
ПРИКЛАДНЫЕ ПРОГРАММЫ
ДЛЯ
ZX-SPECTRUM
ВЫПУСК 2
РУКОВОДСТВО ДЛЯ ПОЛЬЗОВАТЕЛЯ

НАУЧНО – ПРОИЗВОДСТВЕННЫЙ ЦЕНТР " ТИНОПОС"

Прикладные программы
для

ZX-SPECTRUM

Выпуск 2

Руководство для пользователя

НОВОПОЛОЦК 1992

О Г Л А В Л Е Н И Е

1. COPY NEW

1. CAT - клавиша ':' - считывает заголовок без сохранения в памяти.

2. LOAD - клавиша 'J'; TO - клавиша F + SS; AT - клавиша I + SS; считывание в память.

2.1. LOAD - следующий файл.

2.2. LOAD N - с N-го файла.

2.3. LOAD N TO M - с N-го файла по M-тый.

2.4. LOAD AT NN - с адреса NN в память. NN (MIN) с 27040 по 42496.

2.5. LOAD - стирает все считанные файлы с номерами -N.

2.6. LOAD (NN - считывание первых NN байтов с пропуском имени.

2.7. LOAD (NN - 10 - считывание с пропуском первых NN байтов.

2.8. LOAD N (NN - до файла с номером N.

3. SAVE - клавиша S; TO - как и выше, STEP - клавиша I+SS; запись на ленту.

3.1. SAVE - от 1 до последнего без перерыва.

3.2. SAVE N - от N без перерыва.

3.3. SAVE TO M - от 1 до M без перерыва.

3.4. SAVE STEP K - -->>-- K - пауза в секундах; для произвольная длительность.

4. VERIFY - клавиша V: верификация.

4.1. VERIFY N TO M - файлы от N до M.

5. LET - клавиша L: смена параметров заголовка.

6. LET N=имя, длина, START, PROG - новые параметры заменят старые.

6.1. LET / без параметров - распечатка каталога в памяти.

7. LIST - клавиша K - распечатка памяти.

7.1. LIST NN - показывает первые 15 байтов памяти с адреса NN в виде: адрес, значение, код значения (десятичные). ENTER - следующие 15 байтов.

8. POKE - клавиша O: изменить значение в памяти.

8.1. POKE X,NN - байт X на NN (если NN<255 или X и X+1, если NN>=256).

9. USR - клавиша U: запуск программы с адреса, заданного в команде USR NN.

10. RETURN - клавиша Y: возврат в BASIC без стирания файлов.

11. COPY - клавиша Z: копирование больших файлов.

11.1. COPY - копирование до 49096 без заголовка. Запись по нажатию SP. Можно копировать многократно. Во время считывания файла его заголовок автоматически опускается.

12. COPY NN - копирование файлов до 49152 (полная память). Запись по нажатию SP. Файл копируется один раз, перед считыванием лента устанавливается вслед за заголовком (нажать ENTER после заголовка). NN означает адрес, под которым прячется копирующая программа - 129 байтов, 16384 NN 49350.

Примечание: 1. COPY - уничтожает COPY NEW.

2. Модули COPY не будут активны, если записаны

последние 200 байтов памяти.

3. COPY приводит к зависанию системы при ошибке на ленте, а также по окончании записи.

13. Обозначение на экране типов файлов:

P - BASIC;
B - BYTES (машинные коды);
A - табличные значения;
C - таблица, ASCII.

Для файла без заголовка или при несовпадении длины, с указанной в заголовке, на экране показывается реальная длина из этого заголовка. Если обе длины совпадают - символ исчезает с экрана.

2. COPY-COPY

Руководство пользователя.

CAT - клавиша 'C' - просмотр содержимого магнитной ленты;

LOAD - клавиша 'J' - загрузка файлов в память;

LOAD X - загрузить файл в память на место X-го файла. Если X = 1, то загруженные перед этим файлы теряются и загрузка производится в начало рабочей области (с адреса 23296);

LOAD X TO XX - загружаются файлы с номерами от X до XX;

LOAD AT XX - загружать файлы с адреса XX. По умолчанию файл с номером 1 загружается по адресу 23296. Можно задать XX-23040, в этом случае величина рабочей области для загрузки файлов увеличивается до 42496 байт. По умолчанию величина этой области равна 42240 байт (очевидно, XX < 234040 задавать нельзя);

LOAD (XX - считывание первых XX байтов файла.

Пример: LOAD (6912 - считывание только экранной области.

SAVE - клавиша 'S' - сохранение загруженных файлов на м/л;

SAVE - сохранение всех загруженных файлов без пауз;

SAVE X - сохранить файлы, начиная с номера X;

SAVE X TO XX - сохранить файлы с номерами от X до XX;

SAVE TO XX - сохранить файлы с номерами от 1 до XX;

SAVE STEP X - сохранение всех загруженных файлов. Между файлами делать паузы X секунд. Если X-9, то после каждого файла выдавать запрос 'PRESS ANY KEY';

SAVE X TO XX STEP XXX - сохранение файлов с номерами от X до XX с паузами между файлами XXX секунд;

VERIFY - клавиша 'V' - проверка сохраненных файлов;

VERIFY - аналогична SAVE;

VERIFY X TO XX - аналогична SAVE X TO XX;

VERIFY X - аналогична LOAD X;

LET - клавиша 'I' - изменение полей заголовка файла. Например: LET 2-AAA,,1 - файл с номером 2 будет иметь имя AAA, стартовый адрес 1; LET 3-500,1,5 - файл с номером 3 будет иметь длину 500 байт, стартовый адрес 1, длина программы 5 байт;

LIST - клавиша 'K' - распечатка памяти;

LIST XX - задает адрес памяти. Если адрес не задан, то он равен 0. По этой команде выводится 15 байт памяти, для которых показывается: адрес памяти, десятичное значение байта, десятичное значение двух смежных байт памяти и символьное

значение байта. Для вывода следующих 15 байт нажмите 'ENTER';

POKE - клавиша '0' - изменение десятичного значения байта;

POKE X, AA - X - адрес, AA - десятичное значение байта. Если AA лежит в диапазоне 256 - 65535, то считается, что задано значение двух смежных байт;

USR - клавиша 'U' - вызвать программу пользователя;

USR X - вызвать программу пользователя по адресу X, можно использовать эту функцию для полного сброса системы, задав X=0;

RETURN - клавиша 'Y' - возврат в BASIC, инициализируются системные переменные и таблица каналов, полный сброс не выполняется;

COPY - клавиша 'Z' - копирование больших программ;

COPY - перевод программы в специальный режим копирования файла без заголовка длиной до 49096 байт. После выдачи команды программа загружает файл в память, а затем, по нажатию 'CAPS SHIFT', выгружает ее необходимое число раз. Повторная загрузка возможна только если остается не менее 200 байт незанятой памяти. После записи необходимого количества копий компьютер необходимо сбрасывать отключением от сети;

COPY NN - копирование файлов длиной до 49152 байт. Копирование выполняется только один раз.

Условные обозначения типов файлов:

P - программа;
B - машинные коды (BYTES);
A - числовой массив;
  - символьный массив;
- числовой массив;
'ENTER' - повторяет предыдущую команду;
'CAPS SHIFT + 0' - уничтожает набранную строку.

3. TFCOPY

Программа TFCOPY (TAPE FILE COPY) предназначена для копирования информации на магнитную ленту. В процессе загрузки программы (по команде LOAD "TFCOPY") на экране появляется промежуточное сообщение:

```
TAPE FILE COPY - 11/86
OPTIONS:     LOAD
(функции)    SAVE
             VERIFY
             DELETE
             CLOCK
             MODE
             RENAME
             CURSOR KEYS
PARAMS:      NUMBER
(параметры)  ALL
             BEGIN
             END
             PROGRAM (BASIC+...)
             FILE (HEADER + DATA)
```

- SOMPRESSED FORM IN RAM -
(сжатое хранение данных в памяти)

ARNOST VECERKA - OLOMOUC.F

После окончательной загрузки программы появляется стартовое меню (звездочкой отмечены мерцающие цифры - режим по умолчанию):

0 - START	- старт;
1 - CLOCK - SET	- вкл.часов;
2 - MODE - SELECT	- выбор режима;
3 - RENAME	- переименование;
4 - 41984 BYTES - 14 LINES	
5 - 44032 BYTES - 6 LINES	
6 - 44288 BYTES - ATTR USED	

До нажатия на клавишу 0 (START) можно изменить режим работы программы нажатием соответствующих клавиш (1, 3, 4 или 6). Клавиши 4, 5 и 6 выбирают режим вывода на экран каталога загруженных в память программ (6 или 14 строк), при этом изменяется объем памяти, предназначенный для загрузки программ (при выводе большего числа строк объем памяти меньше). По умолчанию выводится 6 строк каталога при 44032 байт свободной памяти.

После нажатия клавиши 0 (START) в верхней части экрана высвечивается объем свободной памяти и команды, которые может выполнять программа.

LOAD (клавиша L) - загрузить файл(ы) в память. После нажатия клавиши L необходимо включить магнитофон. На экране отображается номер загружаемого файла, количество свободной памяти после загрузки очередного файла и каталог загруженных в память программ (имя файла, тип информации, объем и адрес загрузки). Процесс загрузки можно прервать в любой момент одновременным нажатием клавиш CAPS SHIFT и BREAK SPACE. Если в память загружено больше файлов, чем может отображаться на экране, то для просмотра всего каталога следует пользоваться клавишами сдвига курсора вниз и вверх.

SAVE (клавиша S) - выгрузить файл(ы). После нажатия клавиши S необходимо уточнить с какого (FROM) по какой (TO) файлы необходимо выгрузить. Номер надо задавать двухзначный или оканчивать ввод нажатием 'ENTER'. Перед окончательным ответом следует включить магнитофон на запись.

DELETE (клавиша D) - удалить файл из каталога. После нажатия клавиши D необходимо уточнить номера удаляемых файлов.

VERIFY (клавиша V) - сравнить файл, находящийся в памяти и записанный на магнитную ленту. Перемотать ленту на начало сравниваемого файла. После нажатия клавиши V и уточнения номера включить магнитофон на воспроизведение.

MODE (клавиша M) - модификация режима вывода программы с очисткой памяти. После нажатия клавиши M появляется надпись:

CLEAR 1-2-3

Последующее нажатие клавиши 1 переводит программу в режим вывода 14 строк каталога (41984 байт памяти свободно). Клавиша 2 - 6 строк (44032 байт памяти свободно). Клавиша 3 - выводится 6 строк каталога (44288 байт памяти свободно), при этом в нижней

части экрана отображается состояние экранной памяти, записанной загружаемой программой.

RENAME (клавиша R) - переименование программ. После нажатия клавиши необходимо ввести номер файла, который требуется переименовать. Выведется старое имя, после этого следует набрать новое имя (до 10 символов).

CLOCK (клавиша C) - установка параметров ввода информации с магнитофона. После нажатия на клавишу C выведется сообщение:

WAIT:456 SAMPLE:2400 CURSN

Клавишами управления курсором необходимо установить нужные характеристики.

ALL (клавиша A) - выдает команды LOAD, SAVE, DELETE, VERIFY на все файлы, введенные в ОЗУ.

Сообщения:

При ошибках ввода с ленты появляется сообщение:

ACCEPT PARITY BYT - XXXXX

Необходимо повторить ввод.

При прекращении ввода нажатием CAPS SHIFT/BREAK SPACE появляется сообщение:

ACCEPT BREAK BYT - XXXXX.

4. TF COPY 11/86

TAPE FILE COPY - данный копировщик имеет следующие режимы работы:

0 - START - запуск программы копировщика;
1 - CLOCK SET - включение часов программы копировщика;
2 - MODE SELECT - выбор режима копирования;
3 - RENAME - изменение заголовка файла;
4 - 41984 BYTES - свободной памяти ОЗУ;
5 - 44032 BYTES - свободной памяти ОЗУ;
6 - 44288 BYTES - свободной памяти ОЗУ;
В этом копировщике используется компрессия сигнала RAM.

1. LOAD - клавиша 'L'. LOAD 1 - позволяет загрузить программу с номером 1; если ранее уже был загружен файл, то загрузка идет с очередного номера.

ПРИМЕР:

```
01    LST   4,5   BASIC        0      44
02                                     44
03    LST   4,5   CODE       25500    4000
```

где: BASIC - программа на бейсике;
CODE - файл, записанный в машинных кодах;
без наименования - файл, записанный в десятичном коде.

2. SAVE - клавиша 'S'. Позволяет сохранить файлы с номерами от 1 до 4. После этого номера записываемых файлов закрашиваются в другой цвет.

3. VERIFY - клавиша 'V'. Проверка записанных файлов (управление аналогично LOAD).

4. RENAME - клавиша 'R'. Изменение полей заголовков файлов введенных программ.

5. DELETE - клавиша 'D'. Удаляет файлы, записанные с

ошибкой или которые уже были записаны ранее.

6. ALL - клавиша 'A'. Активизирует действие команд LOAD, SAVE, VERIFY, DELETE на все файлы, введенные в ОЗУ.

7. '5' и '6' - позволяют делать просмотр в сторону уменьшения номеров заголовков файлов.

8. '7' и '8' - позволяет делать просмотр в сторону увеличения номеров заголовков файлов.

9. 'C' - CURS N - WAIT: 4160 SAMTE 65536 в первом режиме меню.

В случае ошибки вводимого файла программа TF COPY 11/86 выдает сообщение об ошибке четности или ленты и число ошибок в файлах по программе.

ВНИМАНИЕ! После нажатия последней клавиши команды, команда выполняется сразу.

5. COPY 86/M

LOAD (клавиша L) - загрузка файлов в память очередным номером. Исходной свободной памяти - 45 000 байт.

Внимание: в случае ошибки считывания с магнитной ленты в левом углу появляется вместо LOAD надпись TAPE ERROR и знак вопроса после файла, введенного с ошибкой. Нажать клавишу 'BREAK', после чего клавишу 'L' и продолжать ввод файлов дальше.

Пример:

```
TAPE ERROR  FREE:38541  TIME 1:18
     P      DAN DARE    20          164   164
     C                  65235       168   168
     H                  28723       534   534
     H                  NOT FOUND   41000 41000
```

Где: P - программа (BASIC);
C - машинные коды (BYTES);
H - неименованный файл.

Внимание: в случае появления надписи 'OUT OF MEMORY', необходимо файл, который не умещается в ОЗУ, загрузить вновь (перед этим записать предыдущие файлы на ленту и удалить их).

COPY (клавиша 'C') - перевод программы в специальный режим копирования. Против каждого копируемого файла появляется буква 'C'. После нажатия 'ENTER' появляется надпись 'START TAPE ...', после второго нажатия 'ENTER' - надпись 'COPY' и программа переходит в режим копирования.

COPY/PAUSE (клавиша 'M') - режим копирования файлов с паузой в 6 секунд между файлами. В конце каждого копируемого файла появляется надпись 'CP'. Используется в том случае, если необходимо сделать паузу между программами. Например:

```
COPY        FREE:991        TIME 7:35
P           GREEN BERET     2           175   172  C
C                           30000       6912  6912 C
C                                       41000 41000 C
P           HARD GUY        2           191   191  CP
C                           28000       6912  6912 C
```

VERIFY (клавиша 'V') - проверка скопированных файлов. После каждого проверенного файла появляется буква 'V'.

Клавиша 'X' - отмена ранее заданной операции с файлами.
Клавиша 'S' - просмотр загруженных файлов.

ALL (клавиша 'A') - применяется при операциях со всеми загруженными файлами. Например:

ALL('A') - COPY('C') - копирование всех файлов.

DELETE (клавиша 'D') - удаление ранее загруженных файлов. Напротив файла, подлежащего удалению, ставится буква 'D', а поверх букв - пунктирная линия.

Адресация (клавиша 'H') - триггерная команда. При нажатии вместо десятичного отображения свободной памяти появляется шестнадцатиричное, по второму нажатию - наоборот. При первом нажатии все загруженные файлы будут иметь указанные адреса, символьное значение, десятичное значение, обозначение в шестнадцатиричном коде.

PRINT (клавиша 'B') - распечатка загруженных файлов (BASIC).

'ENTER' - перевод в рабочий режим команд копировщика.

'BREAK' - переход из одного режима в другой.

6. COPIER FM-3

Этот копировщик имеет два режима - загрузка и копирование. После загрузки копировщик находится в режиме загрузка.

ОСНОВНЫЕ УПРАВЛЯЮЩИЕ КЛАВИШИ

клавиша	команда	производимое действие
A	AUTO	копировать все файлы от указателя
C	COPY	копировать файл под указателем
S	SKIP	передвинуть указатель
R	RESET	стереть все программы
BREAK	BREAK	включить режим копирования
CAPS+1	SOUND	вкл./выкл. звук
SYMB+4	RADIX	формат чисел 16/10
CAPS+E	EXIT	выход в BASIC

В каждом из режимов внизу экрана имеется подсказка - список команд, которые можно подавать в этом режиме. При этом клавиша, вызывающая команду, совпадает с первой буквой команды. Список программ на экране выглядит так:

```
        RADIX: HEX          SOUND: ON
T NAME                START         LENGTH        BYTES          E
P GREEN BERET         00A0          00BF          00BF
B GREEN 1             5A00          6FB0          6FB0           *
                                                  2EA0
                                    RESET/AUTO/COPY/SKIP
```

Надписи 'RADIX' и 'SOUND' показывают режимы индикации копировщика - режим шестнадцатиричных кодов и режим включенного звука.

T - тип загруженного файла (P - программа на бейсике, B - байты);

NAME - имя файла;

START - для программ на бейсике номер строки самозапуска, для байтов - адрес начала загрузки;

LENGTH - длина файла (в байтах), указанная в заголовке;

BYTES - фактическая длина файла (в байтах);

E - наличие ошибок (не сходится контрольная сумма или была нарушена структура сигнала).

В нашем примере мы видим на экране три загруженных файла - файлы 'GREEN BERET' с программой на бейсике, загруженный успешно; файл 'GREEN 1' с байтами, загруженный с ошибкой; блок данных без заголовка длиной 3EA0, загруженный успешно.

Клавишей S(SKIP) можно установить указатель (мигающую полоску) на необходимый Вам файл или блок данных, запустить магнитофон и нажать кнопку C (COPY). При этом указанный файл или блок будет записан на магнитофон, после чего указатель перенестится к следующему файлу или блоку, а если он последний, то к началу списка.

Действие клавиши A (AUTO) аналогично действию клавиши C, но при этом записываются все блоки и файлы, начиная с указанного и до конца списка.

Прекратить запись любого файла можно клавишей BREAK.

Клавиша R (RESET) сбрасывает содержимое памяти и включает режим 'загрузка'.

Предупреждение! Некоторые файлы могут не поместиться в память копировщика. Такие файлы этим копировщиком скопировать нельзя. Воспользуйтесь копировщиками COPY NEW 1, ZK COPY 87, TF COPY 86 или COPY 86/M. Некоторые файлы могут иметь нестандартную (не 1500 бод) плотность записи. Такие файлы невозможно скопировать обычными копировщиками.

7. SINCLAIR COPY

Программа SINCLAIR COPY предназначена для перезаписи программ оригиналов. После загрузки на экране появляется таблица, в которой:

1-я колонка (T) - код, обозначающий тип файла:

0 - программа на бейсике;

1 - числовой массив;

2 - текстовый массив;

3 - программа в машинных кодах.

2-я колонка (NEV) - имя файла (не более 10 знаков).

3-я колонка (HOSSZ) - длина в байтах в шестнадцатиричной системе.

4-я колонка (TCIM) - начальный адрес файла в памяти компьютера.

5-я колонка (BYTE) - фактическая длина файла в байтах в шестнадцатиричной системе.

6-я колонка (MEM) - начальный адрес записи файла в программе SINCLAIR COPY, который затем при нормальном завершении ввода файла в компьютер меняется на адрес окончания файла в памяти SINCLAIR COPY.

После появления таблицы, программа SINCLAIR COPY готова к принятию программы-оригинала. Память программы рассчитана примерно на 8000 байтов в шестнадцатиричной системе или на 4000 байтов в десятичной.

Если копируемый файл нормально вошел в память программы, начинает считываться следующий файл. Теперь, если Вы хотите приступить к перезаписи вошедших файлов, то нажмите BREAK и программа предложит Вам четыре возможных действия:

RESTART (клавиша R): - очищается память программы, после чего она готова к вводу следующей программы или файла-оригинала.

END (клавиша E):- очищается память компьютера от программы SINCLAIR COPY. Процедура аналогична нажатию кнопки RESET.

COPY (клавиша C): - приготовьте магнитофон к записи. По нажатию 'C' начинается перезапись из памяти на ленту. Запись файла сопровождается мигающей белой полосой и чередующимися желто-синими полосами на краях экрана. Уже записанный файл отмечается постоянной белой полосой.

SKIP (клавиша S): - выбор файла для перезаписи.

Если памяти программы не хватило или произошел сбой, то данный файл зачеркивается красной полосой, а нормально вошедшие файлы готовы к перезаписи (о чем говорит мигающая белая полоса над первым нормально вошедшим файлом). Невошедшие файлы попробуйте перезаписать отдельно.

8.SPECIAL COPY

После загрузки программы на экране телевизора появляется надпись: 'SPECIAL COPY'.

Поставьте кассету-оригинал, запустите ее, затем нажмите любую клавишу. Появится мигающая надпись: 'HEADER READING' - чтение заголовка и программа ждет ввода файла, после ввода заголовка файла, на экране появляется его содержимое:

тип файла: заголовок;
общая длина: в байтах;
длина программы: в байтах;
адрес начала и номер стартовой строки.

Мигает надпись 'название'. Содержимое загружается.

Далее программа вводит содержимое файла. По окончании ввода дается звуковой сигнал и на экране внизу появляется надпись: 'поставьте новую кассету, затем нажмите любую клавишу'.

При появлении этой надписи по сигналу следует остановить магнитофон, вынуть кассету-оригинал и вставить кассету для копии, включить магнитофон на запись и нажать любую клавишу. После этого появится надпись: 'название файла' записывается.

По окончании записи по звуковому сигналу остановите магнитофон. На экране появится надпись: 'хотите записать снова?'.

Для повтора записи нажмите 'Y', для записи следующего

файла – 'N' – программа перейдет на начало.

Таким образом осуществляется перезапись по одному блоку, что может быть не очень удобно. Кроме того программой не предусмотрена перезапись файлов без названия. Однако, для опытного программиста большое преимущество заключается в том, что программа написана на бейсике и ее легко модифицировать.

9. OMNI COPY-1

Копировщик программ и данных с магнитофона. Команды:
DELETE – TO RESET SYSTEM (сброс системы);
SPACE – TO STOP (останов инструкции);
стрелки – выбор файла (курсорные стрелки 'вверх', 'вниз');
CS – продвижение инструкции;
L – загрузка с магнитофона;
S – запись на магнитофон;
V – сравнение (проверка чтения);
C – TO COUNT;
B – TO BREAK (прекращение считывания/записи);
K – TO KEEP;
Q – уничтожение файла с наименьшим номером;
H – создание заголовка;
R – деактивизация строки запуска бейсик-программы;
D – двойная скорость (плотность) записи/считывания;
P – FOR LEADER PITCH;
N – FOR MAXIFILE;
M – FOR MEGAFILE.

10. MR.COPY

После загрузки программы на экране появляется мистер Копи, который приветствует Вас: 'Хелло! Я мистер Копи!'. На доске написано: 'Мистер Копи версия 1.1983. Автор Ван-Сондерен. Дублирующие программы могут быть нарушением закона об авторском праве'.

Чтобы начать работу с программой нажмите клавишу '0'. Мистер Копи говорит: 'пожалуйста выберите из меню'. На доске предоставлено 'Главное меню':

1 – загрузка оригинального файла;
2 – запись копии на кассету;
3 – верификация записанной копии;
4 – чтение заголовков файлов;
5 – загрузка и запись копии файлов без заголовков;
6 – сброс программы и выход в бейсик.

Выбор из меню производится нажатием соответствующей клавиши. Итак, по порядку:

1. М-р Копи: 'Сколько блоков данных будете вводить? (1-10)'.

На доске вверху появляется надпись 'LOADING' – загрузка. Для ввода 10 блоков нажмите '0'.

М-р Копи: 'Нажмите PLAY' (в программе допущена ошибка,

вместо 'PLAY' м-р Копи советует нажать 'REG').

Блоки: (количество оставшихся незагруженных блоков).

На доске появляется заголовок файла: тип файла (PROGRAMM, DATA ARRAY, BYTE, CHARAKTER ARRAY - программа в бейсике, числовой массив, машинные коды, текстовый массив).

Название файла (не более 10 символов).

Начальный адрес в программе или памяти.

Длина файла в байтах.

Вводится в память программы и ожидается ввод следующего блока данных.

2. М-р Копи: "Нажмите REG и PLAY, затем '0".

На доске вверху появляется надпись 'SAVING' - запись. По нажатию '0' происходит запись всех загруженных блоков последовательно, в том порядке в котором они были загружены с выдержкой стандартных пауз между блоками. По окончании записи дается короткий звуковой сигнал и программа возвращается к главному меню.

4. М-р Копи: "Нажмите PLAY блоки '...'"

На доске вверху 'READING' - чтение. Программа ждет ввода заголовка файлов. На доске появляются заголовки файлов.

5. М-р Копи: "Нажмите PLAY".

На доске вверху 'HEADERLESS' - без заголовка; программа ждет ввода файла без заголовка. Как только файл будет загружен в память программы м-р Копи говорит: "Нажмите REG и PLAY, затем 0". По нажатию '0' происходит запись файла без заголовка на ленту. После записи программа возвращается к главному меню. Файл без заголовка можно проверить в режиме 3.

8. М-р Копи: "Вы уверены? тогда нажмите '6'. После нажатия '6' происходит сброс программы, аналогично нажатию кнопки RESET.

При нажатии BREAK в любом режиме м-р Копи: "Нажата клавиша BREAK - нажмите '0'". По нажатию '0' программа возвращается к главному меню.

11. COPY DE LUXE

SPACE - пауза. Исходный режим; ожидание команды. Действует во всех рабочих режимах (из режима NAME-N включается через ENTER).

L - LOAD загрузка в память файлов с заголовком. Высвечивается заголовок файла, его тип, параметры и оставшаяся свободная память. В случае ошибки при загрузке файл стирается.

D - DATA загрузка блоков без заголовка. Высвечивается длина блока и оставшаяся память. При переполнении памяти подсчитывается, тем не менее, длина блока. В случае ошибки при загрузке можно сохранить блок для последующей записи на ленту, нажав любую клавишу, кроме SPACE или BREAK. При нажатии на SPACE память освобождается для новой загрузки.

H - HEADER просмотр заголовков файлов. Загрузка самих файлов не происходит.

S - SAVE запись на ленту, начиная с 1-го файла или с того, который выбран в режиме VIEW. Звуковой сигнал в конце записи можно предотвратить, своевременно нажав SPACE.

V - VIEW обзор файлов, находящихся в памяти (максимум 8) и выбор начала записи. Из этого режима можно перейти в режим изменения заголовка нажатием N-NAME.

K - CANCEL ликвидация последнего из загруженных файлов.

R - RESET полная очистка памяти копировщика.

Q - QUIT ликвидация копировщика.

M - MAXBUTE максимальное использование ОЗУ (без заголовка).

При необходимости заголовок копируется в режиме D-DATA. При нажатии на M выбирается один из трех режимов нажатием на клавиши 1, 2 или 3. Если клавиша цифровая не нажата, то возможен отказ от режима M через SPACE. Выбранный вариант режима изменить нельзя и выход из режима MAXBUTE не существует.

M1 - многократная загрузка и запись ценой менее полного использования ОЗУ. L - LOAD, S - SAVE.

M2 - однократная загрузка и многократная запись блока известной длины (ее можно предварительно измерить в режиме DATA). Возможно копирование блоков длиной до 65535 байт с наложением на ПЗУ. Перед загрузкой точно установить длину копируемого блока клавишами Z и X (CAPS SHIFT ускоряет их действие). При несоответствии длины блока установленному значению возможен сбой. L - LOAD, ENTER - SAVE.

M3 - однократная загрузка и многократная запись блока произвольной длины с максимальным использованием ОЗУ. L - LOAD, ENTER - SAVE.

12. LERM - 7

Введение.

Перед загрузкой программы убедитесь, что память компьютера очищена. Любая периферия, кроме интерфейса-1, должна быть отключена.

Не пытайтесь копировать наиболее поздние программы, не набрав достаточного опыта. Они хорошо защищены.

Техническая информация.

Этот раздел помещен для расширения Ваших знаний по Спектруму, а также для облегчения операции копирования программы.

1. Пилоттон (иногда просто 'тон', сокращенно - T).

При загрузке программ или при их записи на ленту Вы видите в самом начале широкие синие и красные полосы. Это пилоттон T. Он обеспечивает надежность ввода кодов в память компьютера.

2. Джеркитон (сокращенно - J).

Это прерывистый пилоттон. Он был разработан для защиты от копирования программы. Вместо плавного перемещения красно-голубых полос наблюдается их движение рывками, сопровождающееся отрывистыми звуковыми сигналами. Этот способ довольно широко применяется в последнее время, например фирмами OCEAN, ULTIMATE и другими.

3. Байты (сокращенно B).

После тона идут байты. Они сопровождаются узкими синими и

желтыми полосами на экране. Программист может сделать их другого цвета, а может совсем не выводить, используя свой загрузчик.

Байты - это числа в диапазоне от 0 до 255. Они представляют и бейсик, и машинные коды, и данные. Все это байты, но они имеют разные функции.

4. Блок кодов.

Пилоттон вместе с последующими данными называется блоком кодов.

5. Часть программы

Рассмотрим загрузку в компьютер простой бейсик-программы. Сначала мы видим тон, затем короткий пакет байтов B. Это первый блок. Затем после коротткого промежутка Вы увидите второй тон T и опять B, т.е. второй блок кодов.

```
Блок 1                    Блок 2
-------------- пауза ---------------
T     B             T     B
```

Совокупность этих двух блоков называется частью программы.

Так, если Вы записываете бейсик-программу TB-TB, а затем, скажем, массив данных TB-TB, то Вы получите две части программы. Если же Вы записываете бейсик, затем блок машинных кодов, а затем блок данных, то программа будет иметь три части.

6. Скорость загрузки/выгрузки.

Обычно SPECTRUM выполняет LOAD и SAVE со вполне определенной скоростью. Как правило, это 1500 бод (бит/сек). Если Вы увеличите темп и снизите время загрузки, Вам придется расплачиваться снижением надежности ввода. Современные программы иногда умеют увеличить скорость ввода и LERM-7 может выполнить копирование таких программ.

Загрузка программы с повышенной скоростью иногда сопровождается следующими явлениями:

- ширина желто-голубых полос меньше;
- высота звука выше.

Для обеспечения защиты програм от копирования ряд фирм разработали свои собственные пилоттоны. При этом первая часть программы должна оставаться 'нормальной', иначе SPECTRUM не сможет загрузить ее. После этого программист может специальной программой в машинных кодах обойти обычную систему загрузки и создать свою, например:

- создать значительно более узкий пилоттон;
- создать значительно более широкий пилоттон;
- создать очень короткий пилоттон - 0,1 с вместо 2...5 с.

При этом бордюрные линии становятся широкими сине-желтыми. Этот метод называется тонингом;

- записать программу без зазоров между блоками кодов.

Если при этом послушать запись, то слышен один непрерывный тон, а полосы на экране будут широкими желто-синими.

Непрерывный тон сбивает копирующие программы, поскольку они ждут зазора. С этим методом вполне справляется копирующая программа LERM-6, для ряда из этих методов она имеет специальные клавиши.

Основные способы защиты программ:

- использование очень длинных программ, более 48K;
- применение различных скоростей ввода ('AUTOMENIA');

- прерывистый пилоттон - джеркитон ('DECATHLON');
- тонинг ('CHUCKIE EGG');
- короткий пилоттон;
- конбинация различных методов ('DECATHLON').

Это неполный список, см., например, ниже HEADERLESS - блоки (блоки без заголовка).

Заголовок (HEADER).

Рассмотрим загрузку бейсик-программы. Сначала Вы видите широкие красно-голубые полосы пилоттона, слышите звук. Это идет подготовка к загрузке. Затем короткая серия узких желто-синих полос. Это первый блок: ТВ. Здесь В - заголовок (хедер). Он короткий, т.к. содержит только 17 байт. Именно поэтому на экране появляется сообщение:

PROGRAMM: FRED

или

BYTES: FRED

После загрузки еще одного блока мы будем иметь:

```
Блок 1                    Блок 2
-------------- пауза ---------------
T     H                   T     B
```

[H] - это хедер (раньше мы писали [B]).

Данные о хедере:

SPECTRUM имеет 17 байтов в хедере. Рассмотрим состав этих данных. Например:

HEAD:0-250-1-20

Вы видите 4 числа. Их всегда должно быть 4. Первое число:

0 - бейсик;
1 - числовой массив;
2 - стринговый массив;
3 - коды.

Отсюда видно, что пример, приведенный выше, записан в бейсике!

Второе число: количество байтов, подлежащих загрузке, в нашем примере - 250.

Третье число: в бейсике здесь задается номер строки, с которой программа должна автостартовать. Если это число больше, чем 32768, то автостарт не происходит.

В кодах, т.е. если первое число равно 3, здесь указывается на адрес, в который должен быть помещен первый байт.

Четвертое число: задает количество байтов бейсика.

Пример: HEAD: 3-780-36874-780. Задает блок в кодах, длина его 780 байт. Загрузка начинается с адреса 36874.

Все программы должны начинаться с хедера, за которым идут байты: ТН.

Программируя в машинных кодах, можно обойтись без хедера, поскольку можно управлять тем, какие байты куда будут загружены. Например:

```
Tp 3
TH TB TB TB
----------  <----- эти две части программы не имеют
```

хедеров.

Некоторые программисты вставляют в программы так называемые 'фальшхедеры'. Это блоки кодов длиной 17 байтов,

которые выглядят как хедеры. Иногда они имеют неразрешенное имя и предназначены для сбивания с толку копирующей программы.

LERM не обращает внимания на наличие хедера. Если Вы увидите сообщение:

PROGRAMM NAME FRED: ??????

то будете знать, что '?' стоит вместо неразрешенных символов, которые предназначались для вывода из строя копирующей программы.

Инструкция к LERM-7

Цель программы: копирование очень длинных программ, программ с высокоскоростными загрузчиками, программ с джеркованным пилоттоном.

В конце этого раздела помещены полезные советы. Им необходимо следовать. Это вызвано тем, что программы со скоростными загрузчиками гораздо более чувствительны к качеству ввода, чем программы с обычными загрузчиками.

Порядок работы:

1. Загрузите LERM-7 командой LOAD"";
2. После загрузки экран темнеет и появляется меню:

L - загрузка нормальных хедеров и байтов;
H - загрузка программ без хедеров;
S - запись того, что загружено в режимах L и H;
C - подсчет байтов (счетчик);
Q - прекращение работы (Q-NEW);
B - вход в режим 'B' для загрузки программ избыточной длины и програми со скоростными загрузчиками;
J - вход в режим 'J' для копирования программ с джеркованными пилоттонами.

Режимы 'H' и 'L' копируют только части программ, записанные с нормальной скоростью, длиной до 38590 байтов, здесь копируется только одна часть TH TB или один блок без заголовка TB. Этим она отличается от LERM-6. Наиболее целесообразно для нормальных блоков применять LERM-6, а для более трудных оставлять LERM-7, работая с ними в режимах 'B' и 'J'.

3. Копирование программ

а) загрузив TC-7, установить программу, которую Вы хотите скопировать;

б) нажмите 'L' на Спектруме и 'PLAY' на магнитофоне. Убедитесь в том, что уровень сигнала установлен в 'обычном' для загрузки программы положении.

в) после загрузки первой части TH TB, остановите магнитофон. Вставьте чистую ленту, включите запись, нажмите 'S' на компьютере. Чтобы выполнить вторую копию на другой ленте, нажмите 'S' еще раз.

г) дальнейшие части программы могут быть 'нормальными', с хедерами и байтами, а могут быть без хедеров. Для копирования первых используйте 'L', для вторых - 'H'. Если Вы не уверены, пользуйтесь 'H'. Загрузив очередную часть, повторите пункт в).

д) повторяйте пункт г) до тех пор, пока Вам не встретится:

- часть с джеркованными пилоттонами;
- блок с увеличенной скоростью загрузки;
- очень длинный блок;
- блок с очень коротким пилоттоном.

Для работы с такими блоками Вам надо пользоваться режимами 'B' или 'J'. Для первого - 'J', а для второго - 'B'.

4. Счетчик ('COUNTER').

Эта операция позволяет посчитать количество байтов. Она работает и на скоростных загрузчиках, но не с джерками.

Работа счетчика:

1. установить начало блока, подлежащего измерению;

2. нажать 'PLAY' на магнитофоне и 'C' на компьютере. Цвет бордюра изменится. После прогона блока на экране будет изображена его длина.

Внимание! Если после этого Вы перейдете в режим 'B', то замеренная длина будет передана в этот режим, где есть свой собственный счетчик. Это сэкономит время, т.к. измерение длины придется выполнять только один раз.

Если программа имеет блок данных и Вы не уверены, сможет ли режим 'H' загрузить ее, то после измерения 'C' назначьте 'H', если блок менее 38K. Если блок больше 38K, переходите в режим 'B', при этом измерять длину блока уже не нужно.

Кроме того, имейте в виду, что весьма важно знать длину блока до перехода в режим 'B', т.к. выбор клавиши для его копирования зависит от длины блока.

5. Прекращение работы осуществляется клавишей 'Q'.

6. Фальшхедеры.

Режим 'L' не может загружать фальшхедеры, но это можно преодолеть, если копировать каждый блок по отдельности через 'H'.

TF TB

Здесь F - это фальшхедеры.

7. Вход в режимы 'B' и 'J'.

Для этого нажмите 'B' или 'J'. Бордюр станет красным, а на экране появится сообщение: 'Нажать любую клавишу'.

Замечание: Вы обнаружите, что режимы 'B' и 'J' обычно смещают картинку на экране на один квадрат. Это не ошибка. Во всем остальном картина должна быть нормальной. Если нет, то проверьте уровень громкости.

Когда бордюр станет красным, нажмите пробел. Экран станет черным, бордюр желтым. Если это не так, попробуйте еще раз.

Замечание: в режимах 'B' и 'J' клавиша 'H' предназначена для измерения длины, что она и делает в специально выделенной для этого области памяти. Поэтому если Вы намерены делать 'SAVE' еще раз, ни для чего другого эту клавишу использовать нельзя, т.к. в программе будут испорчены байты. Заметьте также, что если программа очень длинная, то счетчик 'B' может быть 'затерт' загружаемой программой. В этом случае клавиша 'B' станет бесполезной.

8. Режим 'J'.

Этот режим работает с джеркитонами. Предупреждаем сразу, что это не просто, см. раздел 'полезные советы'. Эти загрузчики обычно являются также скоростными, но они запрограммированы, как правило, на одну и ту же скорость.

Меню 'J' -режима следующее (на экране не показывается):

B - измеритель джерков;

L - первая загрузочная программа;

M – первая загрузочная программа;
S – клавиша SAVE;
S/Q – 'SAVE' с нормальным пилоттоном и с нормальной скоростью;
A – прерывание. Очищается экран, бордюр становится желтым;
Q – сброс (аналогично 'NEW').

9. Начало работы.

Прежде всего Вам надо померить 'джерки' клавишей 'B'. Для этого перемотайте ленту и установите ее в начало джеркитона. Нажмите 'PLAY' на магнитофоне и держите 'B' на компьютере до тех пор, пока пройдет вся джеркованная часть программы. Как только пойдут байты, нажмите SPACE и выключите магнитофон. Если Вы будете ждать слишком долго, программа может сбиться.

Фактически нет необходимости в перегоне всего джеркитона, можно начать и с середины. Важно, чтобы измеритель захватил и несколько последующих байтов. Замер можно повторить. Обычно программа имеет один джеркитон на каждый блок.

10. Загрузка частей с джерками.

Для этой цели служат две клавиши – 'L' и 'M'. Более ранние программы обычно требуют 'L', а более поздние – 'M'. Чтобы выбрать, какая именно клавиша Вам нужна, надо:

а) установить программу в том же месте, где начинается загрузка экрана;

б) нажать 'PLAY' на магнитофоне и 'L' на компьютере. Если цветовые атрибуты идут правильно, значит клавиша 'L' выбрана верно;

в) в противном случае использовать клавишу 'M'.

11. Запись.

Установите чистую кассету, включите 'RECORD' и нажмите клавишу 'S' для записи с джеркитонами или для записи без них.

12. Клавиша 'A'.

Нажатие этой клавиши приводит к очистке экрана, но она не влияет на состояние измерителя джерков.

13. Прочие возможности.

Нажатием клавиши SPACE во время загрузки или выгрузки Вы можете прервать работу программы и возвратиться к главному меню.

14. Режим 'B'.

В этом режиме есть возможность копирования программ со скоростными загрузчиками, в том числе и тех, которые при загрузке могут выполнять проверку экрана.

Меню 'B'-режима:

L – загрузка блоков длиной до 48300 байтов;
M – загрузка блоков от 48300 до 50K;
N – загрузка блоков свыше 50K;
J – загрузка блоков, если режим ' ' не работает (только для нормальной скорости);
Z – загрузка длинных блоков, начинающихся на экранной области;
S – запись блоков;
P – спец.клавиша для записи (используется после 'C');
(доллар) – запись с нормальной скоростью и нормальным пилоттоном;
C – подсчет числа байтов для работы в режимах 'J' и 'L';

B – измеритель числа скорости ввода;
Q – сброс;
A – прерывание и очистка экрана.

Замечание: после использования клавиш J или Z для загрузки и последующей выгрузки, программа автоматически выполняет NEW.

15. Изменение скорости ввода.

При первом входе в LERM в компьютере задана нормальная скорость ввода/вывода. Если блок имеет нестандартную скорость, то прежде чем что-либо делать, Вам необходимо замерить скорость, для этого:

а) установите ленту в начале того блока, который Вы собираетесь копировать. Нажмите 'PLAY' и пропустите пилоттон;

б) когда пилоттон пройдет и пройдут байты, нажмите 'B', постарайтесь найти достаточно 'шумную' зону (при изображении экрана, например, встречаются подряд сотни одинаковых байтов, что воспринимается как один чистый звуковой тон).

Примечание: некоторые программы имеют различные скорости для различных блоков, поэтому измеритель надо применять всякий раз снова. Можно проверить, правильно ли производится замер скорости. Если в программе есть SCREENS, попробуйте загрузить экран операцией 'L'. Если экран и атрибуты правильные, но сдвинуты на один элементарный квадрат, то значит замеренное значение – правильное.

16. Операции загрузки.

Самая большая проблема состоит в том, что кроме повышенной скорости блоки могут иметь и избыточную длину. Чтобы с ними справиться, существуют операции, которые загружают программу за счет порчи картины экрана. В то же время, есть программы, которые выполняют проверку экрана, бывают также программы более 48K.

а) клавиша 'L'.

Загружает блоки объемом до 48300 байтов. Картина экрана сдвигается на один элементарный квадрат. После копирования никакой порчи экрана нет. Это основной режим загрузки.

б) клавиша 'M'.

Загружает блоки от 48300 до 50K. Картинка портится. Копия также будет иметь испорченную картинку. Тем не менее, если программа не проверяет состояние экрана, то она будет работать нормально. Этот режим – Ваша первая линия атаки на блоки, превышающие 48.3K.

Все остальные режимы предполагают предварительное использование счетчика 'C'. Помните, что счетчик может быть задействован как до, так и после перехода в режим 'B'.

в) клавиша 'N'.

Загружает программы более длинные, чем 'M', но искажение картинки здесь присутствует также.

г) клавиша 'J'.

Применяется для блоков, которые загружаются с нормальной скоростью и имеют длину более 48.3K и не начинается с загрузки экрана. В этом случае картинка не портится.

д) клавиша 'Z'.

Применяется, как и клавиша 'J', но в тех случаях, когда блок начинает загружаться с экрана.

Режиме. Итак, Вам надо определить длину своего блока, используя 'C' и выбрать загрузочную клавишу в зависимости от длины блока, скорости загрузки и наличия проверки искажений экрана.

Что же делать, если программа имеет и скоростной загрузчик, и проводит проверку экрана одновременно? Для этого есть специальный обходной путь, см. режим 'P'.

17. Загрузка.

Установить ленту на начало блока, который Вы собираетесь копировать. Нажмите 'PLAY' и соответствующую загрузочную клавишу. После загрузки остановите ленту.

18. Запись.

а) клавиша 'S' (нормальная запись).

Вставьте чистую ленту в магнитофон. Нажмите 'RECORD' и затем 'S'. Программа будет записана с той же скоростью, с какой загружалась. Повторение нажатия 'S' даст повторение записи, за исключением режимов 'J' и 'Z', о чем упоминалось выше.

б) клавиша 0.

Если у Вас программа со скоростным загрузчиком, а Вы хотите конвертировать ее в нормальную, нажмите 0 до нажатия 'S'.

в) клавиша 'P' (используется с 'C' и 'L').

Если у Вас есть скоростной блок, слишком длинный для 'L', но в то же время использовать 'M' нельзя, т.к. в программе есть проверка области экрана, то Вы можете попробовать следующее (хотя все это метод проб и ошибок):

- замерить длину 'C';
- загрузить через 'L';
- записать через 'P'.

Заметьте, что счетчик 'C' может быть запущен либо после главного меню (см.п.2), либо из режима 'B' (см.п.19 ниже).

19. Клавиша 'C' для подсчета байтов.

В этом режиме клавиша 'C' работает точно также, как и в п.2. После нажатия 'PLAY' на магнитофоне просто держите клавишу 'C'. Хотя при этом отсчет байтов на экране не изображается.

20. Клавиша 'Q'.

Клавиша 'Q' сбрасывает программу. Ее действие аналогично команде 'NEW'.

21. Клавиша 'A'.

Она очищает экран и устанавливает все в исходное положение.

22. Полезные советы.

Выполнение копий с джеркованных программ и программ со скоростными загрузчиками - это довольно сложная задача, потому что все прочие факторы становятся гораздо более критичными. Вам придется экспериментировать со своим аппаратным обеспечением. Обычно вопрос правильности ввода программы упирается в правильный подбор громкости сигнала.

1. Желательно иметь магнитофон попроще, по возможности без автоматической регулировки уровня сигнала при записи. Головка должна быть чистой и точно установленной. Лента должна быть высококачественной, а оригинал - в достаточно пригодном состоянии. Если программа загружается только один раз из четырех, то Вам придется потратить больше времени на ее ввод.

2. Загружайте программу в компьютер с уровнем громкости, который Вы обычно используете для ввода программ.

13. TURBO - COMP

Этот копировщик позволяет работать с магнитной лентой как в нормальном режиме, так и с повышенной скоростью загрузки и сохранения файлов. Файлы, записанные копировщиком в режиме TURBO, могут свободно запускаться на компьютере, в котором предусмотрен такой режим загрузки программ. В противном случае этот режим может использоваться только для удобного хранения информации. В копировщике используется компрессия принимаемой информации.

Команды:

LOAD - (клавиша 'L') LOAD 1 - позволяет загрузить программу с номером 1; если ранее уже был загружен файл, то загрузка идет с очередного номера;

Пример:

```
01      LST     4.5     BASIC       0           44
02                                                44
03      LST     4.5     CODE      25500       4000
```

SAVE - (клавиша 'S') сохранение файлов на магнитной ленте; SAVE N01 TO 04 - сохранение файлов с номерами от 1 до 4; по окончании номера записываемых файлов закрашиваются в другой цвет;

ВНИМАНИЕ! После нажатия последней клавиши команды эта команда выполняется сразу.

DELETE - (клавиша 'D') удаление файлов из памяти копировщика;

TURBO (клавиша 'T') - изменение режима работы с файлами (L-H-T):

L - загрузка и сохранение файлов с нормальной скоростью;

H - загрузка и сохранение файлов с нормальной скоростью пилот-сигнала и ускоренными данными;

T - загрузка и сохранение файлов в режиме TURBO с ускоренным пилот-тоном и ускоренным блоком данных (устанавливается при вызове программы).

RESET (клавиша 'R') - возврат в BASIC.

Прерывание работы в режимах LOAD, SAVE выполняется нажатием на клавишу BREAK.

Режим проверки записи в данном копировщике не предусмотрен. Проверку можно выполнить повторной загрузкой в память копировщика только что записанной информации.

14. QUICK SAVE

Этот копировщик предоставляет великолепную возможность создавать на лентах программы, время загрузки которых по сравнению с обычными уменьшено в 2-3 раза, не требуя никаких доработок SPEKTRUMa или подзагрузки дополнительных программ.

Копировщик выполняет следующие команды:

L - (LOAD) - загрузка программных блоков в память (до 41K9;

Q - (QUICKSAVE) - выгрузка всего содержимого памяти на ленту с повышенной скоростью (выгрузка выполняется сразу после нажатия клавиши 'Q';

C - (CLEAR) - очистка всей памяти копировщика;

B - (BASIC) - выход из копировщика;

R (RATE) - изменение скорости выгрузки.

Для прерывания работы копировщика используется клавиша 'SPACE'.

Программа, сброшенная на ленту с помощью копировщика QUICK SAVE, загружается для исполнения обычным способом с помощью команды BASICa

LOAD "",

но уже не может быть скопирована с этой ленты даже самим QUICK SAVE.

ВНИМАНИЕ! Для обеспечения надежной работы программы используйте магнитофон и магнитные ленты высокого качества.

15. ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ ПАСКАЛЬ

ОПИСАНИЕ РЕДАКТОРА HP80

УПРАВЛЕНИЕ РЕДАКТОРОМ:

CAPS-SHIFT 1 EDIT - редактировать строку, выделенную экранным курсором;
CAPS-SHIFT 2 CAPS-LOCK - фиксация регистра малых или больших символов;
CAPS-SHIFT 3 TRUE-VIDEO - страница вперед;
CAPS-SHIFT 4 INV-VIDEO - страница назад;
CAPS-SHIFT 5 <--- - курсор влево/ строка влево;
CAPS-SHIFT 6 СТРЕЛКА ВНИЗ - экранный курсор вниз;
CAPS-SHIFT 7 СТРЕЛКА ВВЕРХ - экранный курсор вверх;
CAPS-SHIFT 8 ---> - курсор вправо/ строка вправо;
CAPS-SHIFT 9 GRAPHIC - включить графический регистр;
CAPS-SHIFT 0 DELETE - удалить символ;
CAPS-SHIFT Q - перейти в начало текста;
CAPS-SHIFT W - перейти в конец текста;
CAPS-SHIFT E - установить режим редактирования;
CAPS-SHIFT I - установить режим ввода символов;
CAPS-SHIFT A - установить режим добавления символов;
CAPS-SHIFT S - найти образец;
CAPS-SHIFT D - удалить строку, выделенную экранным курсором.

ВНИМАНИЕ! Звуковой сигнал сообщает о нажатии недопустимой клавиши.

SYMBOL-SHIFT CAPS-SHIFT - запрос "COMMAND" (КОМАНДА): допустимые ответы:

4 OPEN# - открыть блок текста;
5 CLOSE - закрыть блок текста;
6 MOVE - перенести блок текста;
7 ERASE - удалить блок текста;
0 FORMAT - переключить формат экрана 64 на 32 символа и наоборот;
Z COPY - копировать блок текста;
X CLEAR - очистить текстовый буфер;
Y RETURN - перейти к HISOFT формату редактирования и вводить текст средствами БЕЙСИКА;
F - задать образец и установить режим поиска;
R - задать литерал для замены.

ВНИМАНИЕ! Используйте стандартный HP редактор для выполнения прочих команд (C,R,G,P...).

Команды режима поиска:

S - найти образец;
R - заменить найденный образец на заданный литерал;
G - перейти к замене остальных образцов;

ВНИМАНИЕ! Не допускается использование графического регистра вне текста программы.

Далее приводятся в основном сведения, касающиеся отличий от стандартного ПАСКАЛЯ.

ПАСКАЛЬ не имеет встроенных графических процедур и функций.

ОГРАНИЧЕНИЯ:

- тип FILE не реализован;
- тип RECORD не может иметь вариантные поля;
- процедуры и функции не могут являться формальными параметрами;
- недопустимо использование указателей на ранее не определенный тип.

КОМПИЛЯЦИЯ И ИСПОЛНЕНИЕ:

Компиляция: C LINE NUMBER
Остановка листинга: BREAK

В случае успешной компиляции программы на **RUN?** ответить Y, если требуется исполнение программы, либо любым другим символом для возврата в редактор.

Данная реализация ПАСКАЛЯ при контроле правильного использования типов использует эквивалентность имен, а не структур, поэтому переменные

```
VAR A:ARRAY[1..3] OF T;
    B:ARRAY[1..3] OF T;
```

не могут использоваться в выражении совместно. Недопустимо также присваивание

```
A:=B;
```

СИНТАКСИС И СЕМАНТИКА

Число без знака ::=#шестнадцатиричное число;

Строковая константа записывается в кавычках " ", максимальная ее длина 255 символов.

Символьная константа ::CHR("КОНСТАНТА") для управляющих символов CR, LF, NULL и т.д.

Ключевое слово PACKED игнорируется.

Тип множества может иметь от 0 до 256 членов.

Пустой поиск в операторе CASE приводит к ошибке.

Если подходящая ветвь списка CASE не найдена, управление передается оператору, следующему за END.

Управляющая переменная оператора FOR должна быть простой и не может являться параметром.

Оператор GO TO должен передавать управление внутри того блока, где он находится.

Оператор GO TO должен находиться на одном уровне вложения с меткой.

Программа не имеет параметров.

ПРЕДОПРЕДЕЛЕННЫЕ ИДЕНТИФИКАТОРЫ:

Константы: MAXINT ... 32767
TRUE, FALSE

Типы:
INTEGER ... 2 байта;
REAL ... 4 байта;
CHAR ... расширенный ASCII... 256 символов;
BOOLEAN

ПРОЦЕДУРЫ И ФУНКЦИИ

Ввод и вывод:
WRITE

WRITELN - на дисплей или на принтер;
CHR(8) - BS на дисплее (шаг назад);
CHR(12) - очистка экрана (CLEAR SCREEN или NEWPAGE);
CHR(13) - CR и LF (ВК или ПС);
CHR(16) - переключить вывод с дисплея на принтер и наоборот;
READ
READLN - с клавиатуры;
INCH - функция читает символ с клавиатуры, если не нажата ни одна клавиша, возвращает CHR(0).
Арифметические функции:
FRAC - дробная часть.
Другие процедуры и функции:
- исключена процедура DISPOSE, вместо нее MARK и RELEASE
- INLINE (C1,C2,...)
- C1 - шестнадцатиричные числа, размещаются в последовательность байтов;
- V - целое, вызывает подпрограмму с адресом V, она должна завершаться инструкцией RET;
- HALT - завершает программу;
- POKE(X,V) - помещает значение V в ячейку памяти с адресом X;
- TOUT (NAME, START, SIZE) - записывает данные на магнитную ленту;
- TIN (NAME, START) - читает данные с магнитной ленты;
- OUT(P,C) - эквивалентно BC:=P; A:=C; OUT (C),A;
- RANDOM - функция возвращает случайное число в диапазоне 0...255.
- ADDR(V) - функция возвращает адрес аргумента произвольного типа;
- PEEK(X,T)
- X - целое, задает адрес, второй аргумент может иметь произвольный тип, задает значение, которое заносится в ячейку памяти;
- SIZE(V) - функция возвращает размер области памяти, занимаемой переменной V;
- INP(P) - аналогично OUT.

КОММЕНТАРИИ: "(*" "*)" и "{" "}"

ОПЦИИ (КЛЮЧИ): размещаются внутри комментариев вслед за символом $ (доллар). Состоят из знака "+" или "-", за которым следует один из символов:

L - управление листингом при компиляции (по умолчанию +L);
O - контроль переполнения (по умолчанию +O);
S - контроль переполнения стека (по умолчанию +S);
A - контроль границ массивов (по умолчанию A);
I - прочитать часть программы с магнитной ленты, позволяют
P - экономить память.
F -

СТАНДАРТНЫЙ РЕДАКТОР:

I N,M – режим автоматической нумерации строк программы;
L N,M – распечатать на дисплее текст с номерами строк от N до M;
D N,M – удалить строки текста с номерами от N до M;
K N – установить число строк текста, печатаемых без останова при выдаче листинга, равным N;
M N,M – заменить строку с номером M на строку с номером N;

РЕДАКТИРОВАНИЕ:

F N,M,F,S – найти в диапазоне строк с номерами N,M образец F и заменить его на S;
E N – редактировать строку с номером N:
подкоманды:
" " (пробел) – инкремент текстового указателя;
DELETE – декремент текстового указателя;
CAPS SHIFT/I – табуляция вправо;
ENTER – конец редактирования;
Q – отменить редактирование;
R – начать редактирование заново;
F – найти ранее заданный образец;
S – заменить образец;
I – вводить символы с позиции курсора с автораздвижкой (завершается ENTER);
X – курсор перемещается в конец строки;
C – вводить символы с позиции курсора с уничтожением исходного текста (завершается ENTER);
K – удалить символ под курсором;
Z – удалить символы от курсора до конца строки.

КОМАНДЫ ДЛЯ РАБОТЫ С МАГНИТОФОНОМ:

P N,M,S – записать текст от строки N до строки M в файл с именем S;
G ,,S – прочитать текст из файла с именем S;
W N,M,S – записать текст в файл в форме, которая потом может использоваться с ключом F N,M,S.

КОМПИЛЯЦИЯ И ИСПОЛНЕНИЕ ПРОГРАММ:

C N – компилировать текст, начиная со строки N;
R – исполнить скомпилированную программу;
T – отлаженная программа компилируется таким образом, что перекрывает в памяти компилятор. Позднее может быть записана на магнитную ленту и исполнена автономно;
B – возврат в BASIC;
O N,M – используется для работы с текстом, подготовленным другим редактором.

14. ПРОГРАММА "TUTOR"
(35 уроков по ассемблеру)

Программа "TUTOR" научит Вас использовать все команды микропроцессора Z80. Программа содержит 35 уроков (35-й урок - заключение), после каждого из которых можно выполнить соответствующие примеры. По мере освоения тем, Вы можете изменять эти примеры, а затем запускать их, не опасаясь разрушения системы.

Используйте клавишу "ENTER" для вызова помеченного урока или примера, и "SPACE" для перехода к следующему уроку (примеру). Нажатие клавиш "CS" и "BREAK" возвратит Вас в меню.

Ч А С Т Ь П Е Р В А Я

З А Н Я Т И Е 1

РЕГИСТРЫ И ПАМЯТЬ

Регистры - это группа одноразрядных запоминающих ячеек, находящихся в микропроцессоре. Один регистр содержит 8 таких ячеек (или разрядов), и поэтому может запомнить число, не превышающее 255. МП Z80 имеет большое количество таких регистров, но первоначально мы рассмотрим наиболее часто используемые. Они обозначаются буквами: A,B,C,D,E,H и L. Регистр A (обычно называемый аккумулятором) является среди них наиболее важным, т.к. большинство команд использует этот регистр чаще, чем остальные. Кроме регистров, содержащихся в микропроцессоре, компьютер имеет множество ячеек памяти, являющихся внешними по отношению к микропроцессору. Эта память обозначается в литературе сокращением RAM (в англоязычной) или ОЗУ (в русской), и служит для размещения команд и больших массивов информации. Местоположение каждой ячейки памяти однозначно определяется ее адресом в адресном пространстве. Для МП Z80 адреса могут лежать в диапазоне 0..65535. Следует также заметить, что доступ для занесения или чтения информации к ОЗУ осуществляется медленнее, чем к регистрам.

З А Н Я Т И Е 2

ПРОСТАЯ КОМАНДА ЗАГРУЗКИ

Команда загрузки производит копирование содержимого регистра или ячейки памяти в другой регистр (ячейку памяти). При копировании содержимое источника не изменяется. Мнемоника (условное обозначение) команды загрузки: LD.

Простейшая команда загрузки - копирование из одного регистра в другой, например: LD A,B. В этой команде первым указывается регистр-приемник, а вторым - регистр-источник. Так, рассмотренная команда LD A,B производит копирование содержимого регистра B в регистр A.

Вы можете, также, записать в регистр некоторое число, не

превышающее 255, командой LD r,n. При этом в регистр r будет записано число n.

В ОЗУ машинные команды записываются в виде чисел. Команда простой загрузки требует для размещения 1 байта (ячейки памяти), например, команда LD A,B записывается в память в виде числа 78, вообще же существуют команды, требующие до 4-х байтов.

Две рассмотренные команды (LD r,r' и LD r,n) могут оперировать с любым регистром (регистрами). Команда загрузки из ОЗУ может использовать только аккумулятор. Ее вид: LD A,(nn). При ее исполнении содержимое ячейки ОЗУ по адресу nn скопируется в аккумулятор. Аналогично, можно скопировать содержимое аккумулятора в ячейку ОЗУ командой LD (nn),A.

Подведем итог результатам занятия.

```
LD r,r'    r и r' - один из регистров A..E,H,L;
LD r,n     n      - число в диапазоне 0..255;
LD A,(nn)  (nn)   - число в диапазоне 0..65535;
LD (nn),A
```

ВВЕДЕНИЕ В ИСПОЛНЯЮЩУЮ СИСТЕМУ (ЭМУЛЯТОР АССЕМБЛЕРА)

Все упражнения выполняются посредством встроенного эмулятора ассемблера. При работе в нижней части экрана выводится информация о текущем состоянии регистров, а в верхней части - исполняемый код, мнемоника команд и т.п. По ходу дела, Вам будет даваться необходимая информация об эмуляторе.

Ассемблер, преобразующий мнемонические команды в машинный код, допускает использование меток. При этом вместо физического адреса в команду можно подставлять символическое имя метки, что гораздо удобнее. Одновременно мы можем заносить в ячейку информацию (константу и т.п.). Для этого служит директива (не команда!) DEFB (define byte) и DEFW (define word). Об этом мы поговорим позже. В ходе покомандного выполнения упражнения внизу экрана выводится краткое пояснение к текущей команде. Для перехода к следующей команде необходимо нажать любую клавишу.

Вы можете, если хотите, выполнить упражнение несколько раз. Если Вы хотите прервать выполнение упражнения, нажмите клавиши CAPS SHIFT+BREAK, при этом Вы выйдете в главное меню.

Находясь в эмуляторе, Вы можете использовать встроенный редактор для изменения текста программы упражнения. В конце программы обязательно должна стоять команда RET. Если Вы забудете ее поставить, эмулятор Вам об этом напомнит. Не бойтесь экспериментировать и помните, что никакая программа не может вывести Ваш Спектрум из строя.

З А Н Я Т И Е 3

РЕГИСТРОВЫЕ ПАРЫ

Один регистр может вместить в себя число до 255 включительно, память же занимает адреса по 65535 включительно. По этой причине целый ряд команд Z80 имеет дело с парами регистров. В пары могут объединяться регистры B,C,D,E,H,L, образуя при этом пары BC, DE и HL.

Два регистра содержат две части одного числа. Рассмотрим в

качестве примера десятичное число 27. Мы можем представить его в виде двух частей: старшая часть, которую можно изобразить как 2*10 и младшая часть, которую можно изобразить как 7*1, т.е. 2*10+7*1=27. При этом мы используем цифры от 0 до 9, т.е. 10 различных цифр.

Один регистр вмещает число до 255, т.е. 256 различных чисел. Если мы возьмем 2 регистра, например H и L, и поставим их рядом, при условии, что H содержит старший байт числа, а L - младший, то мы сможем разместить в регистровой паре HL число, равное H*256+L*1, так же, как мы брали 27=2*10+7*1. Максимальное число, которое может вместить наша регистровая пара будет равно 256*256+255=65535. Условились, что старший байт размещается при этом в регистре, название которого стоит первым в обозначении регистровой пары (в нашем случае - в регистре H).

Мы можем загрузить любую регистровую пару числом 0..65535 непосредственно используя команду типа LD dd,nn, где dd - регистровая пара, nn - число 0..65535.

Точно так же, как мы соединяли два регистра в пару, мы можем соединить две соседние ячейки ОЗУ. При этом младший байт числа будет содержаться в ячейке с меньшим адресом.

Мы можем также загрузить регистровую пару содержимым пары ячеек ОЗУ при помощи команды LD dd',(nn). Такая команда эквивалентна двум командам: LD (nn),d и LD (nn+1),d'.

Не существует команды, при помощи которой можно было бы непосредственно загрузить регистровую пару содержимым другой регистровой пары. Однако, существует возможность поменять местами содержимые регистровых пар DE и HL, для чего служит команда EX DE,HL.

Подведем итог результатам занятия.

```
LD dd,nn      dd - любая регистровая пара,
              nn - число 0..65535
LD dd,(nn)
LD (nn),dd
EX DE,HL.
```

Упражнение к занятию 3.

Упражнение иллюстрирует применение различных типов команд для загрузки регистровых пар. При этом одновременно с демонстрацией состояния регистровой пары или пари ячеек памяти отображаются состояния каждого регистра/ячейки ОЗУ пары раздельно.

Обратите внимание на использование директивы DEFW. При ее выполнении ассемблер резервирует две ячейки ОЗУ и связывает их с именем метки, в ячейки записывается число, указанное после директивы.

Адрес, указанный в первой колонке - это адрес первого байта команды или числа. Сами машинные коды отображаются во второй колонке.

З А Н Я Т И Е 4

КОСВЕННАЯ АДРЕСАЦИЯ

До сих пор мы использовали только команды, в которых местоположение ячейки ОЗУ определялось в составе самой команды. Однако, широко используется способ определения адреса ячейки ОЗУ, как числа, записанного в регистровой паре. Такой способ адресации известен, как косвенная адресация. Например, команда LD B,(HL) загружает в регистр B содержимое ячейки ОЗУ, адрес которой находится в регистровой паре HL. Иными словами, содержимое пары HL интерпретируется, как адрес, по которому находится нужное нам число.

Любой одиночный регистр может быть загружен с использованием пары HL в качестве указателя исполнительного адреса. Точно так же, ячейки ОЗУ могут загружаться с использованием косвенной адресации через пару HL содержимым одиночного регистра.

В качестве указателя можно также использовать пары BC и DE, но в качестве одиночного регистра с ними можно использовать только аккумулятор. Например: LD A,(BC), LD (DE), A и т.п.

Подведем итог результатам занятия.

```
LD r,(HL)
LD (HL),r
LD A,(BC)
LD (BC),A
LD A,(DE)
LD (DE),A
```

Упражнение к занятию 4.

Это упражнение иллюстрирует применение косвенной адресации. На этом этапе продемонстрировать все богатейшие возможности, которые дает косвенная адресация, еще нет возможности, но она будет широко использоваться в последующих упражнениях.

З А Н Я Т И Е 5

СЛОЖЕНИЕ И ФЛАГ ПЕРЕНОСА

Операция сложения может быть произведена с любым регистром или регистровой парой МП Z80. Мнемоника команды сложения - ADD. Все одиночные регистры выполняют сложение с аккумулятором. При этом число (например, ADD A,6), содержимое регистра (например, ADD A,B), или содержимое косвенно адресуемой ячейки ОЗУ (например, ADD A,(HL)) складывается с числом, находящимся в аккумуляторе. Результат сложения также засылается в аккумулятор. При этом содержимое источника слагаемого не изменяется.

Регистровые пары выполняют операцию сложения с регистровой парой HL. В качестве источника второго слагаемого можно использовать пары BC и DE (например, ADD HL,BC). Результат сложения засылается в HL, содержимое источника слагаемого не изменяется.

Правильный результат сложения одиночных или спаренных регистров может быть получен только в том случае, когда результат может быть размещен в приемнике (регистре или регистровой паре). Если же результат сложения больше максимально возможного числа (255 для аккумулятора и 65535 для пары HL), возникает явление, называемое переносом. При этом процессор воздействует на одиночный бит, называемый флагом. Этот бит называется флагом переноса, и обозначается буквой C (carry). Если при выполнении сложения возникает перенос, то этот бит принимает значение 1, в остальных случаях он имеет значение 0.

Существует второй способ сложения, который может использоваться в Z80 с любыми одиночными регистрами или парами регистров. Он имеет название 'сложения с переносом' и обозначается ADC. Эта команда действует так же, как команда ADD с тем отличием, что если перед ее исполнением был введен флаг C, то результат увеличивается на единицу, или, что то же самое, флаг C прибавляется к результату. В принципе, команда ADD может быть заменена в любом контексте на ADC, но не наоборот.

Команда ADC может использоваться для сложения двух чисел разной длины, как это будет показано в упражнениях.

Подведен итог результатам занятия.

```
ADD A,n       n - число 0..255
ADD A,r       r - одиночный регистр
ADD A,(HL)
ADD HL,BC
ADD HL,DE
ADC A,n       n - число 0..255
ADC A,r       r - одиночный регистр
ADC A,(HL)
ADC HL,BC
ADC HL,DE
```

Упражнения к занятию 5.

1. Сложение одиночных регистров

В этом упражнении два числа (8740 и 1260) используют для сложения одиночные регистры. Обратите внимание на метод их сложения, т.к. он может использоваться и для сложения чисел разной длины. Следите за состоянием флагов, отображаемых слева от регистров.

2. Сложение регистровых пар.

Упражнение показывает способ умножения числа, записанного в пару HL на 10 путем последовательных многократных сложений.

З А Н Я Т И Е 6

ВЫЧИТАНИЕ И ФЛАГ ПЕРЕНОСА

Команда вычитания SUB имеет много общего с командой ADD. Результаты вычитания всегда засылаются в аккумулятор. Все варианты команды ADD применимы к команде SUB. Флаг переноса взводится в случае, если результаты вычитания превышают 255.

Главным отличием является невозможность использования в команде SUB регистровых пар.

Точно так же, по аналогии с командой ADC можно рассматривать команду SBC (вычитание с переносом). При ее выполнении из результата вычитается единица, если перед выполнением оказался взведенным флаг C. Подобно команде ADC команда SBC может использоваться для работы с числами разной длины. Так, вычитание регистровых пар может быть выполнено только с использованием переноса, при этом флаг C перед выполнением команды должен быть сброшен. Например, SBC HL,BC или SBC HL,DE.

Флаг C может быть принудительно взведен при помощи команды SCF. Специальная команда для принудительного сброса флага C в Z80 отсутствует, однако имеется команда инвертирования состояния флага C - команда CCF. Позднее мы увидим, что флаг C сбрасывается всеми логическими командами Z80.

Подведем итог результатам занятия.

```
SUB n------\
SUB r------ > - вычитание из A числа n, содержимого
SUB (HL)---/    регистра r или содержимого ОЗУ по адресу (HL)
SBC A,n----\
SBC A,r---- > - то же, с учетом флага C
SBC A,(HL)-/
SBC HL,BC--\  - вычитание из регистровой пары HL с
SBC HL,DE--/    переносом
CSF - взведение флага C
CCF - инвертирование флага C
```

Упражнения к занятию 6.

1. Вычитание одиночных регистров.

В этом упражнении при вычитании числа 1260 из числа 8740 используются только одиночные регистры. При этом операнды размещены в ячейках памяти. Этот способ вычитания применим к числам любой длины.

2. Вычитание регистровых пар.

В этом упражнении обращайте внимание на то, что если в результате выполнения предыдущей операции флаг C был взведен, эмулятор пропустит следующую команду ADS или SBC. Старайтесь не допускать таких ошибок.

З А Н Я Т И Е 7

ИНКРЕМЕНТ И ДЕКРЕМЕНТ

Предыдущие арифметические команды работали с двумя регистрами или парами регистров. В отличие от них команды INC инкремент) и DEC (декремент) работают с одним регистром или ячейкой ОЗУ, адресуемой через HL. Команда INC прибавляет единицу к содержимому указанного регистра или ячейки памяти, а команда DEC - отнимает единицу. В результате выполнения этих команд флаг C не изменяется.

В основном, эти команды используются при работе с указателями. Операции с регистровыми парами, которыми мы будем пользоваться в дальнейшем, широко используют эти команды для организации последовательной косвенной адресации ячеек памяти.

Подведем итог результатам занятия.

INC r	DEC r
INC (HL)	DEC (HL)
INC dd	DEC dd

З А Н Я Т И Е 8

ФЛАГ НУЛЯ

Очень часто используется другой флаг состояния процессора, называемый флагом нуля или флагом Z (zero). Этот флаг взводится, если результат однорегистровой арифметической операции равен нулю. Во всех остальных случаях флаг Z сброшен. Результат арифметической операции с двойными регистрами воздействует на флаг Z в тех же случаях, в которых они воздействуют на флаг C, т.е. только в результате выполнения команд вида ADC HL,dd или SBC HL,dd.

На флаг Z, как и на флаг C не воздействуют команды LD и EX.

Подведем итог результатам занятия.

КОМАНДА	ФЛАГ C		ФЛАГ Z	
	r	dd	r	dd
LD	–	–	–	–
EX	N	–	N	–
ADD	+	+	+	–
ADC	+	+	+	+
SUB	+	N	+	N
SBC	+	+	+	+
INC	–	–	+	–
DEC	–	–	+	–

r – одиночный регистр; dd – двойной регистр;
– – команда не оказывает воздействия на указанный флаг;
+ – команда оказывает воздействия на указанный флаг;
N – недопустимая команда.

Перед тем, как выполнить упражнение, попробуйте предсказать состояние флага Z после выполнения каждой команды.

З А Н Я Т И Е 9

СРАВНЕНИЕ

До сих пор мы рассматривали команды, которые, воздействуя на содержимое регистров воздействовали на состояние флагов C и Z. Однако, в системе команд есть и такие, которые не затрагивая операндов (чисел, участвующих в операции) воздействуют на флаги. Одна из них – это команда сравнения CP (compare).

Команда CP сравнивает содержимое аккумулятора с числом (CP n), содержимым регистра (CP r) или содержимым ячейки ОЗУ, адрес которой записан в HL (CP (HL)). Эта команда по своему механизму действия подобна команде вычитания SUB, с той разницей, что результат вычитания никуда не записывается.

Для иллюстрации рассмотрим наиболее простой случай –

сравнение содержимого аккумулятора с числом n. При этом возможны следующие варианты:

	ФЛАГ C	ФЛАГ Z
A > n	0	0
A = n	0	1
A < n	1	0

Подведем итог результатам занятия.

CP n – сравнение содержимого аккумулятора с числом n=0..255;

CP r – сравнение содержимых аккумулятора и регистра r;

CP (HL) – сравнение содержимых аккумулятора и ячейки ОЗУ по адресу, записанному в HL.

Упражнение к занятию 9.

Попробуйте предсказать состояние флагов C и Z после выполнения операции сравнения. На последующих занятиях Вы увидите, насколько обширно применяются эти флаги в ассемблерных программах, а также выполните упражнения с ними.

ЧАСТЬ ВТОРАЯ

ВВЕДЕНИЕ К ЧАСТИ ВТОРОЙ

На занятиях части второй Вы изучите команды, применение которых позволяет использовать компьютер не как простой вычислитель, а как систему, способную принимать самостоятельные решения в зависимости от ситуации в программе. При этом Вы должны во всех подробностях понимать работу команд, изученных ранее.

ЗАНЯТИЕ 10

УСЛОВНЫЕ И БЕЗУСЛОВНЫЕ ПЕРЕХОДЫ

Одним из преимуществ ЭВМ перед калькулятором является его способность изменять последовательность действий в зависимости от результата выполнения предыдущих команд. В МП Z80 имеется специальный регистр, называемый программным счетчиком, и обозначаемый сокращенно PC (program counter).

Содержимое PC является адресом ячейки ОЗУ, в которой находится команда, исполняемая процессором на следующем шаге. После выборки команды содержимое PC автоматически увеличивается таким образом, чтобы он указывал на следующую команду. Очевидно, изменить последовательность выполнения команд можно, принудительно изменив содержимое PC. Для этого в системе команд имеются команды переходов.

Наиболее простая из них – команда безусловного перехода JP nn, где nn – адрес команды, на которую совершается переход. При выполнении команды JP nn адрес nn засылается в PC.

Более сложными командами являются команды условных переходов. Перед условным переходом процессор проверяет состояние флагов на предмет выполнения того или иного условия. Условные переходы, использующие флаги C и Z:

JP NC,nn - переход, если флаг C сброшен;
JP C,nn - переход, если флаг C взведен;
JP NZ,nn - переход, если флаг Z сброшен;
JP Z,nn - переход, если флаг Z взведен.

По команде JP (HL) осуществляется переход к команде, находящейся по адресу, записанному в паре HL, иными словами, содержимое HL засылается в программный счетчик. Такой переход называется косвенным, однако, если содержимое HL является результатом вычислений, то такой переход можно также отнести к условным.

Подведен итог результатам занятия.

```
JP nn-----\
           > - безусловные переходы;
JP (HL)---/
JP NC,nn - переход по условию C=0;
JP C,nn  - переход по условию C=1;
JP NZ,nn - переход по условию Z=0;
JP Z,nn  - переход по условию Z=1.
```

Упражнение к занятию 10.

При работе с упражнением не разрешается указывать адрес перехода в виде числа, т.к. эмулятор не сможет выполнить такой переход. При необходимости в таком переходе, Вы можете использовать символические метки. Метка ставится перед командой, на которую Вы хотите передать управление. Для этой цели в упражнении использованы метки LOOP1 и LOOP2.

З А Н Я Т И Е 11

ОТНОСИТЕЛЬНЫЕ ПЕРЕХОДЫ

Одной из трудностей, связанной с употреблением команд перехода, является то, что для изменения стартового адреса программы (т.е. перемещения программы) необходимо, также, изменить все адреса, используемые в командах переходов. Иначе говоря, рассмотренные переходы привязаны к абсолютным адресам. Этот недостаток устранен в командах относительных переходов, в которых вообще не используются абсолютные адреса. Вместо них эти команды используют разность между содержимым PC и адресом команды, на которую необходимо осуществить переход (т.н.смещение). Однако, при употреблении команд относительных переходов, невозможно передать управление команде, находящейся по адресу меньше -128 (от текущего адреса) или больше +127 (от текущего адреса). Это объясняется тем, что для размещения величины смещения в этих командах отводится 8 разрядов с учетом знака.

Команды относительного перехода обозначаются JR (jump relativity), и записываются подобно командам JP, т.е. JR, JR NC, JR C, JR NZ и JR Z.

Особый случай относительного перехода представляет команда DJNZ (decrement and jump no zero) - декремент и переход, если Z=0. При выполнении этой команды из содержимого регистра B вычитается единица и результат сравнивается с нулем. Если

результат не равен нулю, то процессор выполняет относительный переход на исполнительный адрес, иначе выполняется следующая за DJNZ команда. Команда DJNZ очень удобна для организации циклов, т.е. последовательность действий, повторяющаяся заданное число раз. Количество повторений перед входом в цикл заносится в регистр B, который, таким образом, является счетчиком цикла.

Кроме команды DJNZ никакая команда относительного перехода не воздействует на флаги.

Подведем итог результатам занятия.

```
JR e----------\
JR NC,e------- \
JR C,e--------- >- e - смещение от текущего значения PC,
JR NZ,e------- /        может принимать значения -128..+127
JR Z,e--------/
DJNZ e---------- - декремент регистра B и переход при Z=0
```

Упражнения к занятию 11.

1.Упражнение показывает способ умножения с использованием относительных переходов. Существует более быстродействующие способы, которые будут показаны в дальнейшем.

2. По возможности, в ассемблерных программах стараются использовать относительные переходы. Эмулятор ассемблера, используемый в программе TUTOR не является в этом смысле исключением. Относительные переходы должны ссылаться на метки. В данном упражнении используется ячейка ОЗУ, находящаяся по адресу 23606. В Спектруме эта ячейка содержит системную переменную CHARS - число, на 256 меньшее адреса начала области знакогенератора. Программа упражнения записывает символы в верхнюю строку экрана. Вы можете проследить за результатами работы программы, удерживая клавишу нажатой, при этом программа выполняется быстрее.

Если Вы хотите модифицировать эту программу имейте в виду, что Спектрум работает с экраном, имеющим 32 символа в строке, а TUTOR - 42 символа.

З А Н Я Т И Е 12

СТЕК

Стек - это область ОЗУ, адресуемая особым образом и использующая для обращения специальный регистр, называемый указателем стека, и обозначаемый SP (stack pointer). SP состоит из 16 разрядов.

При записи в стек из любой внешней регистровой пары указатель стека уменьшает свое содержимое на 2 (SP всегда указывает на следующую пару ячеек ОЗУ). Иными словами, содержимое регистровой пары записывается в стек посредством косвенно-автодекрементной адресации через SP.

Для записи в стек используется команда PUSH dd. При выполнении этой команды содержимое регистровой пары dd засылается в основание ('дно' стека), а указатель стека после выполнения команды будет показывать, куда заслать следующее число.

Команда POP dd извлекает число из стека и записывает его в

регистровую пару dd, т.е. указанная регистровая пара загружается содержимым пары ячеек ОЗУ путем косвенно-автодекрементной адресации через регистр SP. После выполнения команды содержимое SP автоматически увеличивается на 2, т.е. SP указывает, откуда будет извлечено следующее число.

Таким образом, заслав в стек ряд чисел, в силу устройства стека, извлекаться из стека числа будут в порядке, обратном тому, в котором они были туда засланы (первым пришел - последним вышел).

Содержимое регистровых пар BC, DE и HL может быть заслано в стек беспрепятственно. Содержимое же аккумулятора может быть заслано в стек только в сочетании с регистром флагов F, который будет рассмотрен на последующих занятиях.

Указатель стека может участвовать в составе всех команд, рассмотренных ранее. В частности, команда EX (SP),HL меняет местами содержимые дна стека и регистровой пары HL.

Подведем итог результатам занятия.

```
PUSH dd          , где dd - регистровые пары AF, BC, DE, HL
POP dd
LD SP,nn
LD SP,(nn)
LD (nn),SP
LD SP,HL
ADD HL,SP
ADC HL,SP
SBC HL,SP
INC SP
DEC SP
EX (SP),HL
```

Упражнение к занятию 12.

Для 'ученических' программ эмулятор использует область памяти от 32000 до 32255. При этом изначально указатель стека установлен на адрес 32254, и при работе со своими программами, использующими стек, необходимо следить за тем, чтобы стек не доходил до области памяти, в которой размещена Ваша программа. Имейте в виду, что размер стека не контролируется эмулятором.

Последние 4 пары ячеек ОЗУ, используемые стеком, отображаются в нижнем правом углу экрана. Пара ячеек, на которые указывает SP выделены голубой полосой.

Команда INC SP в конце программы поставлена для того, чтобы исключить возможность сбоя указателя стека.

Область применения стека чрезвычайно широка. Некоторые примеры будут показаны в последующих упражнениях.

З А Н Я Т И Е 13

ВЫЗОВЫ ПОДПРОГРАММ

Подпрограммой называется участок программы, который может быть вызван из головной программы и после выполнения которой осуществляется возврат в головную программу. Подпрограмма может быть вызвана из любого места головной программы, другой

подпрограммы или этой же подпрограммы (т.н.рекурсия).

Для вызова подпрограммы в Z80 используется команда CALL. Она подобна по действию команде JP, с той разницей, что при выполнении команды CALL в стек засылается адрес команды, следующей за командой CALL. Заметим, что, как и при выполнении команды JP, в регистр PC засылается адрес первой команды вызываемой подпрограммы.

Для возврата из подпрограммы используется команда RET. При ее выполнении из стека извлекается ранее засланный туда адрес (называемый адресом возврата), который загружается в PC. При этом программа продолжает выполнение с того места, откуда была вызвана.

При работе в подпрограмме очень важно, чтобы каждой используемой команде PUSH или CALL соответствовала команда POP или RET. Это необходимо для того, чтобы состояния стека после входа в подпрограмму и перед выходом из нее были одинаковыми. В противном случае, при возврате в PC будет заслан неправильный адрес возврата, и произойдет потеря управления.

В МП Z80 различают условные и безусловные команды вызова подпрограмм и возврата из них. В условных командах могут использоваться те же условия, что и в командах условных переходов (C, NC, Z, NZ).

Подведем итог результатам занятия.

CALL NC,nn	RET NC
CALL C,nn	RET C
CALL NZ,nn	RET NZ
CALL Z,nn	RET Z

При отсутствии условия команды CALL и RET считаются безусловными

CALL nn	RET

Упражнения к занятию 13.

1. Безусловные вызовы и возвраты.

Здесь показан способ размещения подпрограмм, при котором одна подпрограмма вызывает другую. Необходимо помнить, что подпрограмма обращается к тем же регистрам, что и головная программа. Поэтому, чтобы не потерять содержимое регистров, при необходимости их использования рекомендуется засылать их содержимое в стек, а перед возвратом в головную программу - извлекать их из стека и возвращать на место.

2. Условные вызовы подпрограмм.

Эта программа подсчитывает количество чисел, больших 192 или меньших 64 находится в первых тридцати двух ячейках памяти.

З А Н Я Т И Е 14

ДВОИЧНАЯ ЗАПИСЬ ЧИСЕЛ

Если бы мы сняли с микросхемы Z80 корпус и посмотрели на регистр в мощный микроскоп, то увидели бы, что он состоит из восьми ячеек. Каждая ячейка способна принимать два устойчивых состояния. Условно одно устойчивое состояние считается нулем, а второе - единицей. Сочетание нулей и единиц в регистре может

быть любым, например таким: 1 0 1 1 1 0 0 0.

Это сочетание может интерпретироваться по-разному, в зависимости от ситуации: как машинная команда, как код символа, как число, знаковое или беззнаковое, и т.д. На этом занятии мы будем рассматривать сочетания нулей и единиц как беззнаковые числа, а в дальнейшем мы рассмотрим и остальные возможности.

Каждая цифра в числе, в зависимости от ее местоположения, обладает разным весом. Для привычной нам десятичной системы веса располагаются следующим образом:

```
   3     2     1     0
1000   100    10     1
```

Как видите, вес числа есть не что иное, как степень основания системы счисления (в данном случае - степень десятки). При этом в десятичной системе цифра может принимать значение 0..9. В регистре же каждая цифра (или 'бит') может принимать только два значения - 0 или 1. Поэтому такая система счисления называется двоичной (binary).

Веса в двоичной системе счисления распределены как степени двойки:

```
          бит:    7     6     5     4     3     2     1     0
десятичный вес:  128    64    32    16     8     4     2     1
```

Взяв наше число 10111000, умножим вес каждого бита на его значение. В результате мы получим десятичный эквивалент нашего двоичного числа:

1*128+0*64+1*32+1*16+1*8+0*4+0*2+0*1=184.

Очевидно, что максимальное число, которое можно записать в восьмиразрядный регистр равно 11111111, что в десятичной записи эквивалентно числу 255.

При сложении двух десятичных чисел, единица переносится в старший разряд, если сумма одного разряда оказывается больше 9. Такие же правила действуют и при сложении двоичных чисел, т.е. единица переносится в старший разряд, если сумма одного разряда превышает единицу.

Например:

```
  01101110
+
  00100101
----------
  10010011
```

Аналогично производится вычитание двоичных чисел:

```
  01100110
-
  00111001
--------
  00111001
```

Упражнение к занятию 14.

В эмуляторе предусмотрена директива BIN. При появлении этой директивы эмулятор резервирует одну ячейку памяти для записи в нее константы, следующей за директивой, при этом константа отображается в двоичном виде. При изменении содержимого ячейки памяти, изменения также будут отображаться.

З А Н Я Т И Е 15

ШЕСТНАДЦАТИРИЧНАЯ ЗАПИСЬ ЧИСЛА

Двоичные числа слишком громоздки, а десятичные - неудобны для отображения чисел, содержащихся в регистрах. Однако, существует еще один способ представления двоичных чисел, позволяющий разрешить эту проблему. Эта система счисления базируется на числе 16, и называется шестнадцатиричной, или, сокращенно HEX (hexadecimal).

Для представления первых десяти цифр в этой системе используются привычные нам цифры 0..9, а для представления остальных шести цифр условились использовать латинские буквы A..F. В приведенной ниже таблице показано соответствие между двоичными, десятичными и HEX-числами:

двоичная	десятичная	HEX
0000	00	0
0001	01	1
0010	02	2
0011	03	3
0100	04	4
0101	05	5
0110	06	6
0111	07	7
1000	08	8
1001	09	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

Хорошо видно, что все 16 состояний, представленные четырьмя битами, могут быть отображены одной HEX-цифрой. Очевидно, что двухразрядное HEX-число отображает состояние восьмиразрядного регистра. Например, если в регистр записано число 10111000, можно представить его как два четырехразрядных числа 1011 и 4000, которые эквивалентны HEX-цифрам B и 8, т.е. число 10111000 эквивалентно HEX-числу B8. Аналогично, состояние регистровой пары, например 1100100110111000 эквивалентно HEX-числу C2B8.

Преобразование HEX-числа в десятичное число может быть произведено обычным порядком, например:

B8 - B*16 + 8*1 = 11*16 + 8*1 = 184

Каждый программист должен твердо помнить основания различных систем счисления: HEX-система - основание 16, десятичная система - основание 10, двоичная - основание 2.

Кроме рассмотренных систем существуют малоупотребимая восьмиричная система (OCT-система) и двоично-десятичная система (BCD-система), которая будет рассмотрена на следующем занятии.

Упражнение к занятию 15.

Теперь, когда мы рассмотрели HEX-запись чисел, мы можем

более подробно рассмотреть вопрос, как программа хранится в памяти компьютера. Как уже упоминалось, команды хранятся в памяти в виде чисел. Все числа, относящиеся к одной команде, эмулятор отображает во второй колонке в одну строку, одно за другим, в HEX-виде. При этом операнд записывается последним. Собственно команда может условно разделяться на число, обозначающее тип команды, и число, обозначающее характер команды. Т.к. под обозначение типа команды отводится 8 бит, число различных команд не может превышать 256. Всего в МП Z80 имеется 158 различных команд, а сочетаний типов и характеров команд - 696.

В директивах DEFB или DEFW можно записывать HEX-числа, которые распознаются эмулятором по букве H в конце, например: 19H, 17AFH и т.п.

З А Н Я Т И Е 16

ДВОИЧНО-ДЕСЯТИЧНАЯ ЗАПИСЬ ЧИСЕЛ (BCD-формат)

Как было показано ранее, для того, чтобы преобразовать число из десятичного в двоичное или HEX-представление, необходимо произвести некоторый объем вычислений. HEX-запись чисел, однако имеет в себе ключ к разрешению этой проблемы.

При использовании этого метода десятичные числа записываются в двоичном коде поразрядно. При этом каждый разряд занимает 4 двоичных разряда - так называемую 'тетраду'. Например, число 18 записывается так: 0001 1000. Такой способ записи называется двоично-десятичной записью в BCD-формате. Недостатков у такого способа записи два. Первый состоит в том, что при его использовании в восьми разрядах можно записать число не более 99, тогда как в двоичном формате в тех же восьми разрядах можно записать число до 255. Второй недостаток заключается в необходимости отслеживания 'запрещенных состояний' тетрад (больших 9). Для борьбы с этим явлением в набор команд Z80 включена команда двоично-десятичной коррекции DAA. При ее выполнении проверяется состояние обоих тетрад аккумулятора, и, если число, записанное в тетраде превышает 9, к нему прибавляется число 6. В результате в старшем разряде корректируемой тетрады возникает единица переноса, соответствующая десятичному числу 16, что эквивалентно вычитанию 16 из данной тетрады. Иными словами, при результате, превышающем 9, в тетраде будет записано число, уменьшенное на 10 с переносом единицы в старшую тетраду, что и требуется.

Команда DAA всегда должна предшествовать команде сложения или вычитания BCD-чисел. Она воздействует на два, до сих пор неизвестных флага: H (флаг полупереноса, т.е. переноса из младшей тетрады в старшую) и N (флаг вычитания, показывающий, что последней выполненной командой было вычитание). Флаг N используется при коррекции BCD-чисел, т.к. коррекция различна для операций сложения и вычитания. Условными командами эти флаги не проверяются.

Упражнение к занятию 16.

Это упражнение показывает пример сложения и вычитания BCD-чисел. Помните, что отображаемые в HEX-виде числа на самом деле записаны в формате BCD.

З А Н Я Т И Е 17

ПРЯМОЙ И ОБРАТНЫЙ КОДЫ

До сих пор мы имели дело только с положительными числами. Но если мы попробуем вычесть 6 из 5, мы получим число, которое до сих пор считали числом 255, хотя должны получить число -1. В чем же дело? Рассмотрим этот вопрос подробнее.

Если мы вычитаем 6 из 5, то получаем число -1, которое в двоичной записи выглядит так: 11111111. Аналогично, 5 - 2 = 11111110 = -2. При этом седьмой бит называется 'знаковым' битом, он равен 1 для отрицательного числа и 0 для положительного. Но это еще не все объяснение. Действительно, если знак числа определяется его сёдьмым битом, то число -2 должно выглядеть так: 10000010. Дело в том, что в микропроцессоре отрицательные числа представлены в обратном дополнительном коде. Получить из положительного числа отрицательное можно инвертировав все его биты и прибавив единицу. Рассмотрим несколько примеров:

```
исходное число: 00000101 = +5
инвертируем:    11111010
прибавл. 1:     11111011 = -5
инвертируем:    00000100
прибавл. 1:     00000101 = +5
```

Это очень важный вопрос!

Очевидно, интерпретируя содержимое одиночного регистра как число со знаком, диапазон чисел, которые можно в него уместить: 01111111..10000000, или в десятичном виде +127..-128.

МП Z80 имеет две команды для выполнения рассмотренных операций:

CPL - побитная инверсия содержимого аккумулятора (дополнение до единицы);

NEG - побитная инверсия содержимого аккумулятора с прибавлением единицы (инверсия числа со знаком, или дополнение до двойки).

В регистре флагов Z80 имеется флаг S (sign), показывающий знак результата любой арифметической операции. Этот флаг взводится в случае отрицательного результата операции, и сбрасывается в случае положительного результата.

Для контроля переполнения при операциях со знаковыми числами служит флаг четности-нечетности P/V (parity/overflow). Он взводится в случае выхода результата операции за пределы -128..+127. Заметим, что применять для этой цели флаг C нельзя, т.к. он не отслеживает переполнение свыше +127 (при этом взводится 7-й бит, что вызовет ошибку в результате).

Рассмотренный способ представления чисел со знаком может аналогично применяться к регистровым парам. При этом в них могут размещаться числа от -32768 до +32767, знаковым считается 15-й

бит.

Флаги S и P/V действуют также после команд ADC и SBC, работающих с регистровой парой HL.

Очень важно помнить, что флаги Z, C, S и P/V затрагиваются восьмибитовыми арифметическими командами и шестнадцатибитовыми сложениями, включающими в себя перенос. Ваша интерпретация результатов операции должна базироваться на состоянии этих флагов.

Упражнение к занятию 17.

При выполнении упражнения Вы можете отличать положительные числа от отрицательных, наблюдая за флагами S и P/V. Кроме того, все числа, большие 127 - отрицательные, не забывайте это. Вы можете быстро преобразовать эти числа в знаковые в уме, вычитая из них 256. Например, число 254 - это отрицательное число, равно 254 - 256 = -2.

Ч А С Т Ь Т Р Е Т Ь Я

ВВЕДЕНИЕ К ЧАСТИ ТРЕТЬЕЙ

Эта часть занятий показывает принципы работы с отдельными битами и их использование во 'флаговых' операциях, а также показывает в упражнениях 'быстрые' способы умножения.

З А Н Я Т И Е 18

ЧЕТНОСТЬ

При передаче информации из одного места в другое всегда возможна ошибка. Проверка на четность - один из способов детектирования ошибок. Суть этого способа заключается в том, что в правильно принятом слове количество единиц должно быть четным числом. В противном случае считается, что слово принято с ошибкой. Существует также способ проверки на нечетность, когда слово считается принятым правильно, если количество единиц в нем нечетно, или, что тоже самое, если в нем четно количество нулей. Оба этих способа взаимно дополняют друг друга.

МП Z80 имеет флаг, взводящийся в случае, когда количество единиц в числе четное. Это флаг P/V, о котором мы уже упоминали. Например:

00110100 - в числе 3 единицы, P/V = 0
01000111 - в числе 4 единицы, P/V = 1

Необходимо различать интерпретацию состояния флага P/V в разных случаях. Во всех командах, работающих с данными, флаг P/V как флаг четности интерпретирует только команда DAA. Все другие команды, как пройденные, так и пока не изученные нами, либо не используют этот флаг, либо воспринимают его как флаг переполнения.

З А Н Я Т И Е 19

РЕГИСТР ФЛАГОВ И РЕГИСТРОВАЯ ПАРА AF

До сих пор мы рассматривали флаги как отдельные биты. На самом деле, эти биты объединены в восьмиразрядный регистр, называемый регистром флагов или регистром F. Соответствие между флагами и битами регистра F следующее:

бит	флаг	название
0	C	флаг переноса
1	N	флаг вычитания
2	P/V	флаг четности/переполнения
3	-	не используется
4	H	флаг полупереноса
5	-	не используется
6	Z	флаг нуля
7	S	флаг знака

Регистр F может объединяться с аккумулятором в регистровую пару AF. Эта регистровая пара может использоваться только в командах PUSH и POP.

З А Н Я Т И Е 20

ФЛАГИ S И P/V В КОМАНДАХ

Эти флаги могут использоваться во всех командах абсолютных (не относительных!) переходов, вызовах подпрограмм и возвратах, и записываются следующим образом:

```
JP PO,nn    CALL PO,nn    RET PO    - флаг P/V = 0
JP PE,nn    CALL PE,nn    RET PE    - флаг P/V = 1
JP P,nn     CALL P,nn     RET P     - флаг S   = 0
JP M,nn     CALL M,nn     RET P     - флаг S   = 1
```

Если указанное условие не выполняется, то переход, вызов подпрограммы или возврат не осуществляется.

Упражнение к занятию 20.

Данное упражнение иллюстрирует занятие примерами условных переходов по положительному и отрицательному результатам JP P,nn и JP M,nn ('переход, если плюс' и 'переход, если минус'). Условия, использующие флаг C были бы более действенны, но если бы перед переходом использовали команды ADC и SBC, или иными способами сбросили флаг C, то результаты перехода могут быть неожиданными.

З А Н Я Т И Е 21

РАБОТА С ОТДЕЛЬНЫМИ БИТАМИ

До сих пор мы обращались к байту или шестнадцатиразрядному слову как к единому целому. Z80 имеет в наборе команд команды,

позволяющие производить операции с отдельными разрядами, не затрагивая при этом соседних разрядов.

Любой бит в регистре или ячейке ОЗУ, адресуемой через (HL) может быть взведен или сброшен командами SET и RES соответственно. Проверку состояния отдельного бита мы можем осуществить командой BIT. При выполнении такой проверки флаг Z взводится, если проверяемый бит равен 0, и сбрасывается в противном случае.

Теперь мы можем в своих программах вырабатывать свои собственные флаги, предназначенные для наших нужд.

Подведем итог результатам занятия.

```
SET N,r      N - номер бита (0..7)
SET N,(HL)
RES N,r
RES N,(HL)
BIT N,r
BIT N,(HL)
```

Упражнение к занятию 21.

Упражнение показывает примеры манипуляции отдельными битами с помощью команд SET, RES и BIT в регистрах и ячейках памяти.

З А Н Я Т И Е 22

ЛОГИЧЕСКИЕ КОМАНДЫ

МП Z80 имеет в системе команд три команды логических операций. Все они выполняются побитно между содержимым аккумулятора и некоторым числом, содержимым регистра или ячейки ОЗУ, адресуемой через (HL). Результат помещается в аккумулятор.

Команда AND выполняет операцию логического 'И' (логического умножения) между битами операндов. При этом бит результата равен единице только тогда, когда оба соответствующих бита операндов также равны единице. Например:

```
   01101100
AND
   01010110
   --------
   01000100
```

С помощью этой команды можно выборочно сбрасывать группы битов в аккумуляторе. Иногда команду AND называют командой сброса битов по маске, в роли маски при этом выступает второй операнд.

Команда OR выполняет операцию логического 'ИЛИ' (логического сложения) между битами операндов. При этом бит результата равен единице, если хотя бы один из соответствующих битов операндов равен единице. Например:

```
   01101100
OR
   01010110
   --------
   01111110
```

С помощью этой команды можно выборочно вводить группы битов в аккумуляторе.

Команда XOR выполняет логическую операцию 'исключающее ИЛИ' (сложение по модулю 2) между битами операдов. При этом, в случае, когда соответствующие биты операндов одновременно равны единице, бит результата принимает значение 0. В остальных случаях эта команда действует аналогично команде OR. Например:

```
   01101100
XOR
   01010110
   --------
   00111010
```

С помощью этой команды можно инвертировать отдельные биты или группу битов.

Рассмотренные команды воздействуют на флаги Z, S и P/V. Во всех случаях выполнения этих команд флаг C=0. Это необходимо иметь в виду, т.к. это ограничивает в некоторых случаях применение логических команд.

Команды AND A и OR A, сбрасывая флаг C, не воздействуют на содержимое аккумулятора. Команда XOR A сбрасывает флаг C и очищает аккумулятор.

Подведем итог результатам занятия.

AND n	OR n	XOR n
AND r	OR r	XOR r
AND (HL)	OR (HL)	XOR (HL)

Упражнение к занятию 22.

Упражнение иллюстрирует случай применения логических команд. Внимательно следите за модификациями содержимого аккумулятора при их выполнении.

ЗАНЯТИЕ 23

КОМАНДЫ СДВИГА

Команды сдвига позволяют сдвигать содержимое регистра или ячейки ОЗУ на один бит вправо или влево. Как мы увидим, это эквивалентно делению или умножению исходного числа.

Команда SRA r и SRA (HL) сдвигают содержимое регистра или ячейки ОЗУ вправо. При этом нулевой бит копируется в флаг C, состояние седьмого бита остается неизменным. Примеры:

биты	7	6	5	4	3	2	1	0	C
до сдвига	0	1	1	0	0	0	1	0	?(некоторое значение)
после	0	0	1	1	0	0	0	1	0(значение 0-го бита)

До сдвига число было равно 98 (десятичное), а после сдвига стало равным 49 (десятичное), т.е. было совершено деление на 2.

биты	7	6	5	4	3	2	1	0	C
до сдвига	1	1	1	0	0	0	1	0	?(число -30 десятичн.)
после	1	1	1	1	0	0	0	1	0(значение 0-го бита)

Таким образом видно, что команда SRA выполняет деление положительного или отрицательного числа на 2. Команда SRA называется командой арифметического сдвига вправо 1.

Команды SRL r и SRL (HL) сдвигают содержимое регистра

или ячейки ОЗУ вправо, при этом седьмой бит сбрасывается, нулевой бит копируется в флаг С. Например:

	биты	7	6	5	4	3	2	1	0	C	
до сдвига		1	0	1	1	0	1	1	1	?	(число 183 десятичн)
после		0	1	0	1	1	0	1	1	1	(число 91 десятичн.)

Команда SRL r выполняет деление на 2 положительных или беззнаковых чисел и называется командой логического сдвига 1 0 вправо. После ее выполнения флаг С указывает на наличие остатка от деления.

Команды SLA r и SLA (HL) сдвигают содержимое регистра или ячейки ОЗУ на 1 бит влево, при этом седьмой бит копируется в флаг С, а нулевой бит сбрасывается. Например:

	биты	C	7	6	5	4	3	2	1	0	
до сдвига		?	0	1	1	0	0	0	1	0	(число 98 десятичн.)
после		0	1	1	0	0	0	1	0	0	(число 196 десятичн.)

Команда SLA называется командой арифметического сдвига влево. Она выполняет умножение положительного или отрицательного числа на 2, при этом флаг С указывает на то, что результат превышает 255.

Все эти команды воздействуют не только на флаг С, но и на флаги Z, S и P/V.

Подведен итог результатам занятия.

```
SRA r       - деление положительного или отрицательного числа
              на 2
SRA (HL)\
SRL r     >- деление положительного числа 0..255 на 2
SRL (HL)/
SLA r       - умножение положительного или отрицательного
              числа на 2
SLA (HL)
```

Упражнение к занятию 23.

Команды SRA, SRL, и SLA в этом упражнении воздействуют на содержимое ячейки памяти, имеющей метку STORE.

З А Н Я Т И Е 24

КОМАНДЫ ЦИКЛИЧЕСКОГО СДВИГА

Команды циклического сдвига действуют подобно уже рассмотренным командам сдвига, с той разницей, что при их выполнении бит, записанный на одном конце регистра (или ячейки ОЗУ перемещается на другой конец. Флаг С при этом может принимать значение этого бита либо может рассматриваться как девятый бит, и перемещаться вместе с остальными битами.

Команды RLC r и RLC (HL) выполняют циклический сдвиг влево, при этом седьмой бит перемещается на место нулевого бита, а флаг С принимает его значение. Например:

	биты	C	7	6	5	4	3	2	1	0	
до сдвига		?	1	0	1	1	0	0	0	1	
после		1	0	1	1	0	0	0	1	1	(бывший 7-й бит)

Команды RL r и RL (HL) выполняют циклический сдвиг влево вместе с флагом С. Например:

биты С	7	6	5	4	3	2	1	0
до сдвига с	1	0	1	1	0	0	0	1
после i	0	1	1	0	0	0	1	с

Команда RL может использоваться для умножения числа любой длина на 2. Однако, при этом между повторениями команды RL необходимо восстанавливать флаг С.

Команды RRC r и RRC (HL) аналогичны команде RLC с той разницей, что сдвиг выполняется вправо, бит 0 переносится на место бита 7, а флаг С принимает его значение. Например:

биты 7	6	5	4	3	2	1	0	С
до сдвига 1	0	1	1	0	0	0	1	?
после 1	1	0	1	1	0	0	0	1(бывший 0-й бит)

Команды RR r и RR (HL) действуют аналогично командам RL с той разницей, что сдвиг выполняется вправо. Например:

биты 7	6	5	4	3	2	1	0	С
до сдвига 1	0	1	1	0	0	0	1	с
после с	1	0	1	1	0	0	0	1

Команда RR может использоваться для деления числа любой длина на 2.

Команды RRC и RLC можно использовать для последовательного опроса битов содержимого регистров или ячейки памяти без риска его испортить.

Как и команды сдвига, команды RLC, RL, RRC и RR воздействуют на флаги С, Z, S, и P/V.

К командам циклического сдвига относятся еще четыре команды: RLCA, RLA, RRCA, RRA. Эти команды воздействуют только на аккумулятор. Они полностью идентичны командам RLC A, RL A, RRC A и RR A, но во-первых, воздействуют только на флаг С, а во-вторых, выполняются в два раза быстрее.

Подведем итог результатам занятия.

RLC r — циклический сдвиг содержимого r влево, флаг С дублируется.

RLC (HL)

RLCA — циклический сдвиг содержимого аккумулятора влево, флаг С дублируется.

RL r — циклический сдвиг содержимого r вместе с флагом С влево.

RL (HL)

RLA — циклический сдвиг содержимого аккумулятора вместе с флагом С влево.

RRC r — циклический сдвиг содержимого r вправо, флаг С дублируется.

RRC (HL)

RRCA — циклический сдвиг содержимого аккумулятора вправо, флаг С дублируется.

RR r — циклический сдвиг содержимого r вместе с флагом С вправо.

RR (HL) — Циклический сдвиг содержимого аккумулятора

RRA вместе с флагом С вправо.

Упражнения к занятию 24.

1. Команды циклического сдвига модифицируют число, записанное в ячейку с меткой STORE с последующим исследованием

результата.

2. Эта программа исследует команды арифметического и логического сдвигов и циклического сдвига для перемножения восьмиразрядных двоичных чисел. Второе число сдвигается арифметически, и числа складываются, если исследуемый бит выведен. Умножение по такому алгоритму производится быстрее, чем умножение путем последовательных сложений.

З А Н Я Т И Е 25

ДЕСЯТИЧНЫЙ ЦИКЛИЧЕСКИЙ СДВИГ

Z80 позволяет нам циклически сдвигать не только отдельные биты, но и целые тетрады. Для этого существуют команды:

RLD - циклический сдвиг BCD-числа влево.

RRD - циклический сдвиг BCD-числа вправо.

Эти команды используют аккумулятор и ячейку ОЗУ, адресуемую через (HL). Они циклически сдвигают младшую тетраду аккумулятора, младшую тетраду ячейки ОЗУ и старшую тетраду этой ячейки. Например:

до сдвига по RLD

аккумулятор	ячейка ОЗУ
1011 1101	0101 1100

после сдвига по RLD

аккумулятор	ячейка ОЗУ
1011 0101	1100 1101

Команды RRD и RLD могут использоваться для деления и умножения BCD-чисел на 10 соответственно, или для выбора меньшего числа из двух.

Упражнение к занятию 25.

Упражнение показывает пример умножения и деления числа в BCD-формате на 10.

Ч А С Т Ь Ч Е Т В Е Р Т А Я

ВВЕДЕНИЕ К ЧАСТИ ЧЕТВЕРТОЙ

Заключительная часть занятий посвящена специальным регистрам, а также командам, с помощью которых Z80 может взаимодействовать с внешним миром.

З А Н Я Т И Е 26

ИНДЕКСНЫЕ РЕГИСТРЫ

Рассмотрим два новых регистра - индексные регистры IX и IY. Они во всех отношениях идентичны. Индексные регистры в большинстве команд могут заменять регистр HL, кроме команд ADC HL,dd, SBC HL,dd и EX DE,HL. Разумеется, они не могут заменять HL в командах, использующих его всегда (например, в команде RRD).

Большое преимущество индексных регистров проявляется в

командах косвенной адресации. В них вместо (HL) записывается (IX+d) или (IY+d), при этом адресация осуществляется по адресу, являющемуся суммой содержимого IX или IY и числа d, находящегося в пределах 0..255. Очень удобно использовать такие команды для опроса таблиц. При этом смещение d задает столбец, а содержимое индексного регистра - строку. Подпрограммы, манипулирующие строковыми данными, как правило, используют индексные регистры, при этом IX и IY могут указывать на разные строки, что облегчает работу.

Использование смещения запрещается в команде косвенного перехода по индексному регистру JP (IX) и JP (IY).

Подведем итог результатам занятия.

```
LD r,(IX+d)     LD IX,nn      LD SP,IX
LD (IX+d),r     LD IX,(nn)
LD (IX+d),n     LD (nn),IX    EX (SP),IX
ADD A,(IX+d)    INC (IX+d)    AND (IX+d)
ADC A,(IX+d)    DEC (IX+d)    OR  (IX+d)
SUB A,(IX+d)                  XOR (IX+d)
SBC A,(IX+d)    CP  (IX+d)
ADD IX,dd       INC IX        DEC IX
SLA (IX+d)      SRA (IX+d)    SRL (IX+d)
RLC (IX+d)      RL  (IX+d)    RRC (IX+d)
                RR  (IX+d)
SET N,(IX+d)    RES N,(IX+d)  BIT N,(IX+d)
                JP  (IX)
```

Аналогично перечисленные команды могут использовать регистр IY.

Упражнение к занятию 26.

В упражнении регистр IX используется как указатель таблицы двухбайтовых чисел. Каждое число, в результате, будет поделено пополам.

З А Н Я Т И Е 27

АЛЬТЕРНАТИВНЫЕ РЕГИСТРЫ

Z80 имеет в своем составе набор регистровых пар, альтернативных изученным парам AF, BC, DE и HL. Альтернативные регистровые пары обозначаются AF', BC', DE' и HL' соответственно. Эти регистры не могут участвовать в большинстве команд, но могут использоваться для оперативного запоминания. Работу с альтернативными регистрами поддерживают две команды:

EX AF,AF' - обмен содержимыми регистровыми парами AF и AF'.

EXX - обмен содержимыми между регистровыми парами BC, DE, HL и BC', DE', HL' соответственно.

Упражнение к занятию 27.

Программа этого упражнения загружает все регистровые пары и обменивает их содержимое с содержимым альтернативных регистровых пар, с последующей перезагрузкой.

З А Н Я Т И Е 28

КОМАНДЫ ВВОДА-ВЫВОДА

Пока что мы пересылали данные только между процессором и памятью. Наш эмулятор, например, может записать какую-нибудь информацию на экран монитора, обращаясь к ячейкам ОЗУ по адресам с 4000H по 5AFFH. Однако, тот же результат можно получить, обращаясь к порту ввода-вывода. Z80 имеет 256 различных портов ввода и столько же портов вывода, работающих с восьмиразрядными данными.

Мы можем вводить данные непосредственно в аккумулятор, используя команду IN A,(n), где n - номер порта (0..255). Аналогично, мы можем вывести информацию из аккумулятора в порт n командой OUT (n),A. Эти команды не воздействуют на флаги.

Мы, также, можем ввести данные в любой восьмибитовый регистр, используя в команде регистры B и C. Команда IN r,(C) передает содержимое регистра B в порт, номер которого записан в регистре C, после чего загружает данные из этого же порта в регистр r. Например, в Спектруме эта команда используется для чтения данных с клавиатуры. Команда IN r,(C) воздействует на флаги Z, S и P/V. Команда IN F,(C) имеет особенность: при ее исполнении данные просто читаются из порта в регистр флагов, а данные в порт не выдаются.

Команда OUT (C),r действует подобно команде IN r,(C), но содержимое регистра r передается в порт (C).

Подведем итог результатам занятия.

```
IN A,(n)    n - номер порта (0..255)
IN r,(C)
IN F,(C)
IN F,(n)
OUT (n),A
OUT (C),r
```

Упражнение к занятию 28.

В Спектруме очень важную задачу выполняет порт ввода 254 (FEH), который используется при вводе с клавиатуры (биты 0..4) и при вводе с магнитофона.

Порт вывода 254 используется для установки цвета бордюра (биты 0..2) и вывода на магнитофон (бит 6).

При работе программа этого упражнения в цикле опрашивает клавиши между 'H' и 'ENTER', вводя их коды при нажатии. Полученный код выводится в порт 254.

З А Н Я Т И Е 29

БЛОКОВЫЕ КОМАНДЫ (ВСТУПЛЕНИЕ)

В системе команд имеется четыре группы из четырех типов команд, выполняющих операции с блоками ОЗУ. Все блоковые команды используют регистр (регистры) как указатель (указатели) для доступа к блокам памяти и как счетчики.

Все блоковые команды имеют похожую мнемонику. Буква I

показывает, что указатель(и) инкрементируется, буква D - что декрементируется, а буква R показывает, что команду необходимо повторять до обращения содержимого счетчика в нуль.

команды загрузки LD .....LDI,LDIR,LDD,LDDR
сравнение CP .....CPI,CPIR,CPD,CPDR
ввод IN .....INI,INIR,IND,INDR
вывод OUT .....OUTI,OTIR,OUTD,OTDR

Буква U в команде OUT отбрасывается потому, что мнемоника команды не может состоять более, чем из четырех символов.

З А Н Я Т И Е 30

БЛОКОВЫЕ КОМАНДЫ ЗАГРУЗКИ

Блоковые команды загрузки позволяют пересылать содержимое из одной области памяти в другую. При этом используется два указателя: HL содержит адрес начала области, подлежащей пересылке, а DE - адрес начала области размещения пересылаемой области. Размер пересылаемой области в байтах указывается в BC.

Команда LDI пересылает содержимое ячейки ОЗУ по адресу (HL) в ячейку по адресу (DE), затем инкрементирует оба указателя и декрементирует BC. Если содержимое BC равно нулю, то флаг P/V=0, иначе этот флаг взводится.

Упражнение к занятию 30.

Упражнение использует команду LDIR для передачи верхней трети экрана в середину экрана. Эта операция хорошо иллюстрирует мощность команды.

Рассмотренные команды могут, также, использоваться для заполнения блока памяти некоторым числом.

З А Н Я Т И Е 31

БЛОКОВЫЕ КОМАНДЫ СРАВНЕНИЯ/ПОИСКА

Блоковые команды сравнения/поиска предназначены для сравнения чисел, содержащихся в некоторой области памяти, с заданным числом. При этом HL служит указателем, а BC - счетчиком, содержащим длину проверяемого блока. Сравниваемое число при этом содержится в аккумуляторе. Если при проверке блока встречается такое же число, то флаг Z взводится. Флаг P/V сбрасывается при обнулении BC.

Когда блоковые команды останавливаются, указатель (указатели) содержат не адрес останова, а следующий адрес. Поэтому в данном случае, после останова указатель будет показывать на число, следующее за найденным.

Подведем итог результатам занятия.

CPI - указатели инкрементируются.
CPIR - то же, с повторением, пока BC не равно 0.
CPD - указатели декрементируются.
CPDR - то же, с повторением, пока BC не равно 0.

Упражнение к занятию 31.

Эта программа отыскивает на цветном экране первый синий

или белый символ. Во втором случае поиск произойдет быстрее.

З А Н Я Т И Е 32

БЛОКОВЫЕ КОМАНДЫ ВВОДА-ВЫВОДА

Группа блоковых команд IN вводит из порта ввода, определяемого содержимым регистра C в блок памяти, начинающийся с адреса, записанного в HL и длиной, записанной в регистре B. Все формы записи блоковых команд IN аналогичны по характеру действия другим блоковым командам.

INI - инкремент
INIR - то же, с повторением
IND - декремент
INDR - то же, с повторением

Группа блоковых команд OUT выводит данные из блока памяти, начинающегося с адреса (HL) и длиной B в порт вывода (C).

OUTI - инкремент
OTIR - то же, с повторением
OUTD - декремент
OTDR - то же, с повторением

Флаг Z во всех случаях взводится при обращении счетчика B в нуль.

Упражнение к занятию 32.

Так как повторение блоковой команды ввода происходит быстро, лучше одну из клавиш держать нажатой. Используйте для пробы разные клавиши.

З А Н Я Т И Е 33

КОМАНДЫ УПРАВЛЕНИЯ ПРОЦЕССОРОМ

Эта группа команд, вместе с командами прерываний, управляет состоянием процессора.

Команда NOP (no operations) заставляет процессор увеличить содержимое PC на единицу, не совершая больше ничего. Так как код этой команды - 00, очищенная область памяти, которую может встретить процессор, будет им пропущена. Эта команда часто используется для организации задержек.

Команда HALT останавливает выполнение процессором программы до тех пор, пока на процессор не поступит запрос прерывания (см. следующее занятие) или не будет произведен системный сброс по сигналу RESET. После прерывания процессор продолжит выполнение программы, начиная с команды, следующей за командой HALT. Таким образом может быть обеспечена максимально быстрая реакция процессора на прерывание.

В Z80 имеется группа команд RST n, сходная с командой CALL. Разница между ними в том, что команды RST вызывают подпрограммы, находящиеся по фиксированным адресам ('точкам входа'). Адреса n=00H, 08H, 10H, 18H, 20H, 28H, 30H или 38H. Иными словами, команда RST 28H соответствует команде CALL 0028H. Эти команды используются, в основном, для обработки прерываний,

и подробнее будут рассмотрены на следующем занятии.

В Спектруме все точки входа по командам RST расположены в ПЗУ. Так, например, команда RST 00H совершает перезапуск Спектрума.

Теперь рассмотрим два новых регистра Z80 - регистр регенерации R и регистр прерываний I. Регистр прерываний отложим на следующее занятие. Регистр R используется Z80 для регенерации динамического ОЗУ, т.е. ОЗУ, запоминающего информацию в виде заряда на внутренних конденсаторах. Т.к. конденсаторы постепенно разряжаются, необходимо периодически восстанавливать их заряд, т.е. производить регенерацию. Регистр R при каждом цикле регенерации инкрементируется, что может использоваться для получения случайных чисел.

Регистры R и I могут быть загружены из аккумулятора. Для этого используются команды LD A,R; LD R,A; LD A,I; LD I,A.

Подведем итог результатам занятия.

```
NOP
HALT
RST n, n=00H,08H,10H,18H,20H,28H,30H,38H
LD A,R        LD A,I;
LD R,A        LD I,A.
```

Упражнение к занятию 33.

Упражнение показывает пример использования регистра R для получения последовательности случайных чисел.

З А Н Я Т И Е 34

ПРЕРЫВАНИЯ

Прерывания поступают к процессору извне, и используются для прерывания текущей последовательности действий и перехода к выполнению другой задачи.

Прерывания делятся на немаскируемые (NMI) и маскируемые (INT). Маскируемое прерывание может быть игнорировано, если в программе в данный момент прерывания запрещены. Немаскируемое прерывание не может быть проигнорировано.

В Z80 могут использоваться оба этих вида прерываний. Немаскируемое прерывание автоматически вызывает подпрограмму обслуживания, начинающуюся по адресу 0066H. После обслуживания прерывания в подпрограмме выполняется команда RETN (return from pop=masking interrupt) и управление передается прерванной программе на то место, где программа находилась в момент прерывания. Иными словами программа не 'замечает' прерывания, и после возврата работает так, словно прерывания не было вовсе.

Программист может 'заставить' программу не реагировать на маскируемые прерывания. В Z80 имеется триггер прерываний, от состояния которого зависит, будет прерывание обслужено или проигнорировано. Для управления этим триггером в Z80 имеются команды:

EI (enable interrupt) - разрешение прерываний
DI (disenable interrupt) - запрет прерываний

Z80 имеет три режима маскируемых прерываний. Режим 0

устанавливается после команды IM 0. При прерывании по режиму 0 прерывающее устройство (или контроллер прерываний) должно передать процессору однобайтовую команду RST n. После того, как процессор примет эту команду, он выполняет ее, передавая тем самым управление на адрес n. Таким образом, в зависимости от того, какое устройство запросило прерывание, программа обслуживания может быть разной, что очень удобно.

Режим 1 устанавливается после команды IM 1. При прерывании по режиму 1 процессор обрабатывает всякое маскируемое прерывание по одному и тому же адресу 0038H. Этот режим используется в ПЗУ Спектрума.

Режим 2 устанавливается командой IM 2, и является самым гибким из трех. При прерывании по этому режиму адрес программы обслуживания находится в некоторой ячейке ОЗУ. Адрес этой ячейки складывается из содержимого регистра прерываний I (старший байт) и прерывающего устройства, выдающего на шину данных младший байт. Такое прерывание называется векторным, а адрес этой ячейки ОЗУ - вектором прерываний.

Прерывание может происходить в любой момент времени. Для того, чтобы программа обслуживания прерывания не повредила содержимого регистров, в ее начало рекомендуется помещать команды засылки содержимого всех регистров в стек, а перед возвратом - команды восстановления содержимого регистров из стека. Перед засылкой (извлечением) содержимого регистров в стек (из стека) необходимо ставить команду DI, запрещающую новые прерывания.

Из программы обслуживания маскируемых прерываний выход осуществляется командой RETI.

Обычно программа обслуживания прерываний имеет такой вид:

- запрет маскируемых прерываний;
- засылка в стек содержимого регистров;
- разрешение маскируемых прерываний;
- обработка текущего прерывания;
- запрет маскируемых прерываний;
- восстановление содержимого регистров;
- разрешение маскируемых прерываний;
- выход из прерывания.