Московский Институт Электроники и Математики

**(технический университет)**

**Кафедра ИТАС**

**РЕФЕРАТ**

**по курсу : “ЭВМ и периферийные устройства”**

**на тему: Микропроцессор Z80 его структура и система команд.**

**Выполнил: студент группы АП-41**

**Завалишин Е. А.**

**МОСКВА 1998**

**Архитектура микропроцессора Z-80**

Архитектура микропроцессора Z-80 фирмы ZILOG основывается на архитектурных принципах микропроцессора 8080 и позволяет выполнять все 78 команд этого микропроцессора, а также 80 дополнительных команд. Всего микропроцессор Z-80 имеет 696 кодов операций (в отличие от 244 кодов микропроцессора 8080).

К числу особенностей микропроцессора Z-80 относятся: использование для питания лишь одного источника напряжением “+5В”, наличие однофазного внешнего синхрогенератора, 17 внутренних регистров и встроенная схема регенерации ОЗУ.

НАЗНАЧЕНИЕ ВЫВОДОВ.

Микропроцессор Z-80 - это микросхема с 40 выводами, пронумерованными от 1 до 40. Рассмотрим функции выводов.

Вывод 11 напряжение питания +5в. Вывод 29- общий. Вывод 06- тактовый вход. В SPECTRUMe используется тактовая частота 3.5 Мгц. Вывод 07-10, 12-15- информационная шина. Вывод 1-5,30-40- адресная шина.

Оставшиеся 13 выводов присоединены к линиям, которые несут управляющие сигналы. Вывод 21- линия считывания RD. Эта линия становится активной, когда байт информации должен быть считан из памяти или порта. Вывод 22- линия записи WR. Эта линия активна, когда байт информации должен быть записан в память или в порт. Вывод 19- запрос памяти MREQ. Эта линия активна в тех случаях, когда требуется обращение к памяти.

Байт информации считывается из памяти в соответствии с адресом, помещенным на адресной шине. Далее, в соответствии с откликом на сигналы RD и MREQ, байт информации поступает на информационную шину, с которой эта информация в дальнейшем считывается микропроцессором. Для записи байта данных в память микропроцессор помещает требуемые адреса на адресную шину и требуемую информацию на шину данных. Сигналы MREQ и WR активизируются, и байт данных записывается в память.

Вывод 28- линия регенерации RFSH. Она используется для регенерации динамической памяти. Частично используется для формирования TV сканирующих сигналов. Вывод 27- активизируется при выполнении машинного цикла М1 и показывает, что проходящий машинный цикл обработки команды находится в состоянии “ввода кода операции” при выполнении некоторой команды. Сигнал М1 при выполнении двухбайтовой команды формируется при вызове каждого байта кода операции. Сигнал M1 появляется вместе с сигналом IORQ в цикле приема прерывания. Выборка инструкции требует, чтобы все три сигнала MI, MREQ и RD были активизированы. В то же время выборка байта данных из ячейки памяти требует, чтобы только MREQ и RD были активизированы. Время необходимое для выборки инструкции - 1.14мкс, что составляет 4 такта. Вывод 20- линия выход lORQ. Эта линия активна при выполнении команд IN или OUT. Вывод 18- останов HALT. Линия активизируется при выполнении команды HALT. Вывод 25- линия запроса BUSRQ. Z-80 позволяет внешним устройствам использовать адресную и информационную шину в режиме пропуска цикла. Запрос микропроцессору пропустить следующий цикл выполняется внешними устройствами путем активизации этой линии. Вывод 23- линия подтверждения, BUSAK. Микропроцессор подтверждает запрос остановки после выполнения команды и активизирует эту линию.

Оставшиеся 4 вывода находятся под контролем пользователя. Вывод 26- линия сброса, RESET. Используется для инициализации микропроцессора. Она активизируется при включении питания. Сброс может быть осуществлен в SPECTRUMe соединением линий RESET и GND. Вывод 24- линия ожидания WAIT. “Медленная” память может требовать большего времени для цикла считывания или записи и об этом сообщает микропроцессору путем активизации линии WAIT. Вывод 17- “немаскируемое прерывание” NMI. Активизация этой линии приводит к остановке выполнения микропроцессором текущей программы, и вместо нее микропроцессор выполняет программу прерывания, записанную специально для этой цели. В SPECTRUMe немаскируемое прерывание требует системного сброса, который выполняется записыванием 0 по адресу 23728. Вывод 16- “маскируемое прерывание”, INT. В SPECTRUMe сканирование клавиатуры и обмен в режиме реального времени называется “управляемым прерыванием”. Это означает, что электроника системы каждые 1/50 секунды активизирует INT, вызывая остановку выполнения микропроцессором основной программы и, вместо этого, выполнение программы сканирования клавиатуры. Способность Z80 реагировать на INT может управляться программистом специальными машинными командами.

