ИГРОВЫЕ ПРОГРАММЫ

ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

ЗАМЕЧЕННЫЕ ОПЕЧАТКИ К КНИГЕ "ОПИСАНИЕ ПРОГРАММ НА КАССЕТЕ 1"

(КНИГА 1)

СТР.	СТРОКА	НАПЕЧАТАНО	СЛЕДУЕТ ЧИТАТЬ
9	8 снизу 7 снизу	G – пуск A,P – объём рычагов	O – пуск Q,P – подъем рычагов
30	16 сверху	BREAK более 1 сек	BREAK и CS более 1 сек
31	23 снизу	ENDPROG	END PROG
37	13 сверху 12 снизу 6 снизу 4 снизу	RAMD USR ее – EPASE RAMD USR BWSICe	RAND USR ее – ERASE RAND USR BASICe
38	1 сверху 2,3 сверху	BLOCK DELET номера до указанного	BLOCK DELETE номера

(КНИГА 2)

СТР.	СТРОКА	НАПЕЧАТАНО	СЛЕДУЕТ ЧИТАТЬ
3	8,10 сверху 16 снизу	RANDOMIZE USE CODE xxxxx11392	RANDOMIZE USR CODE xxxxx,11010
38	18 сверху 20 сверху 25 сверху 12 снизу	#80000 M:80000 #80C00 H:441472	#8000 M:8000 #8000 H:41472
39	11 сверху 25 сверху 26 сверху 8 снизу	J:80000 ... #80000 #90000 CALL 39000 #88C00	J:8000 ... #8000 #9000 CALL #9000 #8800

ОПИСАНИЕ ПРОГРАММ
НА КАССЕТЕ 1
(Книга 1)
(Книга 2)

GENS4 MONS4

НТК Системотехник 1990 г.

Содержание

Глава 1. Игровые программы

В этой главе содержится описание игровых программ, записанных на кассете 1 на строне А.

1. KRAKOUT - 2

Игры в "тенис " для одного участника. После появления исходной картинки нужно выбрать режим игры:

1 - бесконечная жизнь
2 - нормальная игра.

После нажатия клавиши 1 или 2, Вы выходите в основное меню игры.

0 - старт игры;
1 - клавиатура;
2 - джойстик;
3 - курсор джойстик;
4 - интерфейс II ;
o - установка параметров игры;

При работе с клавиатурой ракетка управляется следующим образом:

Q - вверх
A - вниз
CAPS SHIFT - подача мяча.

При нажатии клавиши O можно сменить параметры игры.

Расположение ракетки - справа (слева).
Звук - включен (выключен)
Скорость мяча - 1...6
Контроль магнитофона
Скорость ракетки - 1...9
Загрузка долнительных файлов
Q - выход в основное меню

2. COMMANDO

Цель игры: пройти 10 зон противника. Зоны разделены красными воротами. Ваш боец движется вверх по экрану дисплея уничтожая из автомата и гранатами противника. По пути нужно собирать ящики с боеприпасами.

После появления исходной картинки, Вы получаете возможность выбора управления (K - клавиатура, J - джойстик).

Нажмите "K" и выбирете клавиши удобные для управления бойцом:

вверх
вниз
влево
вправо
огонь
граната

При выборе джойстика Вы попадаете в меню:

1 - KEMPSTON
2 - INTERFES II
3 - FULLER
4 - CURSOR

В Вашем компьютере возможно управление KEMPSTON и CURSOR.

После выбора управления нажимайте "S" и не теряйте времени даром.

3. DEAT STAR

(звезда смерти)

Вам нужно захватить звезду смерти, но сначала успешно взлететь. При взлете ваш звездолет должен попасть в центр пятиугольника.

Клавиши управления:

M - миссия;
L - старт игры;
S - уровень сложности;
1 - контроль с клавиатуры;
2 - синклер джойстик;
3 - .....джойстик;
4 - курсор джойстик;

При выборе контроля с клавиатуры Вы управляете клавишами:

Q - вверх;
A - вниз;
P - вправо;
O - влево;
SYMBOL SHIFT - огонь.

4. JO

После появления исходной картинки и нажатия клавиши ENTER, Вы получаете возможность управления птичкой, которой нужно накормить своих птенцов червячками и мухами. За это Вам компьютер начисляет очки.

Управление птичкой осуществляется джойстиком и клавиатурой. Клавиши управления:

6 - влево;
9 - вверх;
7 - вправо;
8 - вниз;
0 - огонь.

5. S-CHESS-3

(шахматы)

Загрузка программы: LOAD "" .

После загрузки программы на экране появляется изображение меню:

1. Новая игра
2. Анализ позиции
3. Выбор цвета дисплея
4. Функции клавиши
5. Техническая информация

Нажатием клавиш "1...5" выбирается режим игры.

Нажмите клавишу "1" для начала игры.

Запрашивается уровень, на котором хочет играть игрок. Выбирете любую из клавиш от 0 до 9. Для быстрого ответа достаточно одного из низких уровней - от 0 до 3.

Затем игроку задается вопрос - каким цветом фигур он будет играть?

Нажмите "W" для игры белыми фигурами или "B" - для игры черными.

Позиция фигур на доске определяется использованием современной алгебраической системой условных знаков. Поле обозначено цифрами от 0 до 8 и буквами от "A" до "H". Белая ладья находится в начальном положении на поле A1. Поля обозначаются на экране. При выполнении хода нажмите на клавиши, соответствующие координатам (начать буквенными, затем цифровые клавиши). Сначала даются координаты местоположения фигуры, затем координаты поля, которое она занимает. Никаких разделителей между начальными и конечными координатами нет, все происходит автоматически. После того, как Ваш ход сообщен машине, необходимая фигура будет передвинута и компьютер примет решение и покажет свой ход. Разрешены все соответствующие правилам ходы, включая рокировку и первый ход пешки на два поля. Программа не разрешает применение ходов, не соответствующих правилам. Рокировка достигается путем передвижения короля.

Сделав ход и заметив свою ошибку, Вы всегда можете исправить ее, используя пожвижную или неподвижную зачеркивающую клавишу. Ряд команд может быть применен на любом

этапе. Нажимая на клавишу "M", вы можете спросить у машины, какой ход Вам следует сделать. Она сделает этот ход на том же уровне, на каком ведется игра. Клавиша "L" позволяет Вам изменить уровень сложноти игры компьютера. Клавиша "Z" позволяет перенести ситуацию с экрана на печатающее устройство.

Клавиша "T" позволяет Вам перенести позицию, имеющуюся на доске, на кассету с тем, чтобы продолжить ее позже.

Клавиша "X" означает выход из игры для отдыха, либо для установки на доске другой позиции.

После нажатия клавиши 2 Вы можете установить на доске любую позицию и проанализировать ее.

6. SAI COMBAT

Авторам игры SAI COMBAT удалось, наряду с наглядной и четкой графикой создать и программно реализовать хорошее пособие для этого вида восточного единоборства.

Игра имеет управление с клавиатуры и джойстика. Нажимая, после появления на экране монитора исходной картинки, клавиши 1(2), J, K Вы попадаете в игровой режим с одним (двумя)участниками.Управление с джойстика или с клавиатуры.

Управление левым бойцом находится в левой половине буквенной клавиатуры.

Клавиши управления 2, 3, 4, Q, R, A, S, P и при нажатой CAPS SHIFT все перечисленные..

Управление вторым бойцом - в симметричной правой половине. Клавиши: 8, 9, 0, P, U, L, K, J и при нажатой SYMBOL SHIFT все перечисленные клавиши.

Каждый боец может проводить 16 различных ударов. Для изучения клавиши управления удобно начать с режима игры вдвоем и освоить все удары на втором неподвижном бойце.

7. HIGHEN (HIGHWAY ENCOUNTER)

(великий путь со столкновениями)

Цель игры пронести груз через 30 зон с различными препятствиями и установить его в центр квадрата в зоне 0.

В меню выводится сообщение:

1 - управление с клавиатуры;
2 - INTERFACE - 2;
3 - кемстон джойстик;
4 - PROTEK/AGF курсор дж.;
5 - информация;
6 - демонстрация игры;
7 - старт.

При управлении с клавиатуры клавиши управления:

1 - движение вперед;
Q - остановка;
P - вправо;
O - влево;
Z,M - огонь;
H - останов игры;
A+G - выход из игры.

Игра ограничена временем. Торопитесь.

8. THREE WEEK'S IN PARADISE

(три недели в раю)

Наш старый знакомый Волли, известный по играм "PIJAMARAMA" и "EVERYONES A WOLLY", вместе с женой Вильмой и сыном Гербертом на необитаемом острове попадают в лапы людоедов. Волли удается убежать из плена. Его задача - освободить семью, иначе людоеды их сожрут. Вторая задача, стоящая перед Волли - построить плот, на котором они могли бы уплыть с острова. Плот строится автоматически в процессе игры: каждое правильно выполненное задание - сегмент плота.

Освобождение Герберта.

Герберт варится в большом котле, который сторожат два больших льва. Внимание! Не подходите к ним близко, а то сьедят. Также обратите внимание на то, что хвост первого льва прибит колючкой к земле - помогите ему.

1. Заберите с почты (INDIAN POST) тапочки (FLIP FLOPS).

2. Возьмите ведро (CAN)

3. Уйдите с этими вещами к гейзеру через кухню и столовую (переходы вверх и вниз обозначаются двумя направленными друг на друга указателями) по ступеням направо, через поляну с домиком и машиной до поляны с гейзером.

4. Имея с собой тапочки и ведро, дерните за шнурок, включите гейзер, быстро подойдите к нему и наполните ведро.

5. С полным ведром, тапочками идите на пляж, где сидит краб.

6. Вылейте воду из ведра на краба (клавиша ENTER), после чего он отбросит одну клешню, которую нужно забрать.

7. Тапочки оставьте на берегу перед пляжем - они еще пригодятся.

8. С клешней краба нужно дойти до поляны, на которой варится Герберт и, стоя у правого льва, "использовать" клешню (клавиша "ENTER"). Лев начнет махать хвостом - тогда можно будет спокойно проходить мимо него.

9. Затем нужно забрать два перекрещенных полена, которые лежат перед крокодилом и отнести их в кузницу.

10. Через столовую, а затем по ступенькам налево идти до колодца. На первый камешек с левой стороны можно спрыгнуть.

11. Перепрыгнуть на колодец и забрать оттуда кузнечные меха (BELLOWE). Внимание! Не нажмите "ENTER" - можете провалиться в колодец.

12. Вернитесь в кузницу и разожгите огонь. После того как он загорится, задуйте его мехами, возьмите пепел с собой (кл. "ENTER").

13. Пепел и меха несите с собой до бога дождя, который стоит на поляне перед почтой.

14. Стоя перед богом дождя, используйте пепел. Божок начнет подпрыгивать, а висящая на небе туча передвинется вправо к пальмам и там остановится.

15. Нужно идти следом за тучей, из которой вылетает сверкающая молния. Внимание! Обгонять тучу нельзя - она вернется назад. Туча двигается только тогда, когда у вас есть с собой меха - если их выпустить, она остановится.

16. Туча дойдет до домика под крышей. Молния разобьет его и на развалинах останется только раковина. Оставьте меха и заберите раковину.

17. Идите с раковиной до колодца и прыгайте в него (кл. "ENTER").

18. На дне колодца падающими каплями воды наполните раковину. Заберите бутылку и по правой стене, используя клавишу "ENTER", выберетесь из колодца.

19. С полной раковиной идите к котлу, в котором людоеды варят Герберта и "опорожните" раковину.

20. После этого Герберт будет спасен. Он появится в нижней правой части экрана на месте маленького скелетика.

Спасение Вильмы.

Вильма подвешена на дереве, которое охраняет страхник. Осторожно! Не подходите к нему слишком близко, это опасно. Для освобождения Вильмы его нужно убить.

1. Для этого нужно заточить топор:

- имеющуюся бутылку отнести к крокодилу и там оставить;
- в кузнице забрать штопор и сумку Вильмы на пляже;
- обе вещи отнести к крокодилу (когда сумка с собой, крокодил неопасен), и поочередно (сначала штопор, затем бутылку), перенести к находящемуся за крокодилом кокосовому ореху;
- "выжать" из кокосового ореха масло (кл. "ENTER") в бутылку;
- выходя, иметь с собой топор и бутылку с маслом;
- оба предмета отнести на поляну, где стоит машина с квадратными колесами. Там нужно "использовать" оба предмета. Топор станет острым.

2. Пойти в столовую, забрать со стола миску с кашей и все нести к охраняющему Вильму стражнику. Оставить топор.

3. С миской каши пойти налево до поляны, где стоит огромный цыпленок.

4. Цыпленок снесет яйцо.

5. Забрать яйцо, отнести и оставить его у гейзера - оно еще пригодится.

6. Идти на почту и перед входом, за столом с указателем, взять ритуальную монету (она не видна).

7. С монетой идти к крокодилу, пройти мимо него (имея в руках сумку) на следующую поляну (замороженный лес), где лежит шестигранный кристалл.

8. "Использовать" монету, стоя рядом с кристаллом,после чего появится дырка. Ее нужно взять (Внимание! Она не видна, появится только надпись!).

9. С дыркой нужно идти к колодцу, захватив по дороге аквариум для золотых рыбок.

10. С аквариумом и дыркой дойти к левой стене у колодца. После "использования" дырки в высокой стене появится проход. Войдите в него.

11. Вы окажетесь в кладовой, где на столе висят скелеты смельчаков, которые пытали счастья перед Вами. Но вам нечего бояться - вас охраняет волшебный аквариум для золотых рыбок. Если бы его не было, паук сделал бы с Вами то же, что и с предшественниками.

12. Нужно забрать ключ и выйти. У колодца оставьте аквариум, он больше не понадобится.

13. С ключом идите на пляж, возьмите тапочки и выходите на берег моря.

14. Ныряйте.

15. Под водой отворите морской шкаф и заберите из него шпинат, выбирайтесь на поверхность и возвращайтесь к гейзеру.

16. Возьмите яйцо. Имея с собой его и шпинат, дергайте за веревку и быстро прыгайте в действующий гейзер.

17. Гейзер вынесет вас на вершину дерева, где находится орлиное гнездо.

18. Из гнезда нужно забрать лук и стрелы, оставить шпинат и выскочить из гнезда в струю гейзера, которая отнесет Вас обратно.

19. Оставить яйцо, оно уже больше не понадобится.

20. С луком и стрелами идти к тому месту, где висит Вильма.

21. Если выстрелить из лука (клавиша "ENTER") в "ходячего вождя" то он пойдет в другую сторону.

22. На поляне, где подвешена Вильма, нужно выстрелить из лука в стражника. Он пропадет.

23. Взять оставленный здесь топор и "использовать" его, стоя под веревкой, на которой висит Вильма. Она покажеться на месте большого скелетика.

Обратите внимание, что в процессе игры Вы уже собрали плот - можете спасаться с острова. Со всей семьей идите на пляж. Входите в воду и - отплывайте на поиски новых приключений.

Управление

1. Возможно использование джойстиков - кемпстона и протека.

2. Клавиатура:

- налево: от "Q" до "O" через клавишу;
- направо: от "W" до "P" через клавишу;
- прыжок: нижний ряд клавиш;
- "использование" предметов, а также переход из картинки в картинку: клавиша "ENTER".

Кроме того, используются клавиши:

- клавиша "1" - взятие предметов с переднего плана (когда ничего нет, появляется надпись "NOTHING").
- клавиша "2" - то же, что и "1", но предметы появляются на заднем плане.
- клавиша "3" - выключение цвета для черно - белого телевизора.
- клавиша "4" - пауза.
- клавиша "5" - включение/выключение музыки.

9. PINBALL

Загрузка программы: LOAD", затем ENTER и нажмите PLAY магнитофона.

Распространенная детская игра. Пружинная пусковая установка (смотрите с правой стороны экрана) выбрасывает шарик. Шарик пролетая, ударяет встречающиеся предметы. За это игроку начисляются очки. Падает шарик в провалы с правой и левой стороны игровой доски. Однако, можно несколько продлить игру, если вовремя отбить шарик специальными рычагами.

Клавиши управления:

G - пуск шарика
A,P - обьем рычагов

Игроку дается пять попыток. Сила запуска шарика зависит от длительности нажатия клавиши, но не передержите.На экране видно, как сжимается пружина. Поймайте момент, когда пружина максимально сжата. Количество очков показано на табло в центре игровой доски. Для возобновления игры нажмите ENTER.

10. TF - COPY - 6

(копировщик программ)

Описание копировщика программ TF-COPY-6 находится в главе 2.

Глава 2. Системные программы.

В этой главе содержатся описания программ, написанных на кассете 1 на стороне B.

1. TF-COPY-6

(копировщик программ)

Описиние.

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

```
TAPE FILE COPY  -  11/86

OPTIONS:   LOAD
           SAVE
           VERIFY
           DELETE
           CLOCK
           MODE
           RENAME
           CURSOR KEYS
PARAMS:    NUMBER
           ALL
           BEGIN
           PROGRAM ( BASIC + ...)
           FILE ( HEADER + DATA )

- COMPRESSED FORM IN RAM -
  (C)ARNOST VESERKA - OLOMOVC -
```

После стартовой загрузки программы появляется стартовое меню (звездочкой отмечены мерцающие цифры-режим по умолчанию

```
        0 - START
        1 - CLOCK-SET
  *     2 - MODE-SELECT
        3 - RENAME
        4 - 41984 BYTES - 14 LINES
  *     5 - 44032 BYTES - 6 LINES
        6 - 44288 BYTES - ATTR USED
        S - SAVE - BEEP ON
        X - SAVE - BEEP OFF
```

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

После нажатия на клавишу 0 (START) в верхней части экрана высвечивается обьем свободной памяти и команды, которые может выполнять программа:

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

SAVE (клавиша S) - выгрузить файлы. 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 файлы 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 пустить магнитофон на воспроизведение.

MODE (клавиша M) - модификация режима вывода программы с очисткой памяти.После нажатия на клавишу м появляется надпись: " CLEAR 1-2-3 "

>> последующее нажатие клавиши 1 переводит программу в режим вывода 14 строк каталога (41984 байт памяти свободно).

>> Клавиши 3 выводится 6 строк каталога (44288 байт памяти свободно), при этом в нижней части экрана отображается состояние экранной памяти, занимаемой 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 выдается сообщение:

WAIT: 456 SAMPLE: 2400 CURSN

Клавишами управления курсором необходимо установить нужные характеристики.

