1	1	4	A <-- A + L	+ + + + +
86	ADD M	1	2	7	A <-- A + M	+ + + + +
87	ADD A	1	1	4	A <-- A + A	+ + + + +

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z P A C
C6	ADI D8	2	2	7	A <-- A + D8	+ + + + +
88	ADC B	1	1	4	A <-- A + B + CY	+ + + + +
89	ADC C	1	1	4	A <-- A + C + CY	+ + + + +
8A	ADC D	1	1	4	A <-- A + D + CY	+ + + + +
8B	ADC E	1	1	4	A <-- A + E + CY	+ + + + +
8C	ADC H	1	1	4	A <-- A + H + CY	+ + + + +
8D	ADC L	1	1	4	A <-- A + L + CY	+ + + + +
8E	ADC M	1	2	7	A <-- A + M + CY	+ + + + +
8F	ADC A	1	1	4	A <-- A + A + CY	+ + + + +
CE	ACI D8	2	2	7	A <-- A + D8+ CY	+ + + + +
	Вычитание однобайтных чисел					
90	SUB B	1	1	4	A <-- A - B	+ + + + +
91	SUB C	1	1	4	A <-- A - C	+ + + + +
92	SUB D	1	1	4	A <-- A - D	+ + + + +
93	SUB E	1	1	4	A <-- A - E	+ + + + +
94	SUB H	1	1	4	A <-- A - H	+ + + + +
95	SUB L	1	1	4	A <-- A - L	+ + + + +
96	SUB M	1	2	7	A <-- A - (M)	+ + + + +
97	SUB A	1	1	4	A <-- A - A; A=0	+ 1 + + +
D6	SUI D8	2	2	7	A <-- A - D8	+ + + + +
98	SBB B	1	1	4	A <-- A - B - CY	+ + + + +
99	SBB C	1	1	4	A <-- A - C - CY	+ + + + +
9A	SBB D	1	1	4	A <-- A - D - CY	+ + + + +
9B	SBB E	1	1	4	A <-- A - E - CY	+ + + + +
9C	SBB H	1	1	4	A <-- A - H - CY	+ + + + +
9D	SBB L	1	1	4	A <-- A - L - CY	+ + + + +
9E	SBB M	1	2	7	A <-- A -(M)- CY	+ + + + +
9F	SBB A	1	1	4	A <-- A - A - CY	+ + + + +
DE	SBI D8	2	2	7	A <-- A - D8- CY	+ + + + +
	Сравнение однобайтных чисел					
B8	CMP B	1	1	4	A - B -/->	+ + + + +
B9	CMP C	1	1	4	A - C -/->	+ + + + +
BA	CMP D	1	1	4	A - D -/->	+ + + + +
BB	CMP E	1	1	4	A - E -/->	+ + + + +
BC	CMP H	1	1	4	A - H -/->	+ + + + +
BD	CMP L	1	1	4	A - L -/->	+ + + + +
BE	CMP M	1	2	7	A - (M)-/->	+ + + + +
BF	CMP A	1	1	4	A - A -/->	+ + + + +

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
FE	CPI D8	2	2	7	A - D8 -/->	+ + + + +
			Декремент однобайтных чисел			
05	DCR B	1	1	5	B <-- B - 1	+ + + + -
0D	DCR C	1	1	5	C <-- C - 1	+ + + + -
15	DCR D	1	1	5	D <-- D - 1	+ + + + -
1D	DCR E	1	1	5	E <-- E - 1	+ + + + -
25	DCR H	1	1	5	H <-- H - 1	+ + + + -
2D	DCR L	1	1	5	L <-- L - 1	+ + + + -
35	DCR M	1	3	10	(M)<--(M)- 1	+ + + + -
3D	DCR A	1	1	5	A <-- A - 1	+ + + + -
			Инкремент однобайтных чисел			
04	INR B	1	1	5	B <-- B + 1	+ + + + -
0C	INR C	1	1	5	C <-- C + 1	+ + + + -
14	INR D	1	1	5	D <-- D + 1	+ + + + -
1C	INR E	1	1	5	E <-- E + 1	+ + + + -
24	INR H	1	1	5	H <-- H + 1	+ + + + -
2C	INR L	1	1	5	L <-- L + 1	+ + + + -
34	INR M	1	3	10	(M)<--(M)+ 1	+ + + + -
3C	INR A	1	1	5	A <-- A + 1	+ + + + -
			Сложение двухбайтных чисел			
09	DAD B	1	3	10	HL <-- HL + BC	- - - - +
19	DAD D	1	3	10	HL <-- HL + DE	- - - - +
29	DAD H	1	3	10	HL <-- HL + HL	- - - - +
39	DAD SP	1	3	10	HL <-- HL + SP	- - - - +
			Декремент двухбайтных чисел			
0B	DCX B	1	1	5	BC <-- BC - 1	- - - - -
1B	DCX D	1	1	5	DE <-- DE - 1	- - - - -
2B	DCX H	1	1	5	HL <-- HL - 1	- - - - -
3B	DCX SP	1	1	5	SP <-- SP - 1	- - - - -
			Инкремент двухбайтных чисел			
03	INX B	1	1	5	BC <-- BC + 1	- - - - -
13	INX D	1	1	5	DE <-- DE + 1	- - - - -
23	INX H	1	1	5	HL <-- HL + 1	- - - - -
33	INX SP	1	1	5	SP <-- SP + 1	- - - - -

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
			ЛОГИЧЕСКИЕ ОПЕРАЦИИ Операция И			
A0	ANA B	1	1	4	A <-- A & B	+ + x + 0
A1	ANA C	1	1	4	A <-- A & C	+ + x + 0
A2	ANA D	1	1	4	A <-- A & D	+ + x + 0
A3	ANA E	1	1	4	A <-- A & E	+ + x + 0
A4	ANA H	1	1	4	A <-- A & H	+ + x + 0
A5	ANA L	1	1	4	A <-- A & L	+ + x + 0
A6	ANA M	1	2	7	A <-- A & M	+ + x + 0
A7	ANA A	1	1	4	A <-- A & A	+ + x + 0
E6	ANI D8	2	2	7	A <-- A & D8	+ + x + 0
			Операция ИЛИ			
B0	ORA B	1	1	4	A <-- A or B	+ + 0 + 0
B1	ORA C	1	1	4	A <-- A or C	+ + 0 + 0
B2	ORA D	1	1	4	A <-- A or D	+ + 0 + 0
B3	ORA E	1	1	4	A <-- A or E	+ + 0 + 0
B4	ORA H	1	1	4	A <-- A or H	+ + 0 + 0
B5	ORA L	1	1	4	A <-- A or L	+ + 0 + 0
B6	ORA M	1	2	7	A <-- A or (M)	+ + 0 + 0
B7	ORA A	1	1	4	A <-- A or A	+ + 0 + 0
F6	ORI D8	2	2	7	A <-- A or D8	+ + 0 + 0
			Операция ИСКЛЮЧАЮЩЕЕ ИЛИ			
A8	XRA B	1	1	4	A <-- A xor B	+ + 0 + 0
A9	XRA C	1	1	4	A <-- A xor C	+ + 0 + 0
AA	XRA D	1	1	4	A <-- A xor D	+ + 0 + 0
AB	XRA E	1	1	4	A <-- A xor E	+ + 0 + 0
AC	XRA H	1	1	4	A <-- A xor H	+ + 0 + 0
AD	XRA L	1	1	4	A <-- A xor L	+ + 0 + 0
AE	XRA M	1	2	7	A <-- A xor (M)	+ + 0 + 0
AF	XRA A	1	1	4	A <-- A xor A	+ + 0 + 0
EE	XRI D8	2	2	7	A <-- A xor D8	+ + 0 + 0
			Операции сдвигов			
17	RAL	1	1	4	Арифм.сдвиг влево Ai+1 <-- Ai, A0 <-- C, C <-- A7	- - - - +
1F	RAR	1	1	4	Арифм.сдвиг вправо Ai <-- Ai+1, A7 <-- C, C <-- A0	- - - - +

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
07	RLC	1	1	4	Цикл. сдвиг влево Ai+1 <-- Ai, A0 <-- A7, C<--A7	- - - - +
0F	RRC	1	1	4	Цикл. сдвиг вправо Ai <-- Ai+1, A7 <-- A0, C<--A0	- - - - +
	КОМАНДЫ ПЕРЕХОДОВ Безусловный переход					
C3	JMP ADR	3	5	17	(SP) <-- PC, SP = SP - 2, PC <-- ADR	- - - - -
	Условные переходы					
DA	JC ADR	3	3/5	11/17	Если CY=1, то <-- ADR, иначе PC = PC+3	- - - - -
D2	JNC ADR	3	3/5	11/17	Если CY=0, то ...	- - - - -
CA	JZ ADR	3	3/5	11/17	Если Z=1, то ...	- - - - -
C2	JNZ ADR	3	3/5	11/17	Если Z=0, то ...	- - - - -
FA	JM ADR	3	3/5	11/17	Если S=1, то ...	- - - - -
F2	JP ADR	3	3/5	11/17	Если S=0, то ...	- - - - -
EA	JPO ADR	3	3/5	11/17	Если P=1, то ...	- - - - -
E2	JPE ADR	3	3/5	11/17	Если P=0, то ...	- - - - -
	КОМАНДЫ ОБРАЩЕНИЯ К ПОДПРОГРАММАМ Безусловный вызов подпрограмм					
CD	CALL ADR	3	5	17	(SP) <-- PC, SP = SP - 2, PC <-- ADR	- - - - -
	Условные вызовы подпрограмм					
DD	CC ADR	3	3/5	11/17	Если CY=1, то (SP)(SP+1)<--PC SP = SP - 2, PC <-- ADR, иначе PC = PC+3	- - - - -
D4	CNC ADR	3	3/5	11/17	Если CY=0, то ...	- - - - -
CC	CZ ADR	3	3/5	11/17	Если Z=1, то ...	- - - - -
C4	CNZ ADR	3	3/5	11/17	Если Z=0, то ...	- - - - -
F4	CM ADR	3	3/5	11/17	Если S=1, то ...	- - - - -
FC	CP ADR	3	3/5	11/17	Если S=0, то ...	- - - - -
EC	CPO ADR	3	3/5	11/17	Если P=1, то ...	- - - - -
E4	CPE ADR	3	3/5	11/17	Если P=0, то ...	- - - - -

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
	КОМАНДЫ ВОЗВРАТА ИЗ ПОДПРОГРАММ Безусловный возврат					
C9	RET	1	3	10	PC<-- (SP)(SP+1), SP = SP + 2	- - - - -
	Условные возвраты					
D8	RC	1	1/3	5/11	Если CY=1, то PC<--(SP)(SP+1) SP = SP + 2; иначе PC = PC + 1	- - - - -
D0	RNC	1	1/3	5/11	Если CY=0, то ...	- - - - -
C8	RZ	1	1/3	5/11	Если Z=1, то ...	- - - - -
C0	RNZ	1	1/3	5/11	Если Z=0, то ...	- - - - -
F0	RM	1	1/3	5/11	Если S=1, то ...	- - - - -
F8	RP	1	1/3	5/11	Если S=0, то ...	- - - - -
E8	RPE	1	1/3	5/11	Если P=1, то ...	- - - - -
E0	RPO	1	1/3	5/11	Если P=0, то ...	- - - - -
	КОМАНДЫ ОБРАЩЕНИЯ К СТЕКУ Чтение из стека					
C1	POP B	1	3	10	BC <-- (SP+1)(SP), SP = SP + 2	- - - - -
D1	POP D	1	3	10	DE <-- (SP+1)(SP), SP = SP + 2	- - - - -
E1	POP H	1	3	10	HL <-- (SP+1)(SP), SP = SP + 2	- - - - -
F1	POP PSW	1	3	10	F <--(SP), A <--(SP+1), SP = SP + 2	из стека
	Запись в стек					
C5	PUSH B	1	3	10	(SP-1)(SP-2)<--BC, SP = SP - 2	- - - - -
D5	PUSH D	1	3	10	(SP-1)(SP-2)<--DE, SP = SP - 2	- - - - -
E5	PUSH H	1	3	10	(SP-1)(SP-2)<--HL, SP = SP - 2	- - - - -
F5	PUSH PSW	1	3	10	(SP-1) <-- A, (SP-2) <-- F, SP = SP - 2	- - - - -

Код	Мнемоника	Число байтов	Число циклов	Число тактов	Описание операции	Признаки S Z A P C
			КОМАНДЫ ПРЕРЫВАНИЙ			
C7	RST0	1	3	11	(SP-1)(SP-2)<-- PC, SP = SP -2, PC <-- 0000h	- - - - -
D7	RST1	1	3	11	... PC <-- 0008h	- - - - -
DF	RST2	1	3	11	... PC <-- 0010h	- - - - -
E7	RST3	1	3	11	... PC <-- 0018h	- - - - -
EF	RST4	1	3	11	... PC <-- 0020h	- - - - -
CF	RST5	1	3	11	... PC <-- 0028h	- - - - -
F7	RST6	1	3	11	... PC <-- 0030h	- - - - -
FF	RST7	1	3	11	... PC <-- 0038h	- - - - -
F3	DI	1	1	4	запретить прерывания	- - - - -
FB	EI	1	1	4	разрешить прерывания	- - - - -

ПРИЛОЖЕНИЕ 2

Таблица ASCII-кодов

Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII
0	NUL	20		40	@	60	`
01	^A	21	!	41	A	61	А
02	^B	22	"	42	B	62	Б
03	^C	23	#	43	C	63	Ц
04	^D	24	$	44	D	64	Д
05	^E	25	%	45	E	65	Е
06	^F	26	&	46	F	66	Ф
07	BEL	27	'	47	G	67	Г
08	BS	28	(	48	H	68	Х
09	TAB	29	)	49	I	69	И
0A	LF	2A	*	4A	J	6A	Й
0B	^K	2B	+	4B	K	6B	К
0C	^L	2C	,	4C	L	6C	Л
0D	CR	2D	-	4D	M	6D	М
0E	^N	2E	.	4E	N	6E	Н
0F	^O	2F	/	4F	O	6F	О
10	^P	30	0	50	P	70	П
11	^Q	31	1	51	Q	71	Я
12	^R	32	2	52	R	72	Р
13	^S	33	3	53	S	73	С
14	^T	34	4	54	T	74	Т
15	^U	35	5	55	U	75	У
16	^V	36	6	56	V	76	Ф
17	^W	37	7	57	W	77	В
18	^X	38	8	58	X	78	Ь
19	^Y	39	9	59	Y	79	Ы
1A	^Z	3A	:	5A	Z	7A	З
1B	ESC	3B	;	5B	[	7B	Ш
1C	^\	3C	<	5C	\	7C	Э
1D	^]	3D	=	5D	]	7D	Щ
1E	^^	3E	>	5E	^	7E	Ю
1F	^_	3F	?	5F	_	7F	DEL