ЛОГИЧЕСКАЯ ОРГАНИЗАЦИЯ Z80

В состав Z80 входят: устройство управления, регистр команд, программный счетчик, 24 регистра пользователя и арифметико-логическое устройство.

Устройство управления.

В Z80 управляющее устройство выдает огромное число внутренних управляющих сигналов, а также обеспечивает формирование внешних управляющих сигналов.

*Регистр команд.*

Термин “регистр” используется для описания простой ячейки памяти внутри Z80. Регистр содержит 8 бит (1 байт). В Z80 имеется большой блок регистров, и пересылка байтов данных к регистрам и от них является простейшей и наиболее важной чертой программирования в машинных кодах.

Регистр команд - специальный регистр, где микропроцессор содержит копию выполняемой текущей команды. Одной из черт набора команд Z80 является то, что определенные инструкции содержатся в двух байтах данных. В этих случаях регистр команд содержит каждую команду по очереди.

Программный счетчик.

Программный счетчик - это пара регистров, которые используются совместно, поэтому программный счетчик содержит 16-ти битные значения.

Когда инструкции выбираются, управляющее устройство использует информацию в программном счетчике как адрес ячейки памяти, содержащей команду, которая должна выполняться следующей. После выполнения команды значение в программном счетчике увеличивается.

Действие программного счетчика очень похоже на переменную РРС интерпретатора бэйсик, которая содержит номер текущей строки бэйсик и также наращивается.

*Регистры пользователя (основные регистры).*

Имеется 24 регистра пользователя. Все регистры однобайтовые, хотя обычно используются парами. Регистр А называется аккумулятором. Аккумулятор - основной регистр микропроцессора при различных операциях с данными. Большинство арифметических и логических операций осуществляется путем использования АЛУ и аккумулятора. Любая такая операция над двумя словами данных предполагает размещение одного из них в аккумуляторе, а другого - в памяти или еще в каком-нибудь регистре. Так при сложении двух слов, условно называемых А и В и расположенных в аккумуляторе и памяти соответственно, результирующая сумма загружается в аккумулятор, замещая слово А.

Регистр F - это флаговый регистр. Он часто рассматривается как набор восьми флаговых битов, связанных вместе, а не как отдельный регистр. Установление флага соответствует 1, сброс -0. Программист обычно имеет дело с 4-мя основными флагами. Это флаг нуля, флаг знака, флаг переноса и флаг четности-переполнения. Дополнительные флаги используются управляющим устройством и не могут быть использованы программистом непосредственно.

Регистровая пара HL.

При обращении к регистровой паре HL сначала указывается младшая часть (LOW), а затем старшая (HIGH). Память 64К может быть рассмотрена как 256 страниц по 256 адресов в каждой. В этом случае значение старшего байта указывает на используемую страницу.

В микропроцессоре регистровая пара HL является одной из трех регистровых пар, которые используется в качестве адресных регистров. В этом качестве регистровая пара HL является наиболее важной. Регистровая пара HL также может быть использована для хранения 16-ти разрядного числа, и существует определенный ряд арифметических операций, которые могут быть выполнены с этими числами. Н-регистр и L-регистр также могут быть использованы и как отдельные регистры, хотя с ними может быть выполнено ограниченное число операций.

Регистровая пара ВС и DE. Эти пары используются главным образом как адресные регистры. Регистры могут быть использованы как одинарные. Регистр В рекомендуется использовать в качестве счетчиков циклов.

Набор альтернативных регистров.

Z80 имеет альтернативный набор регистров для А,Р,Н,1ДС, D,E. Они обозначаются A’,F, и т.д. Существуют две специальные команды, которые позволяют обменивать основной и альтернативный набор регистров. После обмена для Z80 альтернативный набор становится основным, а основной - альтернативным.

Альтернативные наборы регистров часто используются для сохранения среда, когда запускается независимая задача.

Регистровые пары IX и IY используются для выполнения операции, которая включает индексацию. Это дает возможность работать со списком или таблицей. Начальный адрес списка или таблицы должен быть первоначально занесен в подходящую пару регистров IX иди IY. В программе монитора SPECTRUMa lY пара содержит адрес 23610 (5C3Ah), что является начальным адресом таблицы системных переменных. IX пара широко используется как указатель в программах обработки команд LOAD, SAVE, VERIFY, MERGE.

Указатель стека - это адресный регистр. Он используется для указания в памяти области машинного стека и всегда рассматривается как одинарный двухбайтовый регистр. Z80 использует стек, заполненный в памяти сверху вниз. Аналогией является магазин автомата - по принципу последний пришел, первым ушел. Указатель стека используется для указания различных размещений в области стека в каждом случае. Указатель стека всегда содержит адрес, куда последний раз была произведена запись. Поэтому управляющее устройство сначала уменьшает значение указателя стека, а затем помещает туда значение. Пересылка в стек двухбайтовая, и поэтому указатель стека должен быть дважды уменьшен при помещении в стек и дважды увеличен при выборке из стека. Машинный стек обычно используется как место сохранения адресов возврата, но можно использовать его как рабочую область.

Регистр 1 - регистр вектора прерываний, используется для размещения адресов устройств ввода-вывода. Однако BSPECTRUMe эта возможность не используется и 1- регистр используется для генерации TV - сигнала.

Регистр R- регенерации памяти. Он является простым счетчиком, который увеличивается каждый раз при выполнении цикла регенерации. 3начение в регистре циклически изменяется от 0 до 255.

*Арифметико-логическое устройство (АЛУ).*

В АЛУ выполняются арифметические и логические операции. Возможно выполнение сложения и вычитания, а также АЛУ способно выполнять большое количество битовых операций и устанавливать флаги, чтобы показать результат.

**СИСТЕМА КОМАНД МИКРОПРОЦЕССОРА Z-80.**

Команды и данные.

Команды поделены на 18 групп, каждая из которых состоит из тех команд, которые сильно похожи друг на друга. Существует шесть классов данных, которые могут следовать за командой.

1.0днобайтовая константа (+DD).

Это число А в диапазоне 00-FFh,(0-65535d). Те команды, которые требуют за собой однобайтовую константу, имеют мнемоническую приставку +DD. Например: LD HL,+DDDD.

2. Двухбайтовая константа (+DDDD).

Это число А в диапазоне 0000-FFFFh (0-65535d).Те команды, которые требуют за собой двухбайтовую константу, имеют мнемоническую приставку +DDDD. Например: LD HL,+DEDD.

3. Двухбайтовый адрес (ADDV).

Это число А в диапазоне 0000-FFFFh (0-65535d),то есть число, которое используется как адрес памяти. Те команды, которые требуют за собой двухбайтовый адрес, имеют приставку ADDV, например: JP ADDV.

4. Однобайтовая константа смещения(e).

Это число А в диапазоне 00-FFh(-128+127d).Число вегда представлено в дополнительном коде.Те команды,которые требуют за собой однобайтовую константу,имеют мнемоническую приставку e.Наприме:JP e.

5. Однобайтовая индуксирующая константа смещения (+D).

Это число А в диапазоне 00-FFh, (-128 +127d) представлено дополнительной арифметике. Те команды, которые требуют за собой однобайтовую индексирующую константу смещения, имеют мнемоническую приставку +D. Например: LD A,(JX+D)

**6 Однобайтовая индексирующая константа смещения и однобайтовая константа (+D,+DD).**

Это два числа в диапазоне 00-FFh, первое из которых рассматривается как десятичное -128+127,а второе - как десятичное 0-255. Команды, требующие два байта данных, сопровождаются мнемониками D и +DD.Например:LD (JX+D),+DD.

*ГРУППА КОМАНД.*

Существует много путей для разделения на группы сотен различных команд. Метод ,выбранный, здесь, разделяет команды на 18 функциональных групп.

Группа 1. Команда “нет операции”

Мнемоника 16-ный код

NOP 00

Выполнение команды требует 1,14 мкс. Ни один из регистров или флагов не изменяется. Команда NOP используется программистом для организации задержек, но чаше для удаления ненужных команд из программы.

Группа 2. Команды загрузки регистра константами.

Мнемоника 16-ный код

LD A,+DD 3EDD

LD H,+DD 26DD

LD L,+DD 2EDD

LD B,+DD 06DD

LD C,+DD 0EDD

LD D,+DD 16DD

LD E,+DD lEDD

Каждя из этих команд требует два байта памяти: один для кода операции, второй для константы. Команды записывают в регистр ответствующие значения, старые значения регистра пропадают.

Сведущие команды выполняют загрузку пары регистров двухбайтовыми константами.

Мнемоника 16-ный код

LD HL,+DDDD 21 DD DD

LD BC,+DDDD 01 DD DD

LD DE,+DDDD 11 DD DD

LD IX,+DDDD DD 21 DD DD

LD IY,+DDDD FD 21 DD DD

LD SP,+DDDD 31 DD DD

Строка команды требует 3 или 4 байта в памяти. Код операции занимает 1 или 2 байта ,и 2 байта занимает константа. Первый байт константы загружается в младший регистровой пары т.е. L,C,E,X,Y,P, а второй байт в старший регистр, т.е. H,B,D,l,S. Эти команды записывают в регистровые пары данные, которые часто рассматриваются программистом как двухбайтовый адрес, но могут быть также и двухбайтовым числовым значением и двумя отдельными однобайтовыми числовыми значениями.

Команды этой группы не изменяют флагов.

**Группа 3. Команды копирования регистров и обмена.** Существует 59 команд, которые выполняют копирование регистров и регистровых пар .Эти команды можно разделить на 4 подгруппы.

**Подгруппа А.** Команды копирования типа регистр-регистр. Следующая таблица дает коды операций команд, выполняющих копирование содержимого одного регистра в другой.

LD LD LD LD LD LD LD

регистр A,R H,R L,R B,R C,R D,R E,R

A 7F 67 6F 47 4F 57 5F

H 7C 64 6C 44 4C 54 5C

L 7D 65 6D 45 4D 55 5D

B 78 60 68 40 48 50 58

C 79 61 69 41 49 51 59

D 7A 62 6А 42 4A 52 5A

E 7B 63 6B 43 4B 53 5B

Ни одна из команд, приведенных в таблице, не изменяет флаги. Существует кроме того 4 команды для I и R регистров.

Мнемоника 16-ный код

LD A,I ED57

LD A,R ED5F

LD I,A ED47

LD R,A ED4F

Эти последние команды влияют на флаг переполнения четности.

**Подгруппа В. Команды копирования типа регистровая пара- регистровая пара**

Существует только 3 команды этой подгруппы. Они копируют значения в указатель стека.

Мнемоника 16-ный код

LD SP,HL F9

LD SP,IX DD F9

LD SP,IY FDF9

Эти команды не изменяют содержимое флагов. Заметим, если содержимое регистровой пары надо копировать в другую регистровую пару и эти команды не подходят, необходимо выполнить две команды копирования регистр-регистр. Например, нет команды LD HL,DE, и её заменяют, используя LD H,D и LD L,E. Или содержимое первой регистровой пары может быть сохранено в стеке и затем переписано во вторую регистровую пару.

**Подгруппа С.** Команды DE,NL.

Существует только одна команда, которая позволяет обменивать содержимое регистровых пар в пределах основного набора регистров.

Мнемоника 16-ный код

EXX DE,HL EB

Это очень полезная команда позволяет обменивать содержимое пары DE с содержимым пары HL. Значение флагов не изменяется. Команда используется, когда необходимо, чтобы адрес или двухбайтовая числовая константа из пары DE была записана в пару HK, но содержимое последней не было потеряно.

**Подгруппа D.** Команды альтернативного набора регистров.

В этой группе 2 команды.

Мнемоника 16-ный код

EXX D9

EX AF,A’F 08

Команда EXX вызывает переключение регистров H,L,B,C,D,E на регистры H’,L’,B’,C’,D’,E’. Команды EXAF,A’F, как следует из мнемоники, переключают рег. A и F на A’ F. Альтернативные регистры часто используются для хранения адресов и данных. Помещенные в альтернативные регистры эти значения сохраняются от искажения и могут быть легко и просто восстановлены.

Группа 4.Команды загрузки регистров из памяти.

В систему Z80 входит много команд, позволяющих находить данные в памяти и затем загружать их в регистры. Все эти команды требуют, чтобы программист указал адрес или пару адресов в памяти, откуда данные должны быть скопированы в регистр-получатель данных.

Команды этой группы лучше рассматривать как команды трех подгрупп в соответствии с методом адресации. Виды адресации:

1. **косвенная**- двухбайтовый адрес уже размещен в адресной регистровой паре;
2. **непосредственная** -действительный адрес указан в двух байтах вслед за кодом операции;
3. **индексная**- адрес данных должен быть вычислен путем сложения значения смещения D с основным адресом, уже содержавшимся в регистровой паре IX или IY.

**Подгруппа A .**Команды, использующие непосредственную адресацию.

Мнемоника 16-ный код

LD A ,(ADDR) 3A ADDR

LD HL ,(ADDR) 2AADDR

LD BC ,(ADDR) ED 4B ADDR

LD DE ,(ADDR) ED 5B ADDR

LD D ,(IX+D) DD 5D D

LD E ,(IX+D) DD 5E D

Для команд, использующих пару IY ,надо изменить IX на IY и DD на FD. Ни одна из команд этой группы не меняет флаги.

Группа 5.Команды записи в память содержимого регистра или константы.

Обычно команды этой группы выполняют операции, противоположные тем, которые делают команды группы 4.Команды позволяют содержимое регистров переписывать в память или записывать туда константы. Эти команды лучше рассмотреть по трём группам.

**Подгруппа A.** Команды непосредственной адресации.

Мнемоника 16-ный код

LD(ADDR) ,A 32 ADDR

LD(ADDR),HL 22 ADDR (обычная форма)

ED 63 ADDR (необычная форма)

LD(ADDR),BC ED 43 ADDR

LD(ADDR),DE ED 53 ADDR

LD(ADDR),IX ЕD 22 ADDR

LD(ADDR),IY ЕD 22 ADDR

LD(ADDR),SP ЕD 73 ADDR

Приведённые команды приводят только непосредственную адресацию и важно заметить, что нет команд для записи в память констант. Если это необходимо, константа предварительно должна быть загружена в регистр A. Затем выполняется LD(ADDR),A. Команды типа LD(ADDR),HL-фактически двойные команды: LD(ADDR),L и LD(ADDR+1),H. Команды подгруппы часто используются для сохранения адресов и значений в памяти, когда эти значения используются как переменные. Например, часто используется LD(RAMTOP),HL, где RAMTOP -метка пары адресов памяти, используемых для хранения текущего значения верхней границы памяти. Выборка текущего значения верхней границы может быть позднее выполнена команда 4 группы, например:

LD HL,(RAMTOP).

**Подгруппа B**. Команды косвенной адресации.

Команды этой подгруппы позволяют копировать содержимое регистров в память, адрес которой содержится в регистровой паре HL,BC или DE.

существует также команда записи однобайтовой константы по адресу, указанному в паре HL.

Мнемоника 16-ный код

LD(HL),A 77

LD(BC),A 02

LD(DE),A 12

LD(HL),X 74

LD(HL),L 75

LD(HL),B 70

LD(HL),C 71

LD(HL),D 72

LD(HL),E 73

LD(HL),+DD 36DD

**Подгруппа C.** Команды индексной адресации.

Мнемоника 16-ный код

LD(IX+D),A DD 77 D

LD(IX+D),H DD 74 D

LD(IX+D),L DD 75 D

LD(IX+D),B DD 70 D

LD(IX+D),C DD 71 D

LD(IX+D),E DD 73 D

LD(IX+D),+DD DD 36 DDD

Для команд, использующих IY-регистровую пару, надо изменить IX на IY и DD на DF.

Группа 6.Команды сложения.

Эта группа команд выполняет арифметические действия. Команды сложения позволяют программисту прибавить( в абсолютной двоичной арифметике) заданное число к регистровой паре, регистру или индексному адресу памяти.

Команды этой группы могут быть побелены на 3 подгруппы:

1. команды ADD;
2. команды INC. Специальные случаи сложения, когда к числу прибавляется 1;
3. команда ADC. Значение флага переноса прибавляется к результату. Флаг переноса -это один из битов регистров флагов, который используется для сигнализации о том, было ли при выполнении последней арифметической операции переполнение регистра или байта памяти.

Команды ADD и ADC меняют флаг переноса, а INC-не меняет.

**Подгруппа A**. Команды ADD.

Мнемоника 16-ный код

ADD A,+DD C6 DD

ADD A,A 87

ADD A,H 84

ADD A,L 85

ADD A,B 80

ADD A,C 81

ADD A,D 82

ADD HL,HL 29

ADD HL,BC 09

ADD HL,DE 19

ADD HL,SP 39

ADD IX,IX DD 29

ADD IX,BC DD 09

ADD IX,DE DD 19

ADD A,E 83

ADD A,(HL) 86

ADD A,(IX +D) DD 86 D

Для команд использующих IY регистр, надо поменять IX на IY и DD на FD.

**Подгруппа В**. Команды INC.

Команды этой группы позволяют прибавить 1 к регистру, ячейке памяти или паре регистров. Во всех случаях флаг переноса не изменяется.

Мнемоника 16-ный код

INC A 3C

INC H 24

INC L 2C

INC B 04

INC C 0C

INC D 14

INC E 1C

INC (HL) 34

INC (IX+D) DD 34 D

INC HL 23

INC BC 03

INC DE 13

INC SP 33

INC IX DD 23

INC IY FD 23

**Подгруппа С**. Команды ADC.

Мнемоника 16-ный код

ADC A,+DD CE DD

ADC A,A 8F

ADC A,H 8C

ADC A,L 8D

ADC A,B 88

ADC A,C 89

ADC A,D 8A

ADC A,E 8B

ADC A,(HL) 8E

ADC A,(IX+D) DD 8E

ADC A,(IY+D) FD 8E

ADC HL,HL ED 6A

ADC HL,BC ED 4A

ADC HL,DE ED 5A

ADC HL,SP ED 7A

Команды этой группы позволяют сложить два числа вместе с текущим значением флага переноса. Все команды этой группы меняют флаг переноса. Он сбрасывается, если операция ADC не даёт переполнения, и устанавливается если даёт.

Группа 7.Команды вычитания.

Команды вычитания позволяют вычисть заданное число из одинарного регистра, пары регистров или индексного адреса памяти. Команды этой группы полезно разделить на три группы, каждая из которых имеет свою мнемонику:

1. команда SUB;
2. команда DEC. Специальный случай вычитания, когда из числа вычитается 1;
3. команда SBC. Значение флага переноса вычитается из результата. Все команды SUB и SBC изменяют флаг переноса в зависимости от того, требовался ли двоичный заём. Команда DEC оставляет флаг переноса без изменения.

Группа 8.Команды сравнения.

Команды этой группы используюточень часто во всех программах. Они позволяют программисту сравнить значение, находящееся в регистре A, с константой, значением в регистрах и области памяти. Команды выполняют операцию вычитание без переноса, без запоминания результата вычитания и только устанавливают флаги в регистре флагов. Первоначальное значение в регистре А не изменяется. Флаг переноса устанавливается, как при операциях вычитания. Сравнение, которое >=,сбрасывает флаг переноса, а < устанавливает. Команды этой группы -команда одинарного сравнения.

Мнемоника 16-ный код

CP +DD FE DD

CP A BF

CP H BC

CP L BD

CP B B8

CP C B9

CP D BA

CP E BB

CP (HL) BE

CP (IX) +D DD BED

CP (IY) +D FD BED

Группа 9.Команды логики.

Это команды AND, OR, XOR .Они переменяются к регистру А и другой заданной переменной. Операция выполняется побитно, и 8-битовый результат возвращается в регистр А.

**Подгруппа А. Команда AND.**

Логическая операция выполняется над двумя двоичными цифрами и результат равен 1,только если оба тестируемых биты установлены. В противном случае результирующий бит равен 0.Командой AND удобно сбрасывать биты 0-7 регистра А. Этот процесс называется демаскированием и позволяет проверять определённые биты байта данных.

Мнемоника 16-ный код

AND +DD E6

**Подгруппа B. Команда OR.**

Логическая операция “или” выполняется над двумя двоичными цифрами, и результат равен 1,если одна или обе цифры установлены. В противном случае бит результата сброшен.

Мнемоника 16-ный код

OR +DD E6 DD

Подгруппа C. Команда XOR.

При работе команды проверяются два разряда. Если хотя бы один из них, но не вместе, установлен в единицу, то результирующий бит устанавливается также в 1,иначе результирующий бит сбрасывается.

**Группа 10.Команда DJNZ.**

Единственная команда в этой группе является одной из самых полезных и самой широко используемой в Z80.Работа этой команды может быть уподоблена циклу FOR-NEXT программы бэйсик следующего вида: FOR B=X TO 0 STEP-1:NEXT B

в этом цикле переменной “B” присваивается значение X. Затем с каждым проходом цикла она уменьшается до тех пор, пока не достигнет значения 0.Команда DJNZ используется подобным же образом.

Группа 12. Команда стека.

В большинстве программ машинных кодов осуществляется активное использование машинного стека: программистом в качестве места где можно хранить данные, микропроцессором- для сохранения адресов возврата. Команда, которые образуют эту группу, можно разделить на две подгруппы для пользователя и три подгруппы для микропроцессора.

Подгруппа А.PUSH и POP

Эти команды позволяют программисту по PUSH сохранить 2 байта данных в машинном стеке, а по POP-копировать 2 байта из машинного стека. Эти 2 байта данных должны копироваться в строго определенную пару регистров, но важно знать, что не производится никакой записи, которая указывала бы, к какому из двух регистров какой байт данных принадлежит.

Когда выполняется команда PUSH, указатель стека предварительно уменьшается, чтобы указать на свободный адрес, и в стек копируется содержимое старшего регистра. Затем указатель стека уменьшается вторично, и величина из младшего регистра копируется в стек. Противоположные действия имеет место во время выполнения команды POP.

Подгруппа B. Команды CALL.

Действие команд CALL следующие:

1.Текущее значение счетчика команд, т.е. адрес первой ячейки после ADDR, сохраняется в стеке. Указатель стека изменяется как в случае команды PUSH. Старший байт счетчика команд сохраняется в стеке над младшим байтом.

2.Адрес ADDR затем записывается в счетчик команд, и выполняется программа.

3.По команде RET осуществляется возврат из программы.

Подгруппа C. Команды RET.

По команде RET адрес возврата из машинного стека побайтно восстанавливается в счетчике команд. При этом указатель стека дважды модифицируется в сторону увеличения. Важно понимать, что восстановленный адрес возврата из машинного стека не обязательно тот, что был в начале помещен туда командой CALL.

Подгруппа D. Команды RST.

Последняя подгруппа команд в этой группе содержит специальную команду RST или команду “повторный пуск”.Эти команды являются аналогичные командам CALL, но не требуют, что адрес определялся.

**Группа 13.Команды сдвига.**

Набор команд Z80 имеет большое число команд для сдвигов битов определенного байта. Эти команды очень полезны. Тем более, что все они сдвигают биты через флаг переноса. Сдвиг байтов влево удваивает величину ,записанную в этом байте (регистре),старшие биты при этом не теряются. Во время сдвига вправо величина делится пополам.

**Группа 14.Команды обработки битов.**

Набор команд Z80 имеет команды, которые позволяют определять, устанавливать и сбрасывать определенные биты в пределах байта, помещенного в регистр или находящегося в памяти.

Подгруппа А. Команды BIT.

Они позволяют определять состояние определенного бита. Команды BIT устанавливают флаг ноль, если используемый бит сброшен, и наоборот.

Подгруппа B. Команда SET.

Эти команды позволяют устанавливать определенный бит, при этом не один из флагов не изменяется.

Подгруппа C. Команда RES.

Данная команда сбрасывает определенный бит. Флаги не меняются.

**Команды подгруппы AUTOMATIC(NON-AUTOMATIC).**

LDIR

Это самая распространенная команда из этой группы. Команда будет перемещать данные, адрес источника которых содержится в регистровой паре HL, в область памяти с адреса, находящегося в регистровой паре DE. Число перемещаемых байтов определяется в пара BC. При работе одиночный байт перемещается из(HL) в (DE). Величина в регистровой паре BC затем уменьшается, а величина в HL и DE увеличивается.

LDDR

Эта команда является такой же как команда LDIR, за исключением того, что после пересылки каждого байта значения в HL и DE уменьшаются. Поэтому команда требует, что бы базовый адрес блока соответствовал последнему адресу блока. Место назначения информации также должно относится к последнему адресу области памяти.

CPIR

Эта команда просматривает определенную область памяти для поиска первого наличия эталонной величины. Регистровая пара HL должна содержать базовый адрес, BC-число байтов для исследования, а регистр А эталонную величину. При работе байт по адресу HL сравнивается с байтом , находящемся в А. Если они не совпали, то BC уменьшается, а HL увеличивается, и сравниваются следующие байты. Так продолжается до тех пор, пока не совпадают байты, либо BC не станет равным 0.

*Неавтоматические команды.*

LDI

Выполнение этой команды даст перемещение одиночного байта данных из адреса записанного в регистровой паре HL, по адресу записанного в регистровой паре DE. Величина в BC уменьшается.

LDD

Как LDI за исключением того, что величина в HL и DE уменьшаются.

CPI

Выполнение этой команды заключается в сравнении байта по адресу, определенному в паре HL, с байтом в регистре А. Флаг 0 устанавливается, если они совпали, в противном случае сбрасывается.

**Группа 16. Команды ввода \ вывода.**

В Z80 имеется исчерпывающий набор команд, позволяющих программиступолучать данные от внешнего источника (IN) или выдавать данные на периферию (OUT). Имеются простые, неавтоматические и автоматические команды в этой группе. Во всех случаях данные, которые передаются через IN и OUT определяют форму 8-ми битовых данных. По команде IN Z80 принимает байты данных с шины данных и копирует их в заданный регистр. Во время выполнения команды IN линия IORQ активна, так же как и RD. По команде OUT Z80 передает данные с заданного регистра на шину данных, оттуда они поступают на периферийное устройство. Во время работы команды OUT линия IORQ и WR будут активны. В дополнение к состоянию RD, WR и IORQ периферийное устройство будет активизироваться с использованием адреса, помещенного на адресной шине во время выполнения либо IN , либо OUT команды. Этот адрес указывает конкретный порт и в Z80 представляет собой 16- битный адрес.

**Группа 17. Команды прерывания.**

EI

При включении Z80 система маскированного прерывания не может прервать выполнение команд. Эта ситуация существует до тех пор, пока прерывания не будут разрешены программистом по команде EI.

DI

В любом месте программы программист может запретить прерывания по команде DI, которая не позволяет микропроцессору принимать сигналы с линии INT.

IM0

Имеется три типа прерывания. Режим 0 выбирается автоматически при включении или по команде IM0. Этот режим позволяет периферийным устройствам обмениваться информацией с микропроцессорм, программа перезапуска которого должна следовать за приемом маскированного прерывания на линии INT.

IM1

Режим прерывания 1 выполняется по команде IM1 при выполнении программы ROM монитора. В других случаях необходимо включить IM1 в начало программы. В этом режиме перезапуск будет всегда выбираться с адреса 0038h при приеме сигнала на линии INT, длительность которого соответствует возможности обработки системы маскированного прерывания.

RETI

Это специальная команда возврата для использования с программой маскированного прерывания. По этой команде происходит возврат с сохранением маскированного прерывания, принятого раньше.

RETN

Это команда подобная RETI, но она применима в конце программы немаскированного прерывания.

**Группа 18. Дополнительные команды.**

CPL

Представляет собой простую команду, которая инвертирует содержимое регистра А, т. е. она устанавливает сброшенный бит и сбрасывает установленный. Основные флаги не изменяются.

NEG

Это команда дополнения до двух содержимого регистра А, т. е. выполняется преобразование А=-А. Эта команда влияет на основные флаги. Флаги знака и 0 зависят от результата. Флаг переноса будет установлен, если регистр А первоначально был 0.

SCF

Установка флага переноса.

CCF

Инверсия флага переноса.

HALT

Это специальная команда, заставляющая мп останавливать выполнение команд, пока не возникает прерывание. Следовательно, при поступлении прерываний, команда HALT заканчивается. Команда “PAUSE” использует это, чтобы подсчитать количество 1/50 долей секунды.

**Список Литературы:**

**МАШИННЫЕ КОДЫ издательство “СОЛОН” 1993 г.**