Сообщения:

При ошибках ввода с ленты появляется сообщение

ACCERT PARITY BYT=XXXXX

Необходимо повторить ввод.

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

ACCEPT BREAK BYT=XXXXX

2. ART STUDIO

Возможности редактора.

1. Система работы с перекрывающимся меню (аналогично "Макинтошу").
2. Использование следующих инструментов:
   - карандаша с произвольно назначаемым сечением грифеля;
   - кисти с произвольными размерами и формой;
   - краскораспылителя с регулируемым факелом краски;
   - валика для заливки с различной текстурой (кирпичи, рельеф, и т.д.);
   - лупы для рассматривания мелких деталей: 2-х, 4-х, 8-кратная.
3. Работа с окнами: размножение, переносы, повороты и т.д.
4. Работа с изображением: растяжение, сплющивание, наложение и т.д.
5. Работа с примитивами: точки, линии, окружности, треугольники, лучи и т.д.
6. Режим резиновых нитей (все прtмитивы - резиновые).
7. Печать надписей любым шрифтом в любом направлении.
8. Операции с файлами на кассетах (дискетах).
9. Работа с цветом и атрибутами.
10. Отмена любого неправильного действия.

SHAPES		Примитивы
POINT		Точка . 1
LINES	выбрано курсор меню	Линия 1. .2
CONT.LINE		Ломаная 1. .2 3. . 4
RECTANGLES		Прямоугольники 1. .2 3. .4
TRIANGLES		Треугольники 1. 3. .2
CIRCLES		Окружность 1. .2
RAYS		Лучи 1. 2. .3 . 4
ELASTIK V	вкл.	Режим кезиновых нитей
SHAP HRZ. X	выкл.	Дискретность координат по "x"
SHAP VRT. X		Дискретность координат по "Y"

Курсор примитивов (на схемах обозначен кружком). Цифры около кружков указывают последовательность ввода точек.

UNDO	Отмена

Отменяет любое последнее (!) действие, выполненное над экраном (в т.ч. операции в окнах). Для отмены подведите курсор меню к надписи "UNDO" и нажмите кнопку "FIRE" ("взять").

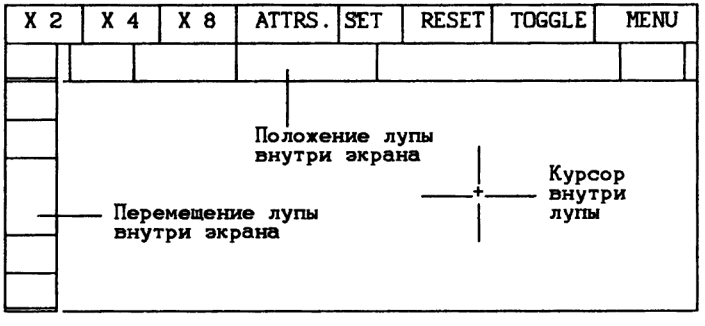
W I N D O W S	О к н а
DEFINE WINDOW	Определить 1. .2 окно . .3
LAST WINDOW	Последнее (то же) окно
WHOLE SCREEN	Весь экран-окно
CLEAR WINDOW	Очистка окна
CUT & PASTE WINDOW	Перенести и размножить
CUT,CLEAR & PASTE	Перенести и стереть старое
INWERT WINDOW	Инвертировать окно
RE-SCALE WINDOW	Перенести и масштабировать
CLEAR & RE-SCALE	Масштабировать и стереть старое
FLIP HORIZONTAL	Горизонтальный переворот (зеркало)
FLIP VERTICAL	Вертикальный переворот (зеркало)
ROTATE 1/4	Поворот на 90
ROTATE 1/2	Поворот на 180
ROTATE 1/4	Поворот на -90
MERGE X	Режим наложения
MULTIPLE X	Режим многократных действий

Окна имеют прямоугольную форму. При масштабировании с уменьшением тонкие линии и мелкие детали исчезают, а при увеличении линии увеличивают свою толщину. Многократные действия выполняются от последнего окна.

M A G N I F Y
MAG. X2
MAG. X4
MAG. X8
GRID X

Л у п а	
Увеличение	X2
Увеличение	X4
Увеличение	X8
Режим сетки	

Курсор лупы



Поле лупы. В нем находится увеличенное изображение фрагмента экрана.

В режиме триггера первое нажатие "F" зажигает точку, второе - гасит (или наоборот, если точка уже была зажжена).

M I S C E L L A N E O U S	С е р в и с - о п е р а ц и и
VIEW SCREEN	Весь экран - на экран
CLEAR SCREEN	Очистить экран
BRIGHT GRID 1	Шахматное поле 1x1 AT
BRIGHT GRID 2	Шахматное поле 2x2 AT
REMOVE GRID	Стереть шахматное поле
CHANGE COLOUR	Назначить цвет
VERSION NUMBER	Номер версии редактора

Шахматное поле получается битами яркости (BRIGHT) в атрибутах и используется для совмещения точек и цветов на экране.

A T T R I B U T E S	А т р и б у т ы
SET INC	Цвет карандаша
SET PAPER	Цвет бумаги
SET BORDER	Цвет бордюра
BRIGHT	Цвет MAX/MIN
FLASH	Мигалка вкл/выкл
OVER X	Режим надпечатки
INVERSE X	Режим инверсии
TRANSPARENT	Все параметры - транспон.
STANDARD	Все параметры - стандарт.

Установка атрибутов действует на все функции (FILL, SHAPES, и т.д.).

P A I N T
PEN
SPAY CAN
BRASH
EDIT BRASH
INVERSE X

Выходят в режим инструментов

К и с т и
Карандащ
Пульверизатор
Кисть
Редактор кисти
Режим инверсии

E D I T B R A S H	Редактирование кисти
D A T A $%$% M A S K	1. Подведите курсор к нужной Точке DATA или MASK. 2. Нажатием кнопки "взять" Зажечь (погасить) точку.

То, чем кисть красит

то, чем кисть стирает старое

F I L L
SOLOD FILL
TEXTURED
WASH TEXTURED
EDIT TEXTURED

выходят в меню текстуры

З а л и в к а
Сплошная заливка
Текстурная заливка
Текстурная размывка
Редактор текстуры

-Курсор заливки (валик)

"EDIT TEXTURE" похож на "EDIT BRUSH".

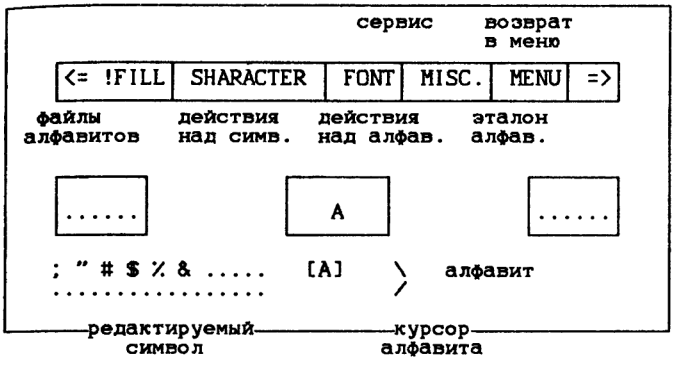
T E X T		Т е к с т
LEFT TO RIGHT		Печатать слева направо
DOWNWARDS		Печатать сверху вниз
NORMAL HEIGHT		Нормальная высота символа
DOUBLE HEIGHT	\/	Двойная высота символа
TREBLE HEIGHT		Тройная высота символа
NORMAL WIDTH		Нормальная ширина символа
DOUBLE WIDTH	\/	Двойная ширина символа
TREBLE WIDTH		Тройная ширина символа
SIDEWAYS	X	Буквы повернуть на 90
BOLD	\/	Жирная печать
CAPS LOCK	X	Только заглавные буквы
SHAP HRZ.	X	Дискретность по горизонтали
SHAP VRT.	X	Дискретность по вертикали
FOND EDITOR		Редактор символов

-

][- Курсор начала строки

__ -курсор символов

Курсор начала строки перемещается как любой другой курсор,но после нажатия клавиши "взять" исчезает,а на его месте появляется курсор ввода символов, после чего можно набивать любой текст с клавиатуры компьютера. После нажатия клавиши "ENT" ввод символов заканчивается и на экране появляется курсор начала строки. "SHAP" совмещает символы с атрибутами.



C H A R A C T E R	Действия над символами
CLEAR	Стирание символа
INWERT	Инверсия символа
FLIP HRZ.	Горизонт.Зеркальный поворот
FLIP VRT	Верт.Зеркальный поворот
ROTATE 1/4	Поворот на 90
SCROLL RIGHT	Роллинг вправо
SCROLL DOWN	Ролинг вниз

Такое же меню для действий над всем алфавитом. Алфавит можно загружать с магнитной ленты и выгружать на ленту.

C A S S E T T E	K a c c e t a
SAVE FTLE ...	Записать файл ...
LOAD FILE ...	Загрузить файл ...
LOAD NEXT FILE	Загрузить следующий файл
VERIFY FILE ...	Проверить файл ...
VERIFY NEXT FILE	Проверить следующий файл
MERGE FILE ...	Наложить файл ...
MERGE NEXT FILE	Наложить слудующий файл

- Для выбора режима нажмите "FIRE" (взять",указав курсором меню на необходимый режим.
- На запрос: "FILE NAME?" Введите с клавиатуры имя файла и нажмите "ENTER".
- Выше изображено меню операций над файлами изображений. Получение файла имеют формат "SCREEN" и могут загружаться в экран командой "LOAD" "SCREEN" из программ на языке "BASIC".
- Операции над файлами есть также в режиме "TEXT" (через меню "FONT EDITOR").

О с н о в н о е м е н ю .

печать	файлы	атрибуты	кисти	экран	отмена	указат. роллинга
PRINT	FILE	ATTRS	PAINT	MISC	UNDO	
WINDOWS	FILL	MAGNIFY		TEXT	SHAPES	
Окна	закраска валиком	лупа		текст	примитивы	ролл-инг экрана

Это меню всегда (кроме особых случаев) находится вверху экрана. Под ним две строки "невидимы", и при попадании курсора в них его не видно. Если это произошло (курсор "пропал"), продолжайте перемещать курсор вверх (вниз), и он появится.

В с е у п р а в л е н и е к у р с о р о м .

Русск.	Англ.	KEYBOARD	SINCLAIR
Вверх	UP	Q	SINCLAIR JOYSTICK N1
Вниз	DOWN	A	
Влево	LEFT	O	
Вправо	RIGHT	P	
Действие	FIRE	любая клавиша нижнего ряда	

Р е ж и м т е к с т
Клавиша "ENT" Завершает Печать строки

3. LIGHT CHOW

(цветомузыка)

Программа позволяет анализировать спектр Вашего магнитофона и работать в режиме цветомузыки.

Меню:

1 - анализ спектра вашего магнитофона;
2 - цветомузыка;
3 - техническая информация.

4. WHAM

Музыкальный редактор позволяет составлять различные мелодии.

Меню:

1 - чтение мелодии;
2 - запись мелодии;
3 - прослушивание мелодии;
4 - компилятор;
5 - изменение типа звучания;
6 - выход в меню;
7 - подсказка работы с редактором.

Описание:

После загрузки программы на экране монитора появляется главное меню звучит мелодия, демонстрирующая возможности редактора.
Нажмите клавишу 7 (подсказка) на экран компьютер выдает следующую информацию:

клавиатура

1-4 - выбор октавы;
5 - изменения цвета бордюра;
6 - выход в основное меню;
7 - удаление мелодии;
8 - изменение музыкальных эффектов;
9 - возврат назад по нотному стану;
0 - возврат назад на 1 ноту;
Q - проигрывание мелодии;
E - барабанный эффект;
T - выбор канала;
J,U,I - ударный эффект;
D - ускоренное проигрование мелодии;
P - пауза;
W - удаление старой мелодии;

0 - прослушивание записи.

Нажмите клавишу "6" и Вы вернетесь в основное меню. На нашей кассете записин пример мелодии. Для ее прослушивания нажмите клавишу "1" (чтение мелодии). Нажмите клавишу "T" (чтение с магнитофона).

Наберите имя файла EXAMPLE и включите магнитофон на воспроизведение. Компьютер считает файл и перейдет в режим воспроизведения мелодии.

Желаем творческих успехов с работой в музыкальном редакторе.

5. SINTEZATOR

Программа позволяет получить 10 различных мелодий. Вы можете выбирать их, нажимая клавиши "1...0" и менять темп мелодии клавишами "P","O".

6,7. RUSTAS,TASWORD

Текстовые редакторы TAWORD - на английском и RUSTAS - на русском языках позволяют редактировать и выводить тексты на экран монитора и принтер. Тексты содержат 64 символа в сборке.

Меню:
------
*

EDIT (едит)	- страница помощи;
CAPS SHIFT(капс шнифт)	- верхний регистр;
TRUE VIDEO (норм видео)	- курсор на слово влево;
INV VIDEO (инк видео)	- курсор на слово вправо;
ARROWS (стрелки)	- передвижение курсора;
GRAPHIC (графика)	- цифры и упр. Символы;
DELETE (делете)	- стирание символа;
<=	- сдвиг стрелки влево;
<>	- центрирование строки;
>=	- сдвиг строки вправо;
AND (анд)	- вставить строку/символ;
OR (ор)	- курсор в конец текста;
AT (ат)	- курсор в начало текста;
STOP (стоп)	- загрузка/запись/начать;
NOT (нот)	- удалить строку;
STEP (степ)	- выравнять параграф;
TO (то)	- скролл вниз;
THEN (тхен)	- скролл вверх;
ENTER (вк)	- начало новой строки;

EX MODE (экстендед моде) - вход/выход из экстендед моде;
CAPS SHIFT + SYMBOL SHIFT - 2-я страница помощи.
ENTER - возврат к тексту

* - в скобках приведены сообщения для редактора RUSTAS

8. PASCAL HP80

Описание редактора HP80.

Управление редактором:

CAPS-SHIFT 1 EDIT - редактировать строку, выделенную экранным курсором;

CAPS-SHIFT 2 CAPS-LOCK - фиксация регистра больших или малых символов;

CAPS-SHIFT 3 TRUE VIDEO - страница вперед;

CAPS-SHIFT 4 INV VIDEO - курсор влево/строка влево;

CAPS-SHIFT 6 <стрелка вниз> - экранный курсор вниз;

CAPS-SHIFT 7 <стрелка вверх> - экранный курсор вверх;

CAPS-SHIFT 8 <стрелка вправо> - 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.

6 MOVE - перенести блок текста;

7 ERASE - удалить блок текста;

0 FORMAT - переключить формат экрана 64/32 символа и наоборот;

Z COPY - копировать блок текста;

X CLEAR - очистить текстовый буфер;

Y RETURN - перейти к HI SOFT формату и вводить текст средствами BASICa;

F - задать литерал и установить режим поиска;

R - задать литерал для замены.

Внимание:

для выполнения прочих команд используйте стандартный редактор HP80.

Команды режима поиска:

S - найти образец;

R - заменить найденный образец на заданный литерал;

G - перейти к замене остальных образцов.

Внимание!

Звуковой сигнал сообщает о нажатии недопустимой клавиши.

Неиспользуйте графический регистр вне текста программы!

PASCAL не имеет встроенных графических процедур и функций!

Далее приводятся сведения, в основном касающиеся отличий от стандартного языка PASCAL .

Ограничения:

- тип FILE не реализован;

- тип RECORD не может иметь вариантные поля;

- процедуры и функции не могут являться формальными параметрами;

- недопустимо использовать указатели на ранее не определенный тип.

Компиляция и исполнение:

компиляция: - C LINE NUMBER;

остановка листинга: - BREAK.

В случае успешной компиляции программы на запрос <RUN?> ответить <Y>, если требуется исполнение программы. Возврат в редактор осуществляется нажатием любого другого символа.

При контроле правильности использования типов данная версия паскаля использует эквивалентность имен, а не структур. Поэтому переменные

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

не могут совместно применяться в выражении.

Присваивание A:=B также недопустимо.

Синтаксис и семантика.

Число без знака ::= #шестнадцатиричное число;

Строковая константа записывается в кавычках <" "> и содержит не более 255 символов.

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

Ключевое слово <PACKED> игнорируется.

Тип множества может иметь не более 256 членов.

Пустой список в операторе <CASE> приводит к ошибке. Если подходящая ветвь в списке <CASE> не найдена, то 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 и быть на одном уровне вложения с меткой.

Программа не имеет параметров.

Предопределенные идентификаторы:

Константы: MAXINT [=32768],
TRUE,
FALSE.

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

Процедуры и функции:

Ввод	- READ;	
	READLN -	ввод с переходом на следующую строку;
Вывод	- WRITE;	
	WRITELN -	вывод с переходом на следующую строку;
- на дисплей	CHR(8) -	BS на дисплее (шаг назад);
или принтер	- CHR(12) -	очистка экрана (CLEAR или NEWPACE);
	CHR(13) -	CR и LF ("возврат каретки" и "перевод строки");
	CHR(16) -	переключить вывод с дисплея на приер и наоборот;
- с клавиатуры	- INCH.	

Функция читает символ с клавиатуры, и если не одна клавиша не нажата, возвращает CHR(0).

Арифметические функции:

FRAS - дробная часть.

Другие процедуры и функции.

Процедура <DISPOSE> исключена, вместо нее имеются процедуры:

- MARK;	
- RELEASE.	
- INLINE(C1,C2,C3...)	- C1, C2, C3... - Шестнадцатеричные числа, размещаются как последовательность байтов;
- USER(V)	- V - целое число. Функция USER(V) вызывает подпрограмму с адресом V, она должна завершаться инструкцией RET;
- HALT	- завершает программу;
- POKE(X,S)	- помещает значение S в ячейку с адресом X;
- TOUT(NAME,START,SIZE)	- записывает данные на магнитную ленту;
- TIN(NAME,START)	- читает данные с магнитной ленты;
- OUT(P,C)	- равносильно: BC:=P; A:=C; OUT(C),A;
- RANDOM	- возвращает случайное число в диапазоне 0...255;
- ADDR	- возвращает адрес аргумента произвольного типа;
- PEEK(X,R)	- R - аргумент произвольного типа, значение которого заносится в ячейку с адресом X;
- SIZE(W)	- возвращает размер памяти, занимаемой переменной W;
- INP(P)	- аналогично OUT.

Комментарии:

"(*" "*)" и " " " "

Ключи располагаются внутри комментариев вслед за символом $ (знак денежной единицы). 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 массива (по умолчанию +A);
I
P
F - прочитать часть программы с магнитной ленты;
позволяют экономить память.

Стандартный редактор:

I M,N - режим автоматической нумерации строк программы;
L N,K - распечатать на дисплее текст от строки N до строки K;
D N,K - удалить часть текста от строки N до строки K;
K N - при выдаче листинга распечатать без останова N строк;
M N,K - заменить строку с номером K на строку с номером N;
F N,K,V,W - найти в диапазоне строк с номерами N, K образец V и заменить его на W;
E N - редактировать строку с номером N.

Подкоманды:

<пробел> - инкремент текстового указателя;
DELETE - декремент текстового указателя;
CAPS SHIFT/T - табуляция вправо;
ENTER - конец редактирования;
Q - отменить редактирование;
R - начать редактирование заново;
F - найти ранее заданный образец;
S - заменить образец;
I - вводить символы после курсора с авторазвижкой (завершается ENTER);
X - курсор перемещается в конец строки;
C - вводить символы с позиции курсора с уничтожением исходного текста;
K - удалить символ под курсором;
Z - удалить символы от курсора до конца строки.

Команды для работы с магнитофоном:

P N,K NAME - текст со строки N до строки K записать в файл с именем NAME;
G NAME - прочитать текст из файла с именем NAME;
W N,M,S - записывает текст в форме, которая потом может использоваться с ключом $F N,M,S.

Компиляция и выполнение программы:

C N - компилировать текст, начиная со строки N;
R - исполнить скомпилированную программу;
T - отлаженная программа компилируется таким образом, что перекрывает в памяти компилятор. В дальнейшем может быть записана на магнитофон и исполнена автономно;
B - возврат в BASIC.

Список рекомендуемой литературы:

1. Йенсен К., Вирт Н.,
Паскаль: руководство для пользователя и описание языка.
М.: Финансы и статистика, 1982.

2. Грогоно П.,
Программирование на языке PASCAL.
М.: Мир, 1982.

3. Вирт Н.,
Алгоритмы + структуры данных = программы.
М.: Мир, 1985.

4. Грэхем П.,
Практический курс языка PASCAL для микро-ЭВМ.
М.: Радио и связь, 1986.

5. FINDLAY W., WATT D.A.,
PASCAL.AN INTRODUKTION TO METHODICAL PROGRAMMING.
THIRD EDITION. - LONDON: PITMAN, 1985.

6. Абрамов Б.Г., Трифонов Н.П., Трифонова Г.Н.,
Введение в язык PASCAL.
М.: Наука, 1988.

9. BETA BASIC

BETA BASIC дополняет BASIC ZX SPECTRUM тридцатью новыми командами и двадцатью одной новой функцией. Все старые команды и функции выполняются.
Новые команды вызываются в режиме "GRAPHIC".

Описание команд BETA BASICa

ALTER <описание атрибутов> TO <описание атрибутов> [A]
позволяет быстро изменить атрибуты экрана (INC, FLASH, BRIGHT, PAPER) без необходимости очищения экрана. Например: ALTER TO PAPER 1, INC 6 или ALTER INC 7 TO PAPER 2,INC 0.

AUTO <номер линии><,шаг> [6]
включение автоматической нумерации линий, начиная от данного номера и с данным шагом (если задан), либо от текущей линии с шагом 10. Выключение - нажатие и удержание клавиши BREAK более 1 сек.

BREAK [SHIFT-SPACE]
это не ключевое слово, но используется оно в режиме GRAPHICS. Оно может прерывать действие каждой программы, т.к. система BETA BASIC работает в режиме INTERRUPT 2 микропроцессора Z80.

CLOCK число или цепочка [C]
управляет 24-часовыми часами, которые выполняют различные функции, зависящие от поданного аргумента.

Аргумент	Переход на Линию по ALARM	Звучание ALARM'A	Показание (Высвечивание)
0	Нет	Выкл	Нет
1	Нет	Выкл	Есть
2	Нет	Установ	Нет
3	Нет	Установ	Есть
4	Да	Выкл	Нет
5	Да	Выкл	Есть
6	Да	Установ	Нет
7	Да	Установ	Есть

Пример:
CLOCK "09:29:05" - установка времени;
CLOCK "A006:20" - установка будильника;
CLOCK "A9000" - переход на линию.

CHR$ 8 - курсор влево;
CHR$ 9 - курсор влево;
CHR$ 10 - курсор влево;
CHR$ 11 - курсор влево.

Эти знаки, примененные в цепи печати с помощью PRINT, изменяют позицию печати в соответствии с описанием.

DEF KEY - цепь однознаковая; цепь SHIFT 1.
DEF KEY - цепь однознаковая: инструкция: инструкция:...
Подстановка под клавишу с данным знаком данной цепочки либо цепи инструкций.
DEF PROC название процедуры [1]
ключевое слово, начинающее определение процедуры, вызванной через название.

DELETE <номер линии> TO <номер линии> [7]
удаление линий программы в заданной области. Допускается использование различных вариантов, например: DELETE TO 50; DELETE 10 TO 50; DELETE TO и т.д.

DO
DO WHILE условие
DO UNTIL условие [D]
ключевые слова, начинающие определение петли. Петля заканчивается словом LOOP.

DPOKE адрес,число [P]
загрузка двух очередей байтов памяти с данным адресом числом из области от 0 до 65535.

EDIT номер линии [0]
нормальный ввод. Позволяет редактировать произвольную линию,строку. Появляется после нажатия клавиши 0, если до этого была нажата клавиша ENTER.

ELSE инструкция [E]
это часть инструкции IF-THEN как альтернатива начальной инструкции. Пример использования:
IF условие THEN инструкция ELSE инструкция.

ENDPROC [3]
слово, оканчивающее определение процедуры, вызываемой через название.

EXIT IF условие [I]
условный выход из петли DO-LOOP.

FILL X,Y [F]
FILL <INC цвет>;X,Y [F]
FILL <PAPER цвет>;X,Y [F]
заполнение области фона цветом знака (если используется FILL или FILL INC) или области знака цветом фона (если используется FILL PAPER). Допускается также использование сложных конструкций, заполнение при этом начинается от текущих координат X, Y:
FILL INC 2; PAPER 1; FLASH 1; X,Y.

GET цифровая переменная или рядовая переменная [G]
присваивает переменной номер клавиши (с цифровой клавишей или 11 - для A, 12 - для B, 13 - для C и т.д.) Или знак при нажатой клавише.

JOIN <номер линии> [SHIFT 6]
связывает линию с данным номером, либо текущую линию, если номера нет, с линией, находящейся в нижней части экрана, придавая новой линии заданный номер (либо номер текущей линии).

KEYIN цепь-ряд [SHIFT 4]
допускается использовать только как инструкцию в программе. Вызывает вставление в программу поданной цепи (к примеру, это может быть строка программы).

KEYWORDS 1 [8]
KEYWORDS 0 [8]
переключатели ключевых слов BETA BASICa на графические знаки ZX SPECTRUM,получаемые в режиме G. Это слово никогда не выключается.

LIST номер линии TO номер линии
(LLIST - также, но не для принтера) расширение синтаксиса нормального ключевого слова SINCLAIR BASICa.

LOOP [L]
LOOP UNTIL условие [L]
LOOP WHILE условие [L]
заканчивающая цикл часть структуры DO-LOOP.

ON [O]
применяется в структурах вида:
GO TO ON переменная;номер линии,номер линии,... Или
GO SUB ON переменная;номер линии,номер линии,...
Производит переход к соответствующей линии в зависимости от переменной.

ON ERROR номер линии [N]
включение обслуживания ошибок через линию с заданным номером (к ней переходит программа при появлении ошибки при ее выполнении). Дополнительно переменной ERROR присваивается значение, равное коду ошибки. Не выполняется при кодах 0 (O.K.) И 9 (STOP). Выключение обслуживания ошибок наступает после перехода к программе обработки ошибок, либо по команде ON ERROR 0.

PLOT X,Y;цепь
нормальный ввод. Позволяет рисовать в любом месте экрана. Координаты относятся к левому верхнему краю первого знака цепи. В цепи можно использовать знаки управления курсором.

POKE адрес,цепь
ключевое слово, расширяющее синтаксис. Производит запись в память заданной цепи знаков, начиная с заданного адреса.

POP <цифровая переменная> [0]
записывает адрес со стека GOSUB,DO-LOOP,PROC. Указанный этим адресом номер линии подставляется как значение переменной, если ее название было задано.

PROC название [2]
выполнение процедуры с заданным названием.

RENUM <начало то конец><LINE новое начало><STEP шаг> [4]
перенумерация строк в заданной области в соответствии с заданными параметрами. Если не был задан ни один параметр, то начальный номер строки принимает значение 10 и шаг равен 10.

ROLL код направления <,пиксель><;X,Y;ширина,высота> [R]
"переворот" определенного окна экрана. Образ, исчезающий с одной стороны, появляется с другой. Заданная ширина окна касается количества позиций знака, а не элементов образа. Координаты X и Y показывают левый верхний угол выбранного окна. Команда вызывает перемещение на 1 элемент.

Если задан только код управления, то смещается целый экран.Можно переместить целый образ вместе с атрибутами, только атрибуты, либо только часть образа. При повороте атрибутов заданное число должно быть равно 8. Коды направлений имеют следующие значения:

Коды направления	Направление	Охватывает
1	Влево	Атрибуты
2	Вниз	Атрибуты
3	Вверх	Атрибуты
4	Вправо	Атрибуты
5	Влево	Содержимое образа
6	Вниз	Содержимое образа
7	Вверх	Содержимое образа
8	Вправо	Содержимое образа
9	Влево	Весь образ
10	Вниз	Весь образ
11	Вверх	Весь образ
12	Вправо	Весь образ

SCROLL <код направления><,пиксель><;X,Y;ширина,высота> [S]
перемещение содержимого определенного окна экрана на один элемент. Синтаксис такой же, как и у ROLL. Здесь также можно не задавать ни одного параметра и тогда наступит переход всего экрана на одну линию вверх (как в ZX-81). Содержание экрана пропадает на краю окна.

SORT таблица или цепь [M]

SORT INVERS таблица или цепь [M]
упорядочивает знаковую таблицу или цепь в очередности кодов знаков или наоборот, а цифровую таблицу - от наибольшей цифры до наименьшей или наоборот. Числовая таблица может иметь не более двух измерений.

SPLIT

это не ключевое слово,вместо него применяется нормальный знак"<>" (SS/W). Он указывает, в каком месте программы строка может быть разделена на две части. Часть перед знаком присоединяется к программе; часть после знака остается в программе с тем же самым номером.

TRACE <номер линии> [T]
команда, приводящая к пошаговому выполнению программы. Ее подача вызывает переход GOSUB к заданной строке непосредственно перед выполнением каждой инструкции программы. Выключение через RUN, CLEAR или TRACE 0.

UNTIL условие [K]
часть петли DO-LOOP, используемая в DO UNTIL или в LOOP UNTIL,позволяющая ее условное выполнение.

USING цепь-образец;число [U]
применяемое в инструкции PRINT USING, позволяет задавать соответствующий формат печатаемой цифры. Знак "#" в цепи-образце означает предварительное разрежение.

WHILE условие [J]
часть петли DO LOOP, используемая в DO WHILE или в LOOP WHILE. Позволяет условное выполнение цикла.

Описание функций BETA BASICa

Ниже дано описание функций, дополненных в системе BETA BASIC, вместе с описанием их аргументов и способов вызова.

AND (число,число) [FN A( ]
логическое произведение двух чисел, выполненное в двоичном представлении.

BIN$ (число) [FN B$( ]
двоичное представление десятичного числа.

SHAR$ (число) [FN C$( ]
преобразование целого числа без знака из области 0...65535 В соответствующее двузначное число.

COSE (число) [FN C( ]
косинус числа (4 значащих цифры, работает быстрее, чем в оригинале системы SPECTRUM).

DEC (цепь) [FN D( ]
значение двухбайтового числа, находящегося по заданному адресу.

FILLED ( ) [FN F( ]
количество элементов образа, заполненных перед последней командой FILL.

HEX$ (число) [FN H$( ]
переводит шестнадцатеричное число в десятичное.

INSTRING (старт, последоват. 1, последоват. 2) [FN I( ]
позиция первого знака последовательности 2 в последовательности 1 при просмотре последовательности от заданной стартовой позиции, либо 0, если элементы последовательности 2 не содержат в последовательности 1.

MEM ( ) [FN M( ]
количество свободных байтов памяти.

MEMORY$ ( )[FN M$( ]
содержимое всей памяти (от адреса 1 до 65 532) трактуется как цепь-последовательность.

MOD (число 1,число 2) [FN V( ]
результат деления числа 1 по модулю числа 2.

NUMBER (последовательность) [FN N( ]
целое число без знака в пределах 0...65535, Которое является соответствием двузначной последовательности.

OR (число 1,число 2) [FN O( ]
логическая сумма двух чисел, представленных в двоичной записи.

RNDM (число) [FN R( ]
псевдослучайное число в пределах от 0 до заданного числа включительно.

SCRN$ (строка,столбец) [FN K$( ]
знак, находящийся на заданной позиции (также знак, определенный пользователем.

SINE (число) [FN S( ]
синус данного числа (4 значащих цифры).

STRING$ (число,цепь) [FN S$( ]
повторение последовательности заданное число раз.

TIME$ ( ) [FN T( ]
прошедшее время, задаваемое через CLOCK.

USING$ (цепь-образец,число) [FN U$( ]
знаковая запись числа в заданном формате (как USING).

XOR (число 1,число 2) [FN X( ]
логическая операция EXCLUSIVE OR на двоичной записи двух чисел.

Специальные переменные

В системе BETA BASIC существуют определенные переменные, генерируемые через систему и доступные через названия. Ниже приведено их описание:

XOS,YOS
координаты середины системы расположения (вначале установлены на 0,0). Их можно изменять: x=0...255 Y=0...175. Обнуление через команды CLEAR и RUN.

XRG,YRG
диапазон координат на экране (XRG=256, YRG=176), которые можно свободно изменять. CLEAR и RUN задают начальные значения.

Во время выполнения команды ON ERROR или TRACE создаются специальные переменные:

ERROR
: выдает код последней обнаруженной ошибки.

LINE
: номер остановленной исполняемой линии (при TRACE) или линии, в которой была обнаружена ошибка (при ON ERROR).

STAT
: номер остановленной выполняемой инструкции (при TRACE), либо инструкции, в которой была найдена ошибка (при ON ERROR).

Названия специальных переменных можно набирать как большими, так и малыми буквами.

10,11. RAMDOS

(квазиэлектронный диск, описание программы)

RAM DOS позволяет быстро менять программы без помощи дополнительной аппаратуры. Она использует RAM как электронный диск при помощи новых BASIC-команд доступа к диску. Новые команды просты и похожи на стандартные команды доступа к магнитофону. RAM DOS не ограничивает количество записанных файлов, но ограничивает количество занятой памяти (примерно 33к), определяемой установкой RAMTOPа. Каждый файл записывается на диск с заголовком, занимающим 17 байтов.

RAM DOS включается командой:

```
RAMD USR 63600
```

С этого момента все новые команды активны, но не мешают выполнению остальных команд. Вы можете использовать команду NEW без риска стереть файлы, т.к. они находятся под RAMTOPом, который перед вводом программы с магнитофона устанавливается командой:

```
CLEAR 3200
```

(или другого, более подходящего значения)

Все новые команды вводятся одной из малых букв, следующих за звездочкой:

*l - LOAD — вводится перед именем программы, которое не должно превышать 10 знаков. Выбор SCREEN, CODE обычен. Если имя файла нуль (""), то вызывается последний введенный файл. Данные не доступны.

*s - SAVE — используется также, но имя не может быть нулем. Отдельно данные не записываются (только вместе с программой). Если на диске уже имеется файл с таким именем, то он будет стерт.

*m - MERGE — работает только с программными файлами как и с магнитофоном.

*d - DELETE — должно следовать за именем файла. Если оно пустое, то будет стерт последний. За именем указывается тип файла.

*c - CATALOG — без параметров. Печатает все имена файлов, а также количество оставшейся свободной памяти на RAMDOSe.

*e - EPACE — очищает RAM-диск.

*t - TAPE — необходимо указывать имя файла. Записывает все содержимое RAM-диска вместе с программой RAMDOS как стандартный код. Перед вводом файла с магнитофона не забывайте ввести CLEAR 30000 или др., затем введите LOAD "" CODE и затем RAMD USR 63600 для включения новых команд.

*f - FREE — печатает число байтов в памяти BASICa, т.е. ниже RAMTOPа. Общее значение свободной памяти определяется сложением с величиной из *C.

*x - BLOCK DELET - стирает все линии BASICa от указанного номера до указанного номера до указанного. Эта команда не относится к RAMDOS, но она коротка и очень полезна.

С о о б щ е н и я :

OUT OF MEMORY - нет места на RAMDOSe для новых файлов. Если стирание старого файла не дает эффекта, попробуйте опустить ниже RAMTOP. Если такое сообщение встретится при вводе программы с магнитофона, значит не хватает места в памяти BASICa. Попробуйте поднять RAMTOP выше.

END OF FILE - файл не найден.

Если Вы захотите изменить атрибуты, введите их новое значение в ячейку 63714 (POKE 63714,XX).

Для работы с файлами из программ, написанных на языке ассемблера, рекомендуется пользоваться стандартными программами, находящимися в ПЗУ компьютера. Их десятичные адреса равны:

подпрограмма чтения с ленты.....1366;
подпрограмма записи на ленту.....1218.

При вызове этих подпрограмм необходимо задать:

регистр 'A' - тип файла (0...255);
Регистр 'DE' - длина файла;
регистр 'IX' - адрес начала загрузки файла.

Кроме того, для подпрограммы чтения с ленты необходимо задать тип работы (проверка или чтение) флажком переноса. Содержимое флажка переноса означает:

0 - режим чтения;
1 - режим проверки (т.е. сравнение содержимого памяти с информацией считываемой с ленты).

Для работы с кассетами, записанными в формате "SPECTRUM" на других компьютерах, необходимо помнить:

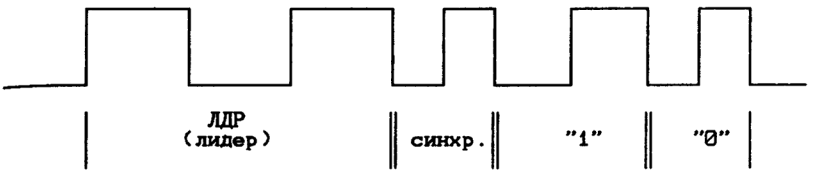
1. Для определения начала файла сигнал "лдр" деректируется не менее 0.5 сек. для предупреждения сбоев;

2. Если длительность очередного полупериода "лдр" стала значительно меньше заданного, то надо начинать проверку на синхропериод;

3. Если длительность очередного полупериода "лдр" стала значительно больше заданного, то весь цикл надо начинать сначала;

4. Проверку на окончание данных надо производить (даже если имеется счетчик считанных байтов) сравнением длины полного периода бита (0.98 или 0.488 мс) с указанными значениями. Если счетчик слишком мал или слишком велик, то считается, что данные кончились.

Временные диаграммы записи-считывания:



Каждый блок информации (файл) на магнитной ленте начинается с синхронизующего сигнала (лдр) длительностью 2 или 5 секунд. Частота этого сигнала - около 810 гц (период 1.25 мс). После этого сигнала идет один период специального синхросигнала - длительность "нуля" - около 0.19 мс, "единицы" - 0.21 мс. Затем следуют байты данных, передаваемые последовательно, начиная со старшего бита. "Нулевой" бит передается одним периодом сигнала частотой около 2047 гц (период - 0.489 мс.), "единичный" - одним периодом сигнала частотой около 1023 гц (период - 0.978 мс).

Каждый файл состоит из байта типа (значение от 0 до 255), собственно данных и байта контольной суммы. Байт типа и байт контрольной суммы не входят в длину файла. В контрольную сумму входит значение байта типа.

Байт типа обычно принимает значение 0 (для заголовков) и 255 (для данных). Если есть необходимость, пользователь может использовать и любой другой тип. От значения байта типа зависит длительность сигнала "лдр". При значении байта типа от 0 до 7 длительность сигнала равна 5 сек. Если байт типа больше 7, то длительность сигнала 2 сек.

Стандартный файл, формируемый компьютером по команде загрузки, состоит из двух файлов: файла заголовка длиной 17 байт, и файла данных.

Файл заголовка имеет следующий формат:

номера байтов	назначение
1	тип информации, описываемой заголовком: 0 - программа на бейсике; 1 - числовой массив; 2 - символьный массив; 3 - "байтовый" файл: программа в машинных кодах или образ экрана.
2-11	Имя файла в кодах КОИ-7; если имя отсутствует, то первый байт равен 255.
12-13	Длина файла данных.
14-15	Для программы - номер строки для автозапуска программы. Если первый байт равен 80H, то автозапуск не

	был задан; для "байтового" файла - адрес загрузки; для массивов - второй байт содержит имя массива (один символ КОИ-7).
16-17	Только для программы - длина программы на BASICe.

12. DIAGNOSTICS TAPE

Назначение.

Тестирование магнитофона для записи программ ZX SPECTRUM.

Возможности.

Вы можете посмотреть осциллограмму сигнала с магнитофона, проанализировать спектр сигнала, сделать тестовую запись на магнитофон и проверить правильность записи.

Описание.

При загрузке программы Вы попадаете в меню выбора функции. Для выполнения функции нажмите первую букву ключевого слова. Вы увидите пояснение. Нажмите кнопку ENTER и функция начнет выполняться. Для возврата в меню нажмите кнопку Q.

Меню:

H - детальное обьяснение работы,
O - полечение осциллограммы сигнала,
T - анализатор спектра,
R - запись тестового файла на магнитофон,
V - проверка записи на магнитофон,
E - конец работы с программой.

13. TEST PROGRAM

Назначение

Проверка работоспособности ZX SPECTRUM и внешнего оборудования.

Возможности

TEST PROGRAM проверяет клавиатуру, цвета на экране, каналы, память и запись на магнитофон.

Описание

После загрузки программы Вы попадаете в меню программы:

1 - клавиатура SPECTRUM 16/48K.
2 - Клавиатура SPECTRUM +.
3 - STOP KEY SPECTRUM +.
4 - Цветовая палитра.
5 - Звук.
6 - Каналы.
7 - Память.
8 - Магнитофон.
9 - Все тесты подряд для SPECTRUM 16/48K.
0 - Все тесты подряд для SPECTRUM +.

14. TV PATTTERN GENERATOR

Назначение.

Проверка и настройка монитора или телевизора.

Возможности.

Тестируется цвет, линейность и искажения на экране телевизора.

Описание.

После загрузки программы Вы попадаете в меню, где описаны кнопки управления:

0-7 - кнопки управления цветом.
P - сетка из точек.
O - сетка из линий (решетка).
I - шахматное поле.
F - установить цвет всего экрана, заданного кнопками управления цветом.
V - вертикальные линии.
H - горизонтальные линии.
C - окружность в центре.
X - окружности в углах.
L - цветовая палитра слева.
R - цветовая палитра справа.
M - смешанная цветовая палитра.
U - цветовая палитра сверху.
D - цветовая палитра снизу.
N - комплексный тест.
? - Выход в меню.

После каждого теста не обязательно возвращаться в меню. Можно, нажимая соответствующие кнопки, запускать следующий.

Внимание!

Звуковой сигнал сообщает о нажатии не допустимой клавиши.

SIMBOL-SHIFT CAPS-SHIFT - запрос "COMMAND" (команда)

Допустимые ответы:

4 OPEN# - открыть блок текста;

5 CLOSE# - закрыть блок текста;

В этой книге находится описание программ GENS4 и MONS4, расположенных на кассете 1 на стороне В.

С о д е р ж а н и е

15. GENS4

(HISOFT GENS4)

Раздел 1. З а п у с к

1.1. Введение и инструкции загрузки

GENS4 - это мощный и легкий в использовании ассемблер для Z80, который очень близок к стандартному ассемблеру ZILOG по определению. В отличии от многих других ассемблеров. Пригодный для компьютеров "SPECTRUM", GENS4 является обширной, профессиональной частью программного обеспечения и побуждает Вас очень тщательно изучать следующие разделы вместе с примером приложения 3 перед попытками использования ассемблера.

Если Вы еще навичок, то вначале проработайте примечание денных в библиографии.

. У нас есть версии систем DEVPAC 4 на диске, для DISCIPLE & OPUS DISCOVERI дисковой системы и для компьютеров SRECTRUM PLUS 3. Эти версии работают точно так же, как описано в этом документе, просто заменяйте слово "микропривод", там где оно встречается, на слова "ДИСК OPUS", "ДИСК PLUS 3" или "DISCIPLE ДИСК" в соответствии с Вашей системой. Существует несколько внешних особенностей для версии PLUS 3, которые описаны на дополнительном листке.

GENS4 занимает приблизительно 10к в памяти и использует свой внешний стек, так что это - замкнутая часть программного обеспечения. Она содержит собственный строчный редактор, который помещает текстовой файл сразу за кодами GENS4, в то время как таблица символов ассемблера создается за текстовым файлом, поэтому Вы должны предоставить достаточное пространство, чтобы поместить сам ассемблер, таблицу символов максимального размера и текстовой файл, который Вы, вероятно, будете использовать в текущем сеансе. Поэтому загружать GENS4 часто удобнее в нижние адреса памяти.

Существует 2 версии ассемблера на кассете, обе на стороне 1. Сначала помещена версия с 51 символом в строке, за ней следует рассматриваемая версия с 32 символами. Их именами на ленте являются соответственно GENS4-51 и GENS4 и Вы должны использовать ту версию, которая больше Вам подходит. Версия GENS4-51 на 400 байт длиннее, чем GENS4.

Для загрузки GENS4 поместите магнитную ленту с ним в кассетный магнитофон и введите:

```
LOAD ""  CODE XXXXX [ENTER]
```

и нажмите клавишу "воспроизведение" магнитофона.

Или:

```
LOAD "GENS4" CODE XXXXX [ENTER]          или:
LOAD "GENS4-51" CODE XXXXX [ENTER]       где:
```

XXXXX - десятичный адрес, с которого Вы хотите загрузить GENS4.

Только после того, как Вы загрузите коды GENS4 в компьютер, Вы можете войти в ассемблер, набрав:

RANDOMIZE USR XXXXX, где

XXXXX - адрес, с которого загружены коды ассемблера.

Если в любой последующий момент Вы вновь пожелаете войти в ассемблер, то Вы должны просто адресоваться к ячейке xxxxx, которая хранит предварительно созданный файл информации.

Например, Вы хотите загрузить ассемблер с десятичного адреса 26000. Для этого наберите:

```
LOAD "" CODE 26000 [ENTER]
RANDOMIZE USE 26000 [ENTER]
```

Для перезапуска ассемблера используйте:

RANDOMIZE USE 26000 в пределах бейсика.

Только после ввода GENS4 на экране возникнет подсказка в виде символа ">" (подсказка команд редактора).
Раздел 3 содержит информацию о том, как нужно вводить и редактировать текст, а раздел 2 - о том, что нужно вводить.

1.2. Получение копии системы

После загрузки GENS4 в память Вашего SPECTRUMа Вы можете получить копию ассемблера следующим образом:

```
SAVE "GENS4-51" CODE XXXXX,11392 [ENTER]          или для
SAVE "GENS4" CODE XXXXX,11010 [ENTER]             кассеты
SAVE * "M";1;"GENS4-51" CODE XXXXX,11392 [ENTER]  или
SAVE * ";1;"GENS4" CODE XXXXX,11392 [ENTER        для микро-
                                                  драйвера,
```

где: XXXXX - адрес, с которого Вы загрузили GENS4.

Вы должны сделать эту копию перед входом в GENS4 для того, чтобы сохранить перемещаемую информацию внутри программы.
Инструкции получения копии для DEVPAC серии PLUS смотрите на дополнительном листке.
Пожалуйста, заметьте, что мы разрешаем Вам сделать копию GENS4 для Вашего собственного пользования, чтобы Вы могли уверенно программировать. Пожалуйста, не делайте копии GENS4 для передачи (или, что еще хуже, для продажи) Вашим друзьям. Мы выпускаем программное обеспечение по очень разумной цене и полной послепродажной поддержкой, но если люди сами начнут копировать наше программное обеспечение, то мы перестанем продолжать это делать. Пожалуйста, покупайте, но не крадите!

Раздел 2. П о д р о б н о с т и G E N S 4

2.0. Принцип работы GENS4, ключи ассемблера, формат листинга

GENS4 является быстрым двухпроходным ассемблером для Z80, который ассемблирует всю стандартную мнемонику для Z80 и содержит следующие особенности: макросы, условное ассемблирование, множество команд ассемблера и таблицу символов в виде двоичных троек.

Когда Вы вызываете ассемблирование, то Вы используете команду "A":

A4,2000,1:TEST [ENTER]

Первый код (4 - см. выше) после a определяет условия трансляции, которые Вы хотите иметь в данный момент. Эти условия трансляции и их ключи будут перечислены ниже. Если Вы не хотите использовать ключи,то номер набирать не надо, поставьте только запятую.

Второй код (2000 - см. выше) - это десятичный размер символов в байтах. По умолчанию (при простом использовании запятой без кодов), GENS4 будет выбирать размер таблицы символов, подходящих к размеру исходного файла - обычно это бывает вполне приемлемо. Однако, при использовании ключа "включить" Вы можете задать размер таблицы символов больше обычного, ассемблер вэтом случае не будет определять размер таблицы символов, которая будет построена.

После указания размера таблицы символов Вы можете ввести имя файла, например, 1:TEST, как указано выше. Если Вы это сделаете, то результирующий обьектный код, полученный в результате ассемблирования будет сохранен автоматически, независимо от размера генерируемого кода. Если Вы не хотите использовать такую возможность, то имя файла не вводите. Не вводите вторую запятую, иначе GENS4 будет считать, что Вы вводите пустое имя файла! Более подробно с командой "A" можно познакомиться в приложении 3.

Ключи ассемблирования:

1 - выдает листинг таблицы символов в конце второго прохода ассемблирования;

2 - запрет на генерацию обьектного кода;

4 - производит листинг ассемблирования (отметим, что в ранних версиях ассемблера такой возможности не было);

8 - направить листинг ассемблирования на печать;

16 - просто помещает обьектный код, если он, конечно, получен, после таблицы символов. Счетчик адресов обновляется таким образом, что обьектный код может быть помещен в одну область памяти, а работать - в какой-либо другой;

32 - исключение проверки того, куда помещается обьектный код (это бывает полезно для ускорения ассемблирования);

Для того, чтобы скомбинировать ключи, просто складывайте их один с другим. Так, код "A33" производит быстрое ассемб-

лирование - не выдается листинг, не производится проверка, чтобы увидеть, куда в настоящий момент помещают обьектный код, при этом в конце выдается листинг таблицы символов.

Отметим, что при использовании ключа "A16" директива "ENT" не будет иметь эффекта. Если использован ключ "A16", то Вы можете определить, где находится обьектный код, используя команду редактора "Y". По этой команде можно определить конец текста (его называет второй номер) и затем добавить к нему назначенный размер таблицы +2.

Ассемблирование происходит за два прохода: в течение первого прохода GENS4 ищет ошибки и собирает таблицу символов, второй проход генерирует обьектный код (если не указан ключ 2). В течение первого прохода на экране и принтере ничего не отображается, пока не встретится ошибка. В случае ошибки будет выдано сообщение об ошибке с номером ошибки за ним (см. Приложение 1). Ассемблирование приостанавливается - нажмите клавишу "E" для возврата в редактор или любую другую клавишу для продолжения ассемблирования следующей строки.

В конце первого прохода будет выдано сообщение:

```
PASS 1 ERRORS: NN
```

Если хоть одна ошибка будет обнаружена, то ассемблирование затем будет остановлено и перехода ко второму просмотру не произойдет. Если какие-нибудь метки были обнаружены в поле операнда, но не обьявлены в поле метка, то последует сообщение:

```
*WARNING* LABEL ABSENT
```

В течение второго прохода генерируется объектный код (если генерация не запрещена ключом 2 (см. выше)). Во время второго прохода листинг ассемблера не производится, кроме случая, когда указан ключ 4 или команда ассемблера L+.

В 32-х символьной версии листинг ассемблера состоит обычно из двух строк в следующей форме:

```
c000  210100    25      LABEL
                        LD HL,1
1     6         11      21...26
```

Тогда как в 51-символьной версии листинг печатается в одну строку. Если листинг не помещается в одну строку, то он завершается на следующей.

На первом месте в строке стоит значение счетчика адресов, который соответствует данной строке. Если в строке встречаются инструкции ORG, EQU или ENT (см. 2.7.), то на первом месте будет стоять значение поля операнда инструкции. Эта запись обычно отображается в шестнадцатеричном виде, хотя используя команду "*D+" ее можно отобразить и в неудобном десятичном виде (см. 2.9.).

Следующая запись с позиции 6 занимает до 8 символов в строке (представляет до 4-х байтов) и является обьектным кодом текущей инструкции (см. ниже команду *C).

Затем следует номер строки - целое число в диапазоне 1...32767.

Позиции 21...26 первой строки содержат 6-символьную метку, определенную в этой строке.

После метки следует мнемоника инструкции, которая занимает поз 21...24. (в 32-символьной версии это будет новая строка, если только не использовать команду *C).

Затем, с позиции 26, следует поле операндов; и завершают строку комментарии, которые при необходимости можно продолжить на другой строке.

Команда *C ассемблера может быть использована для получения короткой строки листинга ассемблирования - ее вводят, чтобы не включать 9 символов, представляющих объектный код инструкции. Это дает возможность отобразить большинство строк ассемблера в виде, удобном для экрана со строкой в 32 позиции (см. ниже разд. 2.8.).

Изменение формата листинга (только для 32-символьной версии)

С помощью команды "POKE" Вы можете модифицировать форму, по которой каждая строка листинга разбивается внутри 32-символьной версии GENS4 на 3 части. Указания, как это сделать, приведены ниже. Мы отличаем понятие "строка ассемблера", которая является текущей строкой листинга ассемблера и хранится во внешнем буфере, от понятия "экранная строка", которая является строкой, действительно возникающей на дисплее. Строка ассемблера обычно порождает более одной экранной строки.

1. 51 (#33#)-й байт от начала GENS4 содержит значение, определяющее позицию, до которой будет отображаться строка листинга на экране. Циклически меняя этот байт от 0 до любого другого значения (<256) можно заканчивать первую экранную строку в необходимом месте (это особенно полезно при использовании микроформатного принтера).

2. 52-й (#34) байт от начала GENS4 дает номер позиции, с которой должна начинаться каждая последующая экранная строка.

3. 53-й (#35) байт от начала GENS4 указывает количество символов остатка строки ассемблера, который будет выведен во второй экранной строке после первой.

Например, Вы хотите, чтобы каждая первая экранная строка содержала 20 символов (т.е. не включала бы поле метка) и затем каждая последующая экранная строка начиналась с позиции 1 и занимала бы всю строку. Также предположим, что Вы загрузили GENS4 с десятичного адреса 26000. Чтобы получить вышеуказанные изменения, надо из бейсика ввести следующие инструкции:

```
POKE 26051,20
POKE 26052,1
POKE 26053,31
```

Вышеприведенные модификации возможны только в том случае, когда не была использована команда *C.

Листинг может быть приостановлен в конце строки нажатием [CAPS SHIFT] и [SPACE] вместе. В последующем нажимайте клавишу "E" для возврата в редактор или любую другую клавишу для продолжения листинга.

Единственными ошибками, которые могут появиться во втором проходе, являются ошибки типа "*ERRAR*10" (см. прил.1) и "BAD ORG!" (которая появляется, когда объектный файл накладывается на GENS4, исходный файл или на таблицу символов - это может быть предотвращено использованием другого формата). *ERROR*10 - это не фатальная ошибка, поэтому Вы можете

продолжать ассемблирование, как и на первом проходе, тогда как ошибка "BAD ORG!" Является фатальной и сразу же возвращает управление редактору.

В конце второго прохода на дисплее будет отображено сообщение, следуя за предупреждением об отсутствующих метках:

```
PASS 2 ERRORS:NN
```

затем появится следующее сообщение:

```
TABLE USED: XXXXX FROM YYYYY
```

Это сообщение информирует о том, какая часть таблицы символов была заполнена. В этом случае, если была правильно использована директива "ENT", выдается сообщение:

```
EXECUTER: NNNNN IS DISPLAYED
```

Это показывает стартовый адрес задачи. Вы можете выполнить задачу, используя команду редактора "R". Будьте осторожны при использовании команды "R" в том случае,если Вы не завершили успешно ассемблирование и не увидели сообщения "EXECUTES:NNNNN".

Итак, если указан ключ "1", то используется список меток в алфавитном порядке и будут напечатаны соответствующие им значения. Количество информации, отображаемой в одной строке, может быть изменено директивой "PAKE" с адреса "начало GENS4 + 50" помещением в эту ячейку уместного значения, по умолчанию заносится 2.

Контролируйте возврат в редактор.

2.1. Формат оператора ассемблера

Каждая строка текста, которая будет обрабатываться GENS4, должна иметь следующий формат, где определение поля необязательно:

```
метка     код команды     операнды     комментарий
START         LD          HL,LABEL;    получение метки
```

Пробелы и табуляция обычно игнорируется. Строка обрабатывается следующим образом:

Первый символ строки проверяют и последующие действия зависят от его сущности:

; - целую строку трактуют как комментарий, т.е. простоигнорируют;

* - предполагается, что следующий символ (символы) образуют команду ассемблера (см. пункт 2.8.). Все символы после команды трактуются как комментарий;

<CR> - символ конца строки. Линия просто игнорируется;

_ - пробел или табуляция. Если первым символом строки является пробел или табуляция, то GENS4 предполагает, что следующий отличный от пробела или табуляции символ будет началом кода команды Z80.

Если первый символ строки отличается от вышеприведенных,

то ассемблер предполагает, что в строке должна быть метка - см. П.2.2.

После обработки присутствующей метки (или если первый символ в строке - пробел или табуляция) ассемблер исследует следующий значащий символ и предполагает, что он будет или символом конца строки, или началом команды Z80 (см. прил.2), Состоящей из 4-х символов и ограниченной пробелом (табуляцией или <CR>).

Иногда присутствует код команды и требуется один или более операндов, тогда поле операндов следует после набора пробелов (табуляций).

Оператор может состоять только из одной, это бывает полезно для улучшения читаемости листинга.

Комментарии могут следовать в любом месте после поля операндов или после поля кода команды (если у нее нет аргументов).

2.2. Метки

Метка - это символьное имя, представляющее 16 бит информации. Метка может быть использована для выделения адреса особенной инструкции, или для выделения области данных, или может быть использована как константа для директивы "EQU" (см. п. 2.7.).

Если метк представить значением более 8 битов и она затем используется в контексте применительно к 8-битовой константе, то ассемблер будет выдавать сообщение об ошибке:

```
LABEL EQ #1234
      LD A,LABEL
```

Будет выдано *ERROR*10, когда во время второго прохода будет проходить обработка второго оператора. Метка может содержать любое количество символов (см. ниже) хотя только первые 6 символов - значащие. Эти первые 6 символов должны быть уникальными, иначе метка будет повторно определена (*ERROR*4). Метка не должна образовывать зарезервированное слово, хотя зарезервированное слово можно внедрить как часть метки.

В метках можно использовать символы 0...9, A...Z и $. Можно использовать как большие, так и малые буквы, символы [, \, ^, #, и _. Метка должна начинаться с буквы. Некоторые примеры возможных меток представлены ниже:

```
LOOP
A_LONG_LABEL
L[1]
L[2]
A_
LDIR      - это незарезирвированная инструкция
TWO^S
```

2.3. Счетчик адресов

Ассемблер содержит счетчик адресов так, что имя в поле метки ассациируется с адресом и заносится в таблицу символов. Этот счетчик адресов может быть установлен в любое значение директивой "ORG" (см. п. 2.7.).

Символ $ может использоваться для передачи текущего значения счетчика адресов, например, команда

```
LD HL,S+5
```

генерирует код, загружающий регистровую пару HL значением, равным текущему значению счетчика +5.

2.4. Таблица символов

Когда метка встречается в первый раз, ее помещают в таблицу вместе с двумя признаками, которые позже показывают, как эту метку соотносят по алфавиту с другими, находящимися внутри таблицы. Если в первый раз метка появляется в поле меток, то ее значение (данное счетчиком адресов или значение выражения директивы "EQU") заносят в таблицу символов. В другом случае значения заносятся, когда бы ни встретилось имя метки в последующем в поле метки.

Данный тип таблицы символов называется таблицей символов типа двоичного дерева и ее структура дает возможность заносить имена и извлекать их за очень короткое время - это существенно для больших программ.

Запись в таблице занимает от 8 до 13 бит, в зависимости от длины имени.

Если за время первого прохода имя встречается более одного раза, то будет выдано сообщение об ошибке (*ERROR*4), так как ассемблер не знает, какое значение должно соответствовать имени метки.

Если значение для имени не найдено, то в конце ассемблирования будет выдано сообщение:

```
*WARNING* SYMBOL ABSENT
```

Отсутствие определения имени не мешает продолжению ассемблирования.

Отметим, что только первые 6 символов метки вводятся в таблицу символов, что определяется ее размером. В конце ассемблирования Вам могут быть выданы сообщения статистики, о том, как много памяти было использовано таблицей символов в течение трансляции - Вы можете изменить максимальный размер памяти, отведенный под таблицу символов.

2.5. Выражения

Выражения - это запись операндов, образованная или простым термом, или комбинацией термов, разделенных оператором. Ниже определены понятия терма и оператора:

терм:

десятичная костанта	например:	1024
шестнадцатиричная константа	например:	$4a5
двоичная константа	например:	%100101
символьная константа	например:	"A"
метка	например:	L1029

Может также использоваться $ для обозначения текущего значения счетчика адресов.

Оператор:

```
* + - сложение;
  - - вычитание;
  & - логическое "AND" (и);
  @ - логическое "OR" (или);
  ! - Логическое "XOR"(исключиельное или);
  * - алгебраическое умножение;
  / - алгебраическое деление;
  ? - MOD-функция (A?B=A-(A/B)*B;
```

Замечание:

```
  # - используют для того, чтобы отметить начало 16-тирич-
      ного числа
  % - используют для того, чтобы отметить начало двоичного
      числа " - символьная константа.
```

При считывании числа (десятичного, шестнадцатиричного или двоичного) GENS4 выбирает последние значащие 16 бит числа (т.е. MOD 65536), например, 70016 станет 4480, и #5A2C4 становится #A2C4.

Обеспечивается широкий набор операторов, но их приоретет не соблюдается: выражения вычисляются строго слева направо. Операторы *, / и ? Приведены для версии с дополнительными возможностями и не являются частью данного набора выражений, который мог бы увеличить размер GENS4. Если выражение заключено в круглые скобки, то его представляют как содержимое адреса памяти. Так, в инструкции LD HL, (LOC+5) в регистровую пару HL загружается 16-ти битовой значение, содержащееся в ячейке памяти с адресом LOC+5.

Некоторые инструкции Z80 (JR и DLNZ) предполагают операнды, которые предполагают 8-ми битовые значения, а не 16-ти битовые - это называется относительной адресацией. Когда имеет место относительная адресация, GENS4 автоматически выделяет значение счетчика адресов следующей инструкции из значения, представленного в поле операндов текущей инструкции для того, чтобы получить относительный адрес для текущей инструкции. Область допустимых значений относительного адреса простирается от -128+значение счетчика адресов следующей инструкции до значения счетчика адресов следующей инструкции+127.

Если же Вы желаете определить относительный переход от значения счетчика адресов текущей инструкции, то Вы должны использовать символ $ (резервную инструкцию), за которым следует требуемое смещение. Относительно значения счетчика адресов текущей инструкции смещение должно находиться в диапазоне от -126 до +129 включительно.

Примеры выражений:

```
5000                        - LABEL
%1001101!%1011              - дает   1001110
#3456?#1000                 - дает   #456
4+5*3-8                     - дает   19
$-LABEL+8
2345/7-1                    - дает   334
"Y"-":"+7
(5*LABEL-#1000&%1111)
17@%1000                    - дает   25
```

Отметим, что пробелы могут быть помещены между термами и операторами и наоборот, но не внутри термов.

Если операция умножения получит ответ с абсолютным значением, большим чем 32767, то появится ошибка *ERROR*15, тогда как при делении на 0 - ошибка *ERROR*14, в противном случае переполнение игнорируется.

Вся арифметика использует вторую дополнительную форму, где любое число, большее чем 32767, представляется как отрицательное, например:

```
60000=-5536   (60000-65536)
```

2.6..Макроопределения

Макроопределения позволяют Вам писать более плотные, более значимые ассемблерные программы, но они должны использоваться осторожно и не должны конфликтовать с подпрограммами.

Макроопределение состоит из набора инструкций ассемблера вместе с именем макроопределения. Когда это имя в последующем используется в поле кода операции, то оно будет заменено на все инструкции ассемблера, составляющие это макроопределение.

Так, макроопределение NSUB может быть определено следующим образом:

```
NSUB    MAC
        OR    A
        SBC   HL,DE
        ADD   HL,DE
        ENDM
```

Когда бы мы в дальнейшем ни применили NSUB как код операции, оно будет генерировать три инструкции ассемблера: OR A; SBC HL,DE и ADD HL,DE.

Это спасает Вас от лишнего печатания и сделает Вашу программу легче для понимания, но Вы должны помнить, что при каждом появлении NSUB генерируется обьектный код, и, может быть, эффективнее использовать CALL для вызова вместо нее подпрограммы. Ниже мы приводим формат макроопределения и его вызов вместе с некоторыми примерами. Изучите их, пожалуйста, внимательно. Макроопределение имеет следующую форму:

```
имя         MAC
            ...
            ...
      Тело макроопределения
            ...
            ...
            ENDM
```

По имени макроопределения будет вызываться его текст когда это имя в последующем встретится в поле кода операции. "MAC" показывает начало макроопределения, а "ENDM" указывает на его конец.

Параметры макроопределения могут упоминаться внутри макроопределения путем использования знака равенства, за которым следует номер параметра (0...31) включительно.

Таким образом, макроопределение

```
MOVE    MAC
        LD      HL,=0
        LD      DE,=1
        LD      BC,=2
        LDIR
        ENDM
```

имеет три параметра, адрес источника, адрес приемника и длину, загружает нужные значения в HL, DE, и BC и затем выполняет инструкцию LDIR. Для того, чтобы вызвать макроопределение из тела Вашей программы, просто используйте его имя в поле команд. За именем макроопределения должны следовать необходимые Вам три параметра:

```
MOVE 16384,16385,4096
```

В этом примере мы используем особенные адреса, но в действительности Вы можете работать с любыми существующими выражениями для определения значения параметра макроопределения:

```
MOVE START,START+1, LENGTH
```

Подумайте, являются ли вышеприведенные примеры хорошим использованием макроопределения? Не могли ли Вы использовать подпрограмму?

Внутри макроопределения параметры могут появляться в любом возможном выражении, например:

```
HMC   . MAC
        LD      HL,=0*3600
        LD      DE,=1*60
        ADD     HL,DE
        LD      DE,=2
        ADD     HL,DE
        ENDM
```

Это макроопределение имеет три параметра: часы, минуты, секунды. Макроопределение вычисляет общее количество секунд, определяемое параметрами, и помещает его в регистр HL. Вы можете использовать его следующим образом:

```
HOURS     EQU 2
MINUTES   EQU 30
SECONDS   EQU 12
START     EQU 0

          HMC   HOURS,MINUTES,SECONDS
          LD    DE,START
          ADD   HL,DE             ;HL - окончательное время.
```

Макроопределения не могут быть вложенными, так что Вы не сможете ни определить макроопределение внутри макроопределения, ни вызвать макроопределение из макроопределения.

Во время ассемблирования, как только имя макроопределения встречается в поле команды, ассемблируется его текст. Обычно этот текст тела макроопределения не попадает в ассемблерный листинг и печатается только для макроопределения. Однако Вы можете усилить листинг расширением макроопределения, используя команду ассемблера *M+, а, используя *M-,

запретить распечатывать текст макроопределения.
Если Вы вышли за пределы пространства, отведенного под буфер макро, то будет выдано сообщение и прервется ассемблирование. Используйте команду редактора "C" для резервирования буфера больших размеров.

2.7. Директивы ассемблера

GENS4 распознает определенные псевдокоманды - так называемые директивы ассемблера. Они не влияют на процессор Z80 при прогоне, т.е. не декодируются, а просто заставляют ассемблер выполнять определенные действия во время трансляции. Эти действия определенным образом изменяют код, генерируемый GENS4.
Псевдокоманды ассемблера точно такие же, как и выполняемые инструкции: они могут помещаться за меткой (необходимо для EQU) и за ними может следовать комментарий.

Возможные директивы:

ORG <выражение> - устанавливает счетчик адресов равным значению выражения. Если ключи 2 и 16 оба не указаны и результат ORG попадает в область GENS4, текстового файла или таблицы символов, то появляется выражение "BAD ORG!" и прекращается трансляция. см. п.2.0. Для более детального ознакомления с тем, как ключи 2 и 16 влияют на использование ORG. см. команду "A" в разделе 3 для некоторых предосторожностей в использовании ORG, когда автоматически сохраняется обьектный код.

EQU <выражение> - этой директиве должна предшествовать метка, значению метки присваивается значение выражения. Выражение не должно содержать имен, которым еще не присвоено значение. В противном случае ассемблер выдаст Вам сообщение об ошибке - *ERROR*13.

DEFB <выражение>,<выражение>... - каждое выражение должно оценивать 8 бит; байт, адрес которого в настоящий момент содержится в счетчике адресов.

COB принимает значение выражения и счетчик адресов увеличивается на 1. Все вышеуказанное повторяется для каждого выражения.

DEFW <выражение>,<выражение>... - дополняет слово (2 байта), адрес которого указан счетчиком адресов, значением выраженияи увеличивает счетчик адресов на 2. Младший байт помещается сначала, за ним следует старший байт. Повторяется для каждого выражения.

DEFS <выражение> - увеличивает счетчик адресов на значение выражения. Эквивалент резервирования области памяти размером, равным значению выражения.

DEFM "S" - определяет, что последовательность N байтов памяти бедет равна представлению строки "S" в коде ASCII, где N - длина строки. Теоретически, длина стро ки может быть от 1 до 255 включителтно, но практически она ограничена длиной строки, которую можно

ввести редактором. Первый символ в поле операндов ", как показано выше, является ограничителем и строка "S" определяется только символами, лежащими между двумя ограничителями; символ конца строки также действует как ограничитель строки.

ENT <выражение> - эта директива никак не влияет на генерируемый объектный код, она просто используется для определения адреса, на который осуществляется переход по команде редактора "R". Выражение ENT устанавливает этот адрес равным значению выражения. Используется совместно с командой редактора "R"(см. разд.3). Для выполнимого адреса нет умолчания.

2.8. Команды условной трансляции

Команды условной трансляции обеспечивают программисту возможность включения или невключения определенных секций исходного текста в процесс ассемблирования. Это возможно при использовании команд IF, ELSE, END.

IF <выражение> - эта команда оценивает выражение. Если результат равен нулю, то прекращается ассемблирование следующих линий до тех пор, пока не встретится ELSE или END. Если значение выражений отлично от нуля, то ассемблирование происходит нормально.

ELSE - эта команда просто переключает ассемблирование. Если перед появлением ELSE было ассемблирование, то в последующем оно выключается и наоборот.

END - просто включает ассемблирование.

Отметим, что команды условной трансляции не могут быть вложенными, на вложенность "IF" не сделано никаких проверок и любая попытка вложения может привести к непредсказуемым результатам.

2.9. Команды ассемблера

Команды ассемблера подобно директивам ассемблера не влияют на работу процессора Z80 т.к. они не декодируются в обьектные коды. Однако, в отличие от директив ассемблера, они не влияют на обьектный код, производимый ассемблером - команды ассемблера просто модифицируют формат листинга. Команда ассемблера - это строка исходного текста, которая начинается с символа *.

Надпись после звездочки определяет тип команды и должна быть выполнена заглавными буквами. Ограничителем строки может быть любой текст. Исключение составляют команды L и D, предполагающие знаки "+" или "-" после команды. В распоряжении программиста имеются следующие команды:

*E - выдает 3 пустые строки на экран или принтер - это полезно для разделения модулей.

*HS - выдает строку "S" - заголовок, который будет печа-

таться после каждого вывода "*E". *H автоматически выполняет *E.

*S - указывает, что с данной строки листинг должен быть остановлен. Листинг может быть продолжен нажатием любой клавиши. Полезно для чтения адресов в середине листинга.

Примечание: если *S появляется после *L-, то листинг не останавливается.

*L+ - прекращает листинг и печать, начиная с данной строки.

*L- - начинает листинг и печать, с данной строки.

*D+ - указывает, что значение счетчика адресов в начале каждой строки должно выдаваться в десятичном виде вместо обычного 16-ичного.

*D- - возвращает к использованию шестнадцатиричного вида значения счетчика адресов в начале каждой строки.

*C- - укорачивает листинг ассемблера, начиная со следующей строки. Листинг характеризуется тем, что он не содержит обьектного кода, генерируемого текущей строки - это укорачивает строку листинга на 9 символов и делает строку ассемблера более удобной для дисплея с 32-мя символами в строке, что улучшает читаемость.

*C+ - возвращает к полному листингу, как описано в разделе 2.0.

*M+ - включает печать макроопределения.

*M- - выключает печать макроопределения.

*F (имя файла) - это очень мощная команда, позволяющая Вам ассемблировать текст с ленты или дисковода - этот текст читается с ленты или дисковода в буфер, блокируется там на время и затем ассемблируется из этого буфера: это позволяет создавать обьектный код большого размера, только если исходный ассемблируемый текст не занимает много места в памяти. Имя текстового файла (до 10 символов), который Вы желаете проассемблировать таким образом, не обязательно должно быть определено и ему должен предшествовать пробел. Если файл находится на кассете, то Вы указываете это, набивая перед именем файла номер привода с двоеточием:

```
*F 2:TEST   - для файла на микроприводе №2;
*F TEST     - для файла на ленте.
```

Если не указано имя файла, то с ленты считывается первый найденный файл, но при считывании с дисковода вышесказанное нельзя принимать в расчет. Если Вы работаете с дисководом, то файл должен быть предварительно сохранен с помощью команды редактора "P" обычным способом.

Если файл находится на ленте, то Вы должны предварительно загрузить его с ленты при помощи команды редактора "T", а

не "P" - это необходимо, так как текстовой файл, содержащийся на ленте, должен быть преобразован в блоки с достаточным размером межблочных интервалов, что позволяет ассемблировать текущий блок перед тем, как следующий начнет грузиться с ленты. Размер блока, используемого данной командой (и командой редактора "T"), устанавливается с помощью команды редактора "C" (см. следующий раздел). Возможность выбора размера этого буфера позволяет Вам оптимизировать отношение размер/скорость для любых выводов текстов с ленты. Например, если Вы не собираетесь использовать команду "F" во время ассемблирования, то полезно будет определить размер буфера равным 1 для минимизации пространства, занимаемого GENS4 и минимизации рабочего пространства.

Где бы ни встретил ассемблер конмаду "F", он определит, откуда нужно считывать файлы, с дискеты или с ленты. Это происходит на обоих просмотрах, так как на каждом просмотре необходимо сканировать текс если считывается лента, то на ней ищется файл с требуемым именем или первый файл.

Если имя файла во время его поиска на ленте не соответствует заданному, то выдается сообщение: "FOUND FILE" и поиск продолжается, в другом случае выдается сообщение "USING FILENAME", файл загружается блок за блоком и обрабатывается. Подробнее ознакомиться с "F" Вам поможет пример приложения 3.

Остальные команды ассемблера обрабатываются только на втором просмотре.

Если ассемблирование было прервано одной из команд условной трансляции, то и работа любой команды ассемблера также прерывается.

Замечание: описанные возможности не годятся для систем DISCIPLE, OPUS и PLUS 3 DEVPAC 4. Быстрее и легче вместо ленты использовать диск. Следствием этого является то, что в этих версиях команда "C" не позволяет Вам определить размер быфера загрузки и команда "T" не сеществует.

Раздел 3. Строчный редактор

3.1. Введение в редактор

Редактор, поставляемый со всеми версиями GENS4, является простым строчным редактором, разработанным для обслуживания всех операционных систем, сделанных для Z80. Редактор прост в использовании и дает возможность редактировать программы просто и эффективно.

Для уменьшения размера текстового файла редактор выполняет определенное сжатие пробелов. Это просходит по следующей схеме: когда вводится строка с клавиатуры, она символ за символом заносится во внешний относительно ассемблера быфер и затем, в конце строки (когда Вы нажимаете клавишу <ENTER>) строка перемещается из буфера текстовой файл Во время этого перемещения происходит определенное сжатие пробелов: если первый символ строки - пробел, то в текстовой файл вводится символ табуляции, и все последующие пробелы пропускаются. Если первый символ в строке не пробел, то символы заносятся из буфера в текстовой файл, пока не встретится пробел, после чего обработка ведется также, как если бы следующий символ был бы первым символом в строке, это продолжается и в дальнейшем. В результате символы табуляции включаются в начале строки или между меткой и кодом операции, а так же между кодом операции, операндами и комментариями. Конечно, если код возврата каретки <ENTER> встречается в любое другое время, то преобразование завершается и управление передается редактору.

Этот процесс сжатия понятен, и Вы можете просто использовать клавишу "-->" для того, чтобы получить кратко протабулированный файл, который также экономичней для хранения. Заметим, что пробелы не сжимаются внутри комментариев и что пробелы не должны присутствовать в полях меток, кодов операций и оперендах.

Режим редактора включается автоматически при запуске GENS4 и за вспомогательным текстом следует подсказка редактоа ">". В ответ на подсказку Вы можете ввести командную строку следующего формата:

C N1,N2,S1,S2 <ENTER> где:

"C" - мнемоника команды, которую необходимо выполнить;
N1 - число в пределах от 1 до 32767 включительно;
N2 - число в пределах от 1 до 32767 включительно;
S1,S2 - строка не более чем из 20 символов.

Запятая используется для разделения различных аргументов, хотя это можно изменить - см. команду "S" пробелы игнорируются во всех случаях кроме тех, когда они находятся внутри строк, накакой из аргументов не является обязательным, хотя, некоторые команды, например, команда "DELETE", не будут работать, если опущены аргументы N1 и N2

Редактор помнит ранее введенное число и строки и использует сформированное значение для применения, если Вы не определили другие значения внутри командной строки. Первоначально значения N1,N2 устанавливаются равными 10, а строки - пустыми.

Если Вы имели неправильную командную строку, например, F - 1,100,HELLO, то эта команда будет проигнорирована и

появится сообщение "PARDON?". При этом Вы должны ввести командную строку правильно,например - F1,100,HELLO.

Это же сообщение появится, если длина строки S2 превышает 20 символов. Если более 20 символов содержит строка S1, то лишние символы игнорируются.

Команды могут вводиться как на верхнем, так и на нижнем регистрах.

При вводе команды определенные комбинации ключей могут использоваться для ее редактирования. Так, клавиша "<--" используется для стирания символов в направлении начала строки, "-->" - для продвижения курсора в следующую позицию табуляции, "CAPS SHIFT 0 " или "DELETE" - для уничтожения предыдущего символа.

Следующий раздел представляет команды, используемые редактором. Заметим, что если аргумент заключен в квадратные скобки, то такой аргумент обязателен для данной команды.

3.2. Команды редактора

3.2.1. Вставка текста.

Текст может быть введен в текстовой файл или указанием номера строки, пробела и требуемого текста, или посредством команды "I". Заметим, что если сразу за номером строки Вы введете <ENTER>, то эта строка будет удалена из текста, если,конечно он существует. Где бы не вводился текст, можно применять клавиши "<--", "-->" и <eDIT> (возврат к метке команды).

Клавиша <DELETE> будет уничтожать предыдущий символ (но не далее начала строки текста).

Текст вводится во внешний буфер внутри GENS4 и Вы должны оградить его от переполнения использованием клавиш <DELETE> или "<--" для освобождения свободного пространства. Если во время ввода текста редактор определит, что конец текста перекрывает вершину адресуемой памяти, то выдается сообщение <BAD MEMORY!>. Это показывает,что далее текст вводить нельзя и что текстовой файл или последняя его часть должна быть сохранена на кассете для дальнейшего восстановления.

Команда " I N,M "

Использование этой команды переводит ввод в автоматический режим: Вам подсказываются номера строк, начиная С N С приращением м на каждом шагу. После высвечивающегося номера строки Вы вводите нужный текс, по желанию используя нужные клавиши, и завершаете строку текста вводом <ENTER>. Для выхода из этого режима используются <EDIT>.

Если Вы вводите строку с уже существующим номером, то строка текста с этим номером удаляется и заменяется на вновь введенную после нажатия <ENTER>.

Если автоматическое увеличение номера строки дает значение, большее 32767, то происходит автоматический выход из режима ввода.

Если при вводе текста Вы добрались до конца строки на экране, но еще не ввели 64 символа (размер буфера), то экран сдвинется вверх на одну строку и Вы можете продолжать ввод со следующей строки - номер строки автоматически будет соответствовать введенному тексту.

3.2.2. Распечатка текста

Текст может быть проверен с помощью команды "L", номер строки,постоянно отображенный при выполнении данной команды, устанавливается заранее, но он может быть изменен с помощью команды "K".

Команда " L N,M "

Эта команда выводит листинг со строки N до строки M включительно на терминал. По умолчанию N присваивается значение 1, M - 32767, т.е. значениями по умолчанию не являются ранее введенные аргументы.

Для листинга целого файла просто используйте команду "L" без аргументов. Строки на экране форматируются по левой границе экрана, так что номер строки отображается ясно. Строка протабулирована автоматически. В результате чего получаем четкое разделение полей в строке. Номер отображаемой на экране строки может быть проконтролирован с помощью команды "K" - после листинга определенного количества строк листинг будет приостановлен (даже если это еще не строка M). Для возврата в точку входа в редактор нажмите клавишу <EDIT> или любую другую клавишу для продолжения.

Команда " K N "

"K" устанавливает количество экранных линий, которые должны отображаться на терминале перед паузой, как это было описано выше (см. команду "L"). Значение N (не более 256) хранится в памяти. Например, если Вы хотите при последующем использовании "L" выдавать на экране по 5 строк, то введите команду "K5".

3.2.3. Редактирование текста.

Однажды созданный текст неизбежно будет нуждаться в редактировании некоторых строк. В GENS4 имеются команды, позволяющие исправлять, стирать, перемещать и перенумеровывать строки.

Команда "D [N,M] "

все линии от N до M включтельно удаляются из текстового файла. Если M<N или определено менее двух аргументов то команда игнорируется. Это сделано для предотвращения ошибок из-за небрежности. Одиночная строка может быть уничтожена указанием N=M. Этого же можно достичь простым введением номера строки, за которым следует <ENTER>.

Команда " M N,M "

Эта команда помещает текст строки с номером N в строку M, уничтожая текст, уже существующий там. Т.е. эта команда позволяет Вам перемещать строки текста внутри текстового файла. Если строки с номером N не существует, то команда игнорируется.

Команда " N [N,M] "

Команда "N" перенумеровывает M строк текстового файла, начиная со строки N. N и M должны быть реальными и если но-

мер линии превышает 32767, то остается первоначальная нумерация.

Команда " F N,M,F,S "

Текст со строки N до строки M исследуется на появление строки F. Если такая строка найдена, то она отбражается на терминале и включается режим "EDIT" (см. ниже).

Вы можете использовать команды внутри режима EDIT для нахождения последующих появлений строки F в пределах определенной области строк или для замещения строкой S текущего появления строки F и затем продолжить поиск строки F. Отметим, что диапазоны строк F и S могут быть установлены ранее другой командой, так что возможно вводить только F для инициализации поиска (см. пример в разделе 3.3).

Команда " E N "

Редактирует строку с номером N. Если строки N не существует, то команда игнорируется; в противном случае строка копируется в буфер и отображается на терминале вместе с номером строки. Номер строки вновь отображается под строкой и включается режим редактирования. Все последующее редактирование происходит внутри буфера, а не внутри самого текста, так что первоначальная строка может быть получена в любой момент.

В этом режиме курсор отображается движущимся по строке (начиная с первого символа) подддерживая различные подкоманды, позволяющие редактировать строку. В Вашем распоряжении имеются следующие подкоманды:

<SPACE> - перемещает курсор на одну позицию к следующему символу в строке. Вы не можете шагнуть за конец строки.

<DELETE> - возвращает курсор на предыдущий символ в строке. Невозможно шагнуть левее первого символа строки.

< --> > - Перемещает курсор на следующую позицию табуляции в каждой экранной строке.

<ENTER> - конец редактирования данной строки. Сохраняет все сделанные изменения.

<Q> - выход из режима редактирования данной строки, т.е. покидание редактируемой строки с игнорированием всех сделанных изменений. Строка остается такой же, как она и была до редактирования.

<R> - перезагружает буфер редактирования текстом, т.е. забывает все сделанные в строке изменения и восстанавливает строку в первоначальном виде.

<L> - распечатывает остаток строки, который должен быть отредактирован, т.е. остаток строки за текущей позицией курсора. Вы остаетесь в режиме редактирования с указателем, перепозиционированным в начало строки.

<K> - уничтожает символ в указываемой курсором позиции.

<Z> - уничтожает все символы, начиная с указанного кур-

сором и до конца строки.

<F> - ищет следующее появление строки "F", ранее определенной в командной строке (см. команду "F" выше). Эта подкоманда будет автоматически выводить систему из режима редактирования текущей строки (сохраняя все изменения) даже если цепь символов "F" в текущей строке не встретилась. Если цепочка "F" встретится в последующих строках текста внутри первоначально определенного диапазона, то будет включен режим редактирования для строки, в которой найдена заданная последовательность символов "F". Отметим, что курсор всегда устанавливается в начало найденной строки.

<S> - замещает ранее орределенной строкой "S" текущее появление цепи символов "F" и затем выполняет подкоманду "F", т.е. осуществляет поиск следующей строки "F". Так, вместе с вышеупомянутой командой " F " она используется для пошаговой замены строк символов "F" строками "S". См. разд. 3.3 для примера.

<I> - вводоит символ в указанную курсором позицию. Вы будете оставаться в этом режиме до тех пор, пока не нажмете <ENTER>. По этой клавише Вы возвращаетесь в основной режим редактирования с указывающим на последний введенный символ. Используя <DELETE> внутри этого режима, Вы уничтожите символ в буфере слева от курсора,тогда как,используя <-->> переместите курсор в следующую позицию та буляции, включая пробелы внутри буфера.

<X> - перемещает курсор в конец строки и включает описанный подрежим.

<C> - изменяет подрежим. Это позволяет перезаписать символ в текущей позиции затем передвигать курсор через одну позицию. Вы останетесь в измененном подрежиме до тех пор, пока не нажмете клавишу <ENTER>, нажатие которой возвратит Вас в основной режим редактирования с курсором, указывающим на последний измененный символ. <DELETE> просто сдвигает через одну позицию курсор влево, а <-->> не имеет эффекта.

3.2.4. Команды дисковода и магнитной ленты.

Текст может быть сохранен на магнитной ленте/дисководе или загружен с них с помощью команд P,Q и T. Объектный код может быть сохранен на магнитной ленте с использованием команды O. Имена файла не должны содержать более 10 символов.

Команда " P N,M,S "

Строки с номерами от N до м сохраняются на ленте/диске в файле с именем, заданным строкой S. Текст будет записан на диск, если перед именем файла через двоеточие стоит номер дисковода. Помните что эти аргументы могут быть установлены предыдущей командой.

Примеры:

P 10,200,EXAMPLE - записать линии 10...200 на магнитофон в файл с именем "EXAMPLE".

P 500,900,1:TEST - записать линии 500...900 на дисковод 1.

Перед введением этой команды убедитесь, что Ваш магнитофон включен в пехим записи. Не используйте эту команду, если Вы желаете на последующей стадии ввести текст с ленты. Вместо этого используйте команду T. Если Вы намерены вводить текст с дисковода, то используйте команду P.

Если Вы заносите на диск файл с уже имеющимся на диске именем, то Вас спросят:

FILE EXIST DELETE (Y/N)? (Сущуствующий файл удалять?)

Отвечайте "Y" для удаления файла и продолжения записи или нажмите любую другую клавишу для возврата в редактор без записи файла.

Команда " G,,S "

На ленте или дискете производится поиск файла с именем "S". Когда файл найден, он загружается в конец текущего текста. Если строка "S" пустая, то загружается первый файл с кассеты. Для дисковода обязательно надо указать имя файла и номер дисковода.

При работе с касетой, после того как Вы ввели команду "G", появится сообщение :

START TAPE

Вы должны нажать клавишу "воспроизведение" своего магнитофона. Ведется поиск файла с указанным иенем или первого файла по умолчанию. Когда нужный файл найден, появляется сообщение:

USING FILENAME

В противном случае высветится сообщение:

FOUND FILENAME

и поиск на ленте продолжается. При использовании дисковода и в том случае, если не найден искомый файл, то появится сообщение "ABSENT".

Заметим, что если в памяти уже находится текстовой файл, то файл, загружаемый с ленты, будет добавлен к существующему и строки всего совокупного файла будут перенумерованы, начиная с первого, с шагом 1.

Команда T N,M,S

выводит блок текста между строками N и M включительно на ленту в формате, подходящем для дальнейшей работы под управлением директивы ассемблера *F - см. раздел 2.9. обрабатывается файл с именем "S". Вывод начинается сразу после нажатия клавиши <ENTER>, так что Вы должны убедиться, что Ваш магнитофон готов к записи перед вводом этой командной линии. Если

Вы намерены осуществлять ввод с дисковода, то используйте лучше команду "P", а не "T". Отметим, что эта команда должна использоваться только в том случае, если Вы хотите позднее ассемблировать текст с ленты. Это неприменимо для диска - только в версиях DEVPAC 4.

Команда " O ,,S "

выводит Ваш обьектный код на кассету или дисковод. Имя файла может иметь длину более 8 символов и должно начинаться с номера привода (1...8) И двоеточия, если Вы хотите сохранить обьектный код на дискете.

Только последний блок кода, произведенный ассеблиром, может быть сохранен этим способом, т.е. если у Вас более одной директивы "ORG" в исходной программе, то сохранится только код, произведенный после последней директивы "ORG".

Код должен быть получен в памяти перед тем, как он может быть сохранен с помощью команды "O". Часто удобнее использовать команду "A", "FILENAME" для автаматического получения файла. Эта команда работает быстрее "O" - см. ниже.

3.2.5. Ассемблирование и запуск из редактора.

Команда " A O,S,F "

Эта команда ассемблирует текст с первой строки текстового файла. Команда "O" позволяет Вам задать ключи,которые должны использоваться во время этой трансляции. Обычно достаточно использовать значения ключей по умолчанию, просто употребляя запятые.

S - дает возможность изменить размер таблицы символов для этой трансляции. Размер таблицы по умолчанию обычно бывает достаточным, за исключением режима ввода (INCLUDING).

F - имя файла, имеющегося на дискете. Имя файла должно начинаться с <D:>, где D - номер дисковода, на котором размещен файл. Имя файла не должно содержать более 10 символов. Присутствие имени файла здесь побуждает ассемблер производить трансляцию иным способом.

Вместо простого ассемблирования обьектного кода в память и остановки по достижении вершины памяти, ассемблер не будет транслировать в память до тех пор, пока не достигнет ее вершины (верхнюю границу памяти Вы можете установить командой "U") и затем полученный обьектный код будет сохранен на диске в указанном Вами файле. Затем ассемблирование будет вновь продолжено с нижних адресов памяти и этот процесс будет продолжаться до тех пор, пока вся программа не будет проассемблирована и сохранена на дискете.

Не существует ограничений на размер программы, которую Вы хотите проассемблировать (кроме размера существующего пространства на Вашей дискете (кассете)).

Пара важных замечаний относительно использования директивы "ORG" для данных средств:

1 - директива "ORG" будет помещать обьектный код по адресам, указанным в данной директиве первоначально и каждый раз после помещения обьектного кода в обьектный файл (если

не был указан ключ 16, чтобы поместить обьектный код сразу после тадлицы символов). Следовательно, ключ 16 более разумно использовать будет в том случае, когда ассемблирование происходит прямо на дмскету, т.к. это дает максимальный размер для Вашего буфера обьектного кода; рабочие адреса Вашего кода не будет поражены.

2 - Вы должны избегать использования более одной директивы "ORG" в Вашей программе, если только Вы не введете нули в память между двумя этими директивами с помощью "DEF".

Например:

```
ORG 50000
; некоторые коды
RET
ORG 60000
;еще коды
```

Этот фрагмент не будет сохранен на ленте правильно, т.к. вторая директива "ORG" переопределяет начало буфера обьектного кода. Но:

```
ORG 50000
; некоторые коды
RET
; добиваем нулями до 60000
DEFS 60000-$
; еще коды
```

Фрагмент будет сохранен правильно, т.к. DEF-$ произведет достаточное количество нулей, чтобы последующий код начинался с адреса 60000.

Очевидно, это неэффективно применительно к качеству кода, сохраняемого на дискете, но простота этого действия сохраняет ассемблер небольшим и быстрым.

Примеры использования команды "A":

```
A20,,1:TEST <ENTER>
```

ассемблирует с листингом, помещает обьектный код сразу после таблицы символов (это увеличивает размер буфера обьектного кода в памяти), использует размер таблицы символов по умолчанию и сохраняет обьектный код на дисководе 1 под именем "TEST".

```
A,3000 <ENTER>
```

ассемблирует программу, используя ключи по умолчанию, с размеров таблицы символов 3000 байт. см. раздел 2 для дольнейшей детализации того, что может случиться при ассемблировании.

Команда "R",

если исходная программа была проассемблирована без ошибок и рабочие адреса были определены с помощью директивы "ENT", то может быть использована команда R для выполнения обьектной программы. Обьектная программа может использовать инструкцию RET (#C9) для возврата в редактор сколько угодно раз если стек в конце выполнения программы будет таким же, каким он был в ее начале.

Заметим, что директива "ENT" не будет действовать, если для ассемблирования использовался ключ 16.

Прерывания перед вводом кода разрешены и регистр IY загружен значениями # 5C3A, важным для подпрограммы ослуживания прерываний SPECTRUM ROM.

3.2.6. Другие команды

Команда "B"

просто возвращает управление операционной системе. Для перезапуска ассемблера используйте

```
RANDOMIZE USR XXXXX
```

где: XXXXX - адрес, с которого был загружен GENS4.

Команда "C"

Позволяет Вам изменить размер входного буфера и буфера макроопределений (только для версий DEVPAC на магнитной ленте).

Входной буфер - это буфер, в котором содержится текст, когда происходит трансляция непосредственно с кассеты или дискеты - чем больше этот буфер, тем боьлшой текст может быть считан с кассеты или дискеты и, следовательно, тем быстрее будет происходить трансляция. Но, с другой стороны, используется большая память. Однако возможен компромисс между скоростью трансляции и используемой памятью, команда "C" позволит Вам управлять этим процессом, предоставляя Вам возможнлсть установки размера входного буфера.

Буфер макроопределений используется для хранения текста макроопределений, которые Вы могли использовать.

Команда "C" печатает подсказку на ввод размера входного быфера, а затем - на ввод размера быфера макроопределений. В обоих случаях вводить надо просто число десятичное байтов, которое Вы желаете зарезервировать, и <ENTER>. Если Вы нажали <ENTER> без ввода числа, то опереция игнорируется. Определяемый Вами размер входного буфера не должен быть меньше 256 байт. Вы можете прервать команду с помощью <CAPS SHIFT>1.

Отметим, что для дисковых версий DEVPAC 4 Вы можете менять только размер буфера макроопределений.

Команда "C" не уничтожает Ваш текст, она просто сдвигает его вверх и вниз в памяти в зависимости от размера буфера. Лучше зарезирвировать буферы такого размера, какой нужен в начале сеанса.

Команда "S,,D"

Эта команда позволяет Вам изменить символ-разделитель аргументов в командной строке. На входе редактора таким разделителем является запятая, это может быть изменено с помощью команды "S" на первый символ орпеделяющей строки "D".

Помните, что однажды определив новый разделитель, Вы должны его использовать до тех пор, пока не определите новый, даже внутри команды "S". Заметьте, что разделителем не может быть пробел.

Команда " U N "

Позволяет установить верхнюю границу памяти равной N. Если N

не указать (т.е. ввести только U и <ENTER>), то отображается текущая верхняя граница памяти.

GENS4 не позволяет Вашему текстовому файлу или объектному коду распространяться выше верхней границы памяти и будет выдавать сообщение об ошибке при приближении к этой границе.

По умолчанию верхняя граница памяти принимается равной вершине стека памяти системы SPECTRUM.

Команда " V "

выдает на дисплей полезную информацию: значение параметров команды N1 и N2 по умолчанию;символ-разделитель команды по умолчанию; десятичное значение начала и конца текста и значение первой командной строки S1.

Команда " W N,M "

выводит строки текста С N пО М включительно на принтер. Если ни N, ни М не указаны, то будет напечатан весь файл. Печать будет приостановлена после вывода некоторого числа линий, определенного командой "K" - нажмите любую клавишу для продолжения печати.

Команда " X N "

выдает каталог диска. В версии с 51-м символом перед выдачей каталога происходит очистка экрана. Каталог всегда распечатывается в строку из 32-х позиций.

Команда " Z "

Эффективно уничтожает весь Ваш текст, но перед этим она спрашивает, уверены ли Вы в необходимости этого. Отвечайте "Y" для уничтожения текста.

Кроме быстрого уничтожения командой D1,32767 команда "Z" позволяет Вам очистить Ваш текстовый файл, если он каким-либо образом был испорчен.

Команда "Z" устраняет неоходимость для стартовой точки входа GENS4.

Команда " H "

выдает подсказку на экран: список возможных команд в 2 столбца с заглавной буквой,►обозначающий команду и отображающий текущее значение определенных важных параметров.

3.3. Пример использования редактора

Предположим, что Вы ввели следующую программу (используя I10,10):

```
10    *H     16 BIT  RANDOM  NUMBERS
20
30    ;INPUT: HL CONTAINS PREVIOUS RANDOM NUMBER
40    ;OUTRUT: HL CONTAINS NEW RANDoN OR SEED NUMBER
50
60    RANDOM PUSH AF ; SAVE REGISTERS
70    PUSH BC
80           PUSH HL
90           ADD HL,HL ; *2
100          ADD HL,HL ; *4
             ............ ...
             ..............
             ..............
140          ADD  HL,HL  ; *64
150          PIP  BC     ; OLD RANDOM NUMBER
160          ADD  HL,DE
170          LD   DE,L1
180          ADD  HL,DE
190          POP  BC     ; RESTORE REGISTERS
200          POP  AF

210          REY
```

Эта программа содержит несколько ошибок:

строка 40: вместо "RANDOM" набрано "RANDON";

строка 70: PUSH BC начинается с поля метки;

строка 150: PIP вместо POP;

строка 160: требуется комментарий
(это не ошибка, а недостаток);

строка 210: вместо REY должно быть RET.

Также должны быть добавлены две команды ADD HL,HL между строками 140 и 150. Кроме того, все ссылки на пару регистров DE в строках 160...180 Должны быть ссылками на BC

Для внесения этих исправлений мы должны выполнить следующее:

```
F40,40,RANDON,RANDOM <ENTER>
        затем подкоманду "S"

E70 <ENTER> затем I (режим вставки),
            один пробел и <ENTER><ENTER>

I142,2<ENTER> 142 ADD HL,HL ; *128
              144 ADD HL,HL ; *256
              <EDIT>

F150,150,PIP,POP <ENTER>
            затем подкоманду "S"
```

E160 <ENTER> затем X,2 пробела; *257 +L1 <ENTER><ENTER>

E160,180,DE,BC <ENTER>
затем повторное использование подкоманды "S"

E210 <ENTER> затем 2 пробела,C,T <ENTER><ENTER>

N10,10 <ENTER>, чтобы перенумеровать текст.

Рекомендуем Вам хорошенько проработать вышеприведенный пример, используя редактор.

П р и л о ж е н и е 1 . Коды ошибок и их значения.

*ERROR*1 - ошибка в контексте этой строки;

*ERROR*2 - мнемоника не распознана;

*ERROR*3 - утверждение плохо сформировано;

*ERROR*4 - символ определен более 1 раза;

*ERROR*5 - строка содержит неверный символ (т.е. ничего не значащий в данном случае);

*ERROR*6 - один из операндов в строке - незаконный;

*ERROR*7 - символ в строке является резервной инструкцией;

*ERROR*8 - ошибочная пара регистров;

*ERROR*9 - слишком много регистров в строке;

*ERROR*10 - выражение, которое должно занимать не более 8 бит, занимает больше;

*ERROR*11 - неверные инструкции JR(IX+N) и JP(IY+N);

*ERROR*12 - ошибка в директиве ассемблера;

*ERROR*13 - незаконная ссылка вперед, т.е. EQU ссылается на символ, который еще не был определен;

*ERROR*14 - деление на ноль;

*ERROR*15 - переполнение в операции умножения;

*eRROR*16 - вложенное макроопределение;

*ERROR*17 - этот идентификатор - не макро;

*ERROR*18 - вложенный макровызов;

*ERROR*19 - вложенный условный оператор.

BAD ORG! - Деректива ORG работает с адресом, который может испортить GENS4, текстовой файл или таблицу символов. Управление возвращается в редактор.

OUT OF TABLE SPACE! - Появляется во время первого прохода, если на таблицу символов выделено недостаточно

памяти. Управление возвращается редактору.

BAD MEMORY! - Нет места для ввода текста, т.е. конец текста близок к вершине ОЗУ памяти. Вы должны спасти текущий текстовой файл или его часть.

П р и л о ж е н и е 2. Резервирование слов, мнемоника и т.д.

Список резервных инструкций внутри GENS4. Эти символы не могут использоваться как метки, хотя они могут быть частью меток. Все резервные инструкции состоят из заглавных букв:

A	B	C	D	E	H	L	I	R	S
AF		AF'		BC		DE		HL	IX
IY		SP		NC		Z		NZ	M
P		PE		PO					

Ниже приводится список мнемоники Z80, директив ассемблера и его команд. Они также должны состоять из заглавных букв.

ADC	ADD	AND	BIT	CALL
CCF	CP	CPD	CPDR	SPI
CPIR	CPL	DAA	DEC	DI
DJNZ	EI	EX	EXX	HALT
IM	IN	INC	IND	INDR
INI	INIR	JP	JR	LD
LDD	LDDR	LDI	LDIR	NEG
NOP	OR	OUTDR	OTIR	OUI
OUTD	OUTI	POP	PUSH	RES
RET	RETI	RETN	PL	RRA
RLC	RLCA	RLD	RR	RLA
RRC	RRCA	RRD	RST	SBC
SCF	SET	SLA	SRA	SRL
DEFB	DEFM	DEFS	DEFW	ELSE
END	ENT	EQU	IF	ORG
MAC	ENDM			
*D	*E	*H	*L	*S
*C	*F	*M		

П р и л о ж е н и е 3 . Рабочий пример

Ниже приводится пример типичного сеанса работы с использованием GENS4. Если Вы новичок по части использования ассемблерных программ или Вы немного не уверены в том, как использовать ассемблер/редактор GENS4, то мы настоятельно советуем Вам тщательно проработать этот пример. Лишний раз заметим, что <ENTER> означает, что Вы должны нажать клавишу "ENTER" на клавиатуре.

Цель примера:

Написать и проверить программу целочисленного умножения,

текст который должен быть сохранен на магнитной ленте, используя команду редактора "T" так, чтобы легко было включать ее в состав будущих программ с магнитной ленты.

Последовательность работы:

1 - напишите программу умножения как подпрограмму и запишите ее на ленту, используя команду редактора "P", чтобы ее можно легко восстановить и отредактировать во время сеанса при наличии ошибок.
2 - Отладьте подрограмму умножения, редактируя ее, если это необходимо.
3 - Запишете отлаженную программу на ленту,используя команду "T" так, чтобы эта подпрограмма могла быть помещена с ленты в другие программы.

Перед стартом мы должны загрузить GENS4 в компьютер. Делайте так:

LOAD " " CODE 26000 <ENTER>,

чтобы загрузить ассемблер с адреса 26000. Затем введите:

RANDOMIZE USR 26000 <ENTER>

Сейчас Вы находитесь в режиме редактирования, т.е. компьютер готов для создания программ на ассемблере.

1-Я стадия: запись программы целочисленного умножения.

Мы используем команду ркдактора I, чтобы ввести текст, используя < -->) (символ табуляции) для получения табулированного листинга.

Ниже мы не указываем, где используется табуляция. Но Вы можете предположить, что табуляция используется между мнемоникой команды и между мнемоникой и операндом. Заметим, что адреса, показанные в ассемблерном листинге примера, который следует, могут не соответствовать полученным в Вашем компьютере, они служат только для иллюстрации:

```
>110,10 <ENTER>
10 ;A FAST INTEGER MULTIPLY <ENTER>
20 ; ROUTINE.MULTIPLIES HL <ENTER>
30 ;BY DE.RETURN THE RESULT <ENTER>
40 ;IN HL.C FLAG SET ON AN <ENTER>
50 ;OVERFLOW.<ENTER>
60 ;<ENTER>
70        ORG #7F00 <ENTER>
80  <ENTER>
90  MULT  OR A <ENTER>
100       SBC HL,DE;HL>DE? <ENTER>
110       ADD HL,DE <ENTER>
120       JR  NC,MU1;YES <ENTER>
130       EX  DE,HL <ENTER>
140 MU1   OR  D <ENTER>
150       SCF;OVERFLOM IT <ENTER>
160       RET NZ;DE>255 <ENTER>
170       OR  E;TIMES 0? <ENTER>
180       LD  E,D <ENTER>
190       JR  NZ,MV4;NO <ENTER>
200       EX  DE,HL;0 <ENTER>
```

```
210         RET <ENTER>
220 <ENTER>
230 ;MAIN ROUTINE <ENTER>
240 <ENTER>
250 MU2     EX  DE,HL <ENTER>
260         ADD HL,DE <ENTER>
270         EX  DE,HL <ENTER>
280 MU3     ADD HL,HL <ENTER>
290         RET C;OVERFLOW <ENTER>
300 MU4     RRA <ENTER>
310         JR  NC,MU3 <ENTER>
320         OR  A <ENTER>
330         JR  NZ,MU2 <ENTER>
340         ADD HL,DE <ENTER>
350         RET <ENTER>
360         <EDIT>
```

Вышеприведенные команды создадут текст программы. Сохраним его на ленте, используя:

```
>P10,350,MULT <ENTER>
```

Помните, что Вы должны включить свой магнитофон в режим записи перед использованием команды "P".

2-Я стадия: отладка программы.

Сначала убедитесь в правильном ассемблировании исходного текста. Мы будем использовать ключ 2, чтобы не производить листинг и обьектный код.

```
>A2 <ENTER>

*HISOFT GENS4 ASSEMBLER*
COPYRIGHT HISOFT 1983,84,87
ALL RIGHT RESERVED

PASS1  ERROR:00
PASS2  ERROR:00

*WARNING* MV4 ABSENT TABLE USED:74 FROM 161
>
```

Мы видим из этой трансляции, что сделали ошибку в строке 190, набрав MV4 вместо MU4, которая является меткой, на которую мы желаем перейти. Редактируем строку 190:

```
>F190,190,MV4,MU4 <ENTER>
 190          JR  NZ, (сейчас используйте подкоманду "S")
```

Теперь ассемблируйте текст вновь. Он должен ассемблироваться без ошибок. Мы должны написать некоторый текст для проверки подпрограммы:

```
>N300,10 <ENTER> (перенумеруем для того, чтобы мы могли на-
                  писать больше текста)
>I10,10 <ENTER>
10 ;SOME CODE TO TEST <ENTER>
20 ;THE MULT ROUTINE <ENTER>
30 <ENTER>
40          LD  HL,50 <ENTER>
```

```
50          LD  DE,20 <ENTER>
60          CALL MULT;MULTIPLY <ENTER>
70          LD  A,H;O/P RESULT <ENTER>
80          CALL AOUT <ENTER>
90          LD  A,L <ENTER>
100         CALL AOUT <ENTER>
110         RET; RETURN TO EDITOR <ENTER>
120 <ENTER>
130 ;ROUTINE TO O/P A IN HEX <ENTER>
140 <ENTER>
150 AOUT    PUSH AF <ENTER>
160         RRCA <ENTER>
170         RRCA <ENTER>
180         RRCA <ENTER>
190         RECA <ENTER>
200         CALL NIBBLE <ENTER>
210         POP  AF  <ENTER>
220 NIBBLE  AND %1111 <ENTER>
230         ADD A,#90 <ENTER>
240         DAA <ENTER>
250         ADC A,#40 <ENTER>
260         DAA <ENTER>
270         LD IY,#5C3A;FOR ROM <ENTER>
280         RST #10;ROM CALL <ENTER>
290         RET <ENTER>
300         <EDIT>
>
```

Теперь ассемблируйте текстовую программу совместно с программой MULT:

```
>a2 <ENTER>

*HISOFT GENS4 ASSEMBLER*
 COPYRIGHT HISOFT 1983,84,87
 ALL RIGHT RESERVED

 7EAC 190 RECA

*ERROR* 02  (нажмите любую клавишу для продолжения)

 PASS 1 ERROR: 01
 TABLE USED: 88 FROM 210
```

В нашей программе есть ошибка: вместо RECA в строке 190 должно быть RRCA. Делаем:

```
>e190
  190 RECA
  190 -->,один пробел,C,R <ENTER>>,<ENTER>
>
```

Теперь проассемблируйте вновь, используя ключи по умолчанию (только a <ENTER>), и текст должен проассемблироваться правильно.

Мы уже в состоянии проверить работу нашей программы MULT, следовательно, нам нужно сообщить редактору, откуда он может выполнить код. Мы делаем это директивой "ENT":

```
>35 ENT S <ENTER>
```

Теперь вновь проассемблируйте текст, и трансляция должна

завершиться правильно с сообщениями:

```
TABLE USED:  88 FROM 211
EXECUTES:  32416
>
```

или подобными этим. Теперь можно запустить наш код на выполнение, используя команду редактора "R". Помножив 50 на 20 мы должны получить 1000, что равно #3e8 в шестнадцатиричном коде:

```
>R <ENTER>
 0032>
```

Не работает! Почему? Распечатайте строки с 380 по 500 (L380,500). Вы увидете,что в строке 430 находится инструкция OR D, за которой следует RET NZ. Все, что она делает - это логическую операцию "или" между регистром D и аккумулятором A и возвращает установленный флаг ошибки <C-флаг>, если результат не равен 0. Целью этого действия является возможность убедиться, что DE<256 и, следовательно, что операция умножения не вызвала переполнения - это делается проверкой того, что в регистре D находится 0. Но "или" будет работать правильно только в том случае, когда в аккумуляторе изначально находится 0, а у нас нет гарантий, что это будет так.

Мы должны убедиться, что A=0 перед выполнением ORD, в противном случае мы получим переполнение со старшими разрядами в качесте результата. Для проверки кода OR a в строке 380 можно изменить на XOR a, которая установит флаги для SBC HL, DE инструкции и установит a=0. Следовательно:

```
>E380 <ENTER>
 380 MULT OR A <ENTER>
 380 --> I (вход в режим вставки) X  <ENTER><ENTER>
>
```

Теперь вновь проассемблируйте и запустите код, используя R. Теперь ответ должен быть правильным: #3E8. Мы можем проверять программу дальше, отредактировав строки 40 и 50, чтобы перемножать различные числа. Ассемблируя и запуская Вы увидете, что программа работает. Теперь, закончив программу, мы можем сохранить ее на магнитной ленте в формате "INCLUDE":

```
>T300,999,MULT <ENTER>
```

Не забудьте включить магнитофон в режим записи перед нажатием <ENTER>. Если Вы хотите сохранить программу на дискете, то не нужно использовать команду "T". Программа, сохраненная обычной командой "P", может быть запущена с дисковода.

Однажды записанная подобным образом подпрограмма может быть вставлена в программу как показана ниже:

```
500  RET
510
520 ;INCLUDE TNE MULT ROUTINE HERE (включаем здесь MULT)
530
540 *F MULT
550
560 ;THE NEXT ROUTINE (следующая подпрограмма)
```

Когда вышеприведенный текст будет транслироваться, ассемблер спросит Вас "START TAPE..." Когда он доберется до строки 540 во время как первого, так и второго проходов. Следовательно, Вы должны направить MULT на вывод с ленты в обоих случаях. Для этого обычно нужно будет перемотать ленту после первого прохода. Вы можете записать два дубля программы MULT на ленту один за другим, и один использовать во время первого прохода, а второй - во время второго.

Когда происходит ввод с дисковода, никаких сообщений не появляется, все работает автоматически.

Пожалуйста, изучите тщательно вышеприведенный пример и постарайтесь выполнить его самостоятельно.

16. MONS4

Раздел 1. З а п у с к

MONS4 поставляется в перемещенной форме. Вы просто загружаете его с желаемого адреса, и затем вызываете с этого же адреса. Если Вы хотите запустить MONS4 вновь (вернувшись из MONS4 в BASIC), то Вы должны ввести адрес, с которого первоначально загрузили отладчик.

Пользователи "PLUS 3" должны прочитать дополнительный листок с особенностями внешнего отладчика, который расположен на адресуемом диске и требует только 100 байтов в обычном 48К-байтовом ОЗУ.

Пример:

Скажем, Вы хотите загрузить MONS4 по адресу #C000 (49152-десятичный). Для этого выполните следующие операции:

```
LOAD "" CODE 49152 <ENTER>
RANDOMIZE USR 49152 <ENTER>
```

для повторного входа используйте:

```
RANDOMIZE USR 49152 <ENTER>
```

Будучи однажды помещенным в память, MONS4 занимает приблизительно 6К в длинну, но Вы должны предоставить непрерывную одласть 7К на загрузку MONS4, т.к. после основных кодов надо поместить таблицу перемещаемых адресов. MONS4 содержит собственный внешний стек, так что он является самообслуживающейся программой.

MONS4 по умолчанию загружается с адреса 55000, хотя Вы можете загрузить его с любого разумного адреса. Обычно удобно загрузить MONS4 в верхние адреса памяти.

Раздел 2. Получение копии

Загрузив MONS4 в память Вашего SPEKTRUMa, Вы можете сделать копию, проделав следующие операции:

```
SAVE "MONS4" CODE XXXXX,6656 <ENTER>          - на кассету
SAVE *"M",1,"MONS4" CODE XXXXX,6656 <ENTER>   - на дискету
SAVE *"M",1,"MONS4" CODE XXXXX,6780 <ENTER>   - на диск OPUS,
```

где XXXXX - адрес, с которого Вы загрузили MONS4.

Когда Вы войдете в MONS4, Вам представится изображение так называемой фронтальной панели (в дальнейшем - Ф.П. (см. примечание с примером)). Оно состоит из регистра Z Z80 и флага ов вместе с их содержимым плюс 24-х байтовая секция памяти, сосредоточенная вокруг указателя памяти, первоначально указывающего на нулевой адрес. В верхней части экрана находится дизассемблированная инструкция, на которую указывает указатель памяти (в дальнейшем - MP).

На входе MONS4 все адреса, отображаемые внутри Ф.П , представлены в шестнадцатиричном формате Вы можете изменить это с помощью команды:

<SIMBOL SHIFT> 3 ,

и адреса будут представлены в десятичном формате (см. следующий раздел). Заметим, однако, что адреса всегда должны вводится в шеснадцатиричном формате. Команды вводятся с клавиатуры в соответствии с подсказкой ">" как с верхнего, так и с нижнего регистров.

Некоторые команды, эффект от которых может быть гибельным в случае их ошибочного использования, требуют нажатия клавиш (SIMBOL SHIFT). Это ручное нажатие клавиши (SIMBOL SHIFT) будем представлять знаком "^", т.е. запись "^Z" означает нажатие (SIMBOL SHIFT) и клавиши "Z" вместе

Команды выполняются немедленно и не требуют после себя ввода <ENTER>. Неверные команды просто игнорируются.

Многие команды требуют ввода 16-ричного числа. При вводе такого числа Вы можете ввести любое количество 16-ричных цифр (0...9 И A...F) и закончить ввод любым отличным от 16-ричного числа символом. Если ограничителем является значащая команда, то эта команда выполняется после того, как будет выполнена предыдущая, если ограничителем является знак "-", то введенное отрицательное число будет возвращено в двойном дополнительном коде: "1800-" дает "E800". Если при вводе 16-тиричного числа Вы ввели более 4-х цифр, то только 4 последние введенные цифры сохраняются и отбражаются на экране.

Для возврата из MONS4 в BASIC просто нажмите (STOP) (т.е. "^A").

Ползователи "SPECTRUM PLUS 3" должны ознакомиться с дополнительными инструкциями, поставляемыми с этим руководством, прежде чем идти дальше.

Раздел 3. Команды MONS4

В MONS4 используются нижеприведенные команды. В этом разделе <ENTER> употребляется в качестве ограничителя 16-ичного числа. Фактически это может быть любой не 16-ичный символ (см. раздел 1). Для обозначения пробела используется символ "-".

1. (SIMBOL SHIFT) 3 или

Переключает систему счисления, в которой отображаются адреса, с 16-ичной на 10-ичную и обратно. На входе MONS4 все адреса показываются в 16-ичном виде, для 10-ичного отображения используйте "^3" и вновь "^3" - для возврата в 16-ичную форму. Это влияет на все адреса, отображаемые MONS4, включая адреса, получаемые дизассемблером, но это не меняет отображения содержимого памяти, которое всегда выдается в 16-ичном виде.

2. (SIMBOL SHIFT) 4 или $

Отображает дизассемблированную страницу, начиная с адреса, указанного MP. полезно посмотреть и следующие инструкции. Нажмите "^4" вновь или <EDIT> для возврата к отображению Ф.П. Или другую клавишу для получения следующей дизассемблированной страницы

3. <ENTER>

Наращивает счетчик адресов на 1, так что 24-х байтовая область памяти сдвигается на 1 байт вперед.

4. <Стрелка вверх>

Уменьшает MP (указатель памяти) на 1.

5. <Стрелка влево>

Уменьшает значение MP на 8. Используется для быстрого просмотра.

6. <Стрелка вправо>

Увеличивает значение MP на 8. Используется для быстрого просмотра.

7. <,> (Запятая)

Заменяет MP адресом, находящимся в стеке (SP). Это полезно, когда Вы хотите посмотреть адрес возврата вызванной подпрограммы.

8. <G>

Поиск по образцу введенной строки. Сначала высвечивается подсказка на ввод первого байта, который необходимо найти. За байтом следует <ENTER>. Затем вводите следующий байт (и <ENTER>) в ответ на ":" до тех пор, пока Вы не определите всю строку образа. После этого введите только <ENTER>, это ограничит строку образа, будет произведен поиск образа, начиная с текущего адреса до первого появления введенной строки. Когда эта строка будет найдена, обновится Ф.П. Так, что Ф.П. будет указывать на первый символ строки.

Например, скажем, Вы хотите исследовать память, начиная с адреса #80000 на появление образа #3E #FF (2 байта). Последовательность действий должна быть следующей:

```
M:80000 <ENTER>    - устанавливает MP
G:3E    <ENTER>    - определяет 1-й байт строки
FF      <ENTER>    - определяет 2-й байт строки
<ENTER>            - ограничивает строку.
```

После последнего <ENTER> (или любого не 16-ичного символа) G начинает поиск с адреса #80000 первого появления #3E #FF. Когда будет найдена строка, то обновится изображение на Ф.П. Для поиска дальнейших появлений строки используйте команду N.

9. <H>

Вам выводится подсказка ":" на ввод 10-ичного числа, ограниченного не цифрой (любым символом, кроме 0...9). Как только Вы введете число и ограничитель, в той же строке высветится знак "=" и 16-ичный эквивалент введенного 10-ичного числа. Теперь нажмите любую клавишу для возврата в командный режим.

Например:

H:441472_=A200 - пробел был использован как ограничитель.

10. <I>

используется для копирования блока памяти из одного места в другое, причем блок памяти может быть скопирован в ячейки с перекрытием своего первоначального местоположения. <I> выдает подсказки на ввод стартового и конечного адресов блока,который должен быть скопирован (<FIRST:>,<LAST:>) и затем для адресов, по которым он должен быть помещен (<to:>). Введите в ответ на каждую подсказку 16-ичные числа. Если стартовый адрес больше конечного, то команда прерывается, в противном случае блок перемещается как указано.

11. <J>

Выполняет задачу с определенного адреса. Эта команда выдает подсказку ":" для 16-ичного числа, после введения которого сбрасывается внешний стек, экран ощищается и выполнение перемещается в область определенного адреса. Если Вы желаете вернуться к Ф.П. После выполнения, то установите точку останова (см. команду W) в месте, где Вы желаете вернуться к отображению.

Например:

J:B0000 <ENTER> - выполняет программу с адреса #B0000.

Вы можете прервать программу перед концом ввода адреса, используя <EDIT>. Отметим, что выполнение команды <J> портит регистры Z80, поэтому программа в мащинных кодах не должна делать присвоений значений, содержащихся в регистрах. Если Вы желаете выполнять программу с регистрами, то нужно использовать "^K".

12. <SIMBOL SHIFT> K

Продолжает выполнение с адреса, находящегося в счетчике инструкций. Эта команда, вероятнее всего, будет использоваться совместно с командой W - пример пояснит это.

Скажем, Вы в пошаговом режиме (используя "^Z") проходите через коды, приведенные ниже, и Вы дошли до адреса #90000, но Вы хотите увидеть, как выставляются флаги после вызова подпрограммы в #8800.

```
891E   3EFF         LD     A,-1
8920   CD0090       CALL   39000
8923   2A0080       LD     HL,(#8000)
8926   7E           LD     A,(HL)
8927   111488       LD     DE,#8814
892A   CD0088       CALL   #8800
892D   2003         JR     NZ<LABEL
892F   320280       LD     (#8002),A
8932   211488 LABEL LD     HL,#8814
```

Действуйте в следующем порядке:
установите точку останова, используя W, по адресу #892D (напомним, что сначала нужно использовать "^M" для установки MP) и затем введите команду "^K". Выполнение будет продолжаться с адреса, который находится в счетчике адресов (PC). В нашем случае это #8920. Выполнение будет продолжаться до тех пор, пока не дойдем до адреса,указанного в точке останова,затем будет обновлен экран и Вы сможете проверить состояние флагов и т.п. После вызова процедуры с адреса #88000. Затем можно возобновить пошаговый режим. "^K" полезна для выполнения кода без первоначального сброса стека или без разрушения содержимого регистров, что делает <J>.

13. <L>

Распечатывает блок памяти, начиная с адреса, находящегося в данный момент в MP.

<L> очищает экран и отображает 16-ичное представление и

ASCII эквиваленты 80 байтов памяти, начиная с текущего адреса, на который указывает MP. Адреса будут указами в 16-ичном или 10-ичном виде в зависимости от текучего состояния Ф.П. (см. "^З"). Отображение состоит из 20 рядов по 4 байта в каждом, ASCII эквиваленты показаны в конце каждого ряда, применительно к ASCII от отображения любого значения выше 127 отнимается код 128 и любые значения между 0...31 Включительно показаны как ".".

В конце страницы листинга Вы можете вернуться к основному отображению Ф.П. Нажатием (EDIT) или продолжать со следующей страницы из 80 байтов нажатием любой другой клавиши.

14. <M>

Заносит в MP определенный адрес. Выдается подсказка ":" на ввод 16- ичного адреса (см. раздел 1). MP обновляется введенным адресом и соответственно на экране меняется отображение области памяти.

15. <N>

Находит следующее появление строки, определенной в прошлый раз командой <G>. <G> позволяет Вам определить строку и затем осуществить поиск ее первого появления, если Вы хотите найти следующие появления строки, используйте <N>.<N> начинает поиск с адреса,указанного в MP и обновляет отображение памяти, когда искомая строка вновь найдена.

16. <O>

Команда берет байт, адресуемый в данный момент MP, и представляет его как относительное смещение, соответственно обновляя отображение.

Например,
допустим, MP установлен в #6800 и содержимое ячеек #67FF и #6800 равны #20 и #16 соответственно (это может быть представлено как JR NS,$+24). Чтобы выяснить, куда будет переход по ненулевому условию, просто нажмите <O>, тогда MP адресуется на назначенный байт #16, экран будет обновлен и отобразится область памяти с #6817, требуемого значения перехода.

Помните, что относительные перемещения выше #7FF (127) трактуются процессором как отрицательные, <O> принимает это в расчет.
См. Также команду <U> в связи с <O>.

17. <P>

Заполняет память между определенными границами нужным байтом. <P> выдает подсказки (FIRST:),(LAST:),(WITH:). Введите 16-ичные числа в соответствии с этими подсказками

(соответственно: начальный и конечный адреса (включительно) блока, который Вы желаете заполнить, и байт - заполнитель). Например:

```
<P>
FIRST: 7000 <ENTER>
LAST:  77FF <ENTER>
WITH:  55   <ENTER>
```

Ячейки с #7000 по #77FF будут зоднены байтом #55. Если стартовый адрес больше конечного, то <P> будет оборвана.

18. <Q>

Переключение наборов регистров. На входе Ф.П. Отображается стандартный набор регистров (AF,HL,DE,BC). При использовании <Q> будет отображаться альтернативный набор регистров (AF',HL',,DE',BC'), который отличается от стандартного штрихом "'" после имени регистра. Если <Q> используется, когда отображается альтернативный набор регистров, то на экране будет выдан стандартный набор.

19. <SIMBOL SHIFT) T

Устанавливает точку останова после текущей инструкции и продолжвет выполнение.
Например:

```
9000    B7      OR      A
9001    C20098  CALL    NZ,#9800
9004    010000  LD      BC,0
9800    21FFFF  LD      HL,-1
```

Вы в пошаговом режиме продвигаетесь по кодам и достигли адреса #9001 с ненулевым значением в регитре A, так что флаг "ZERO" будет в состоянии NZ после инструкции OR A.
Если Вы сейчас используете "^Z" для продолжения пошагового режима, то выполнение продолжится с адреса #9800 подпрограммы. Если Вы не желаете шагать по подпрограмме, то используйте команду "^T", тогда инструкции с #9800 и вызов будут пройдены автоматически и выполнение остановится на адресе #9004 для дальнейшего продолжения в пошаговом режиме. Помните, что "^T" устанавливает точку останова после текущей инструкции и затем использует команду "^K" (см. команду "^Z" с расширенным примером пошагового прохода).

20. <T>

Дизассемблирование. Дизассемблирует блок кодов на пример и/или дискету (по ключу). Сначала выдается подсказка на ввод начального (<FIRST:>) и конечного (<LAST:>) адресов кодов, которые Вы желаете дизассемблировать. Введите их в 16-ичном виде (как описано в разделе 1).
Если начальный адрес больше конечного, то команда аннулируется. После ввода адресов выдается подсказка <PRINTER?>. Для направления результатов дизассемблирования

на принтер отвечайте "Y". При наборе любого другого символа результат будет выведен на терминал.

Затем выдается подсказка <TEXT:> на ввод 16-ичного начального адреса текстового файла, который будет являться результатом дизассемблирования. Если Вы не хотите, чтобы был сгенерирован текстовой файл, то просто нажмите <ENTER> после этой подсказки. Если Вы определили адрес, то дизассемблированный файл будет получен, начиная с этого адреса, в форме, пригодной для использования GENS4. Если Вы хотите загрузить этот файл с помощью GENS4, то Вы должны заметить его конечный и начальный адреса, вернуться в BASIC и сохранить текст как файл типа "CODE". Затем Вы можете загрузить код в редактор GENS4 прямо с помощью команды <G>.

Вместо ввода адреса в ответ на <TEXT:> Вы можете ввести имя файла на дискете и на ней текс будет сохранен, как только пройдет дизассемблирование.

Например:

```
TEXT: 2: DSODE <ENTER>
```

Результат дизассемблирования будет сохранен на 2-м дисководе под именем DCODE. Если Вы дизассемблируете на дисковод, то листинг на экран выдаваться не будет. Файл, полученный на дисководе, является самозагружающимся с помощью команды <G>. Если на некоторой стадии генерации текстового файла текст будет перекрывать MONS4, то дизассемблирование будет прекращено - нажмите любую клавишу для возврата к Ф.П. Если Вы определили адрес текстоаого файла, то Вас попросят определить "WORKSPACE:" - начальный адрес свободной области памяти, которая используется как примитивная таблица символов для любых меток, которые производит дизассемблер. Требуется по два байта на каждую метку. По умолчанию (когда Вы просто нажимаете <ENTER>) резервируется 4K пространства ниже MONS4.

После этого повторно появляются запросы на начальный и конечный адреса любых областей данных, существующих внутри блока, который Вы желаете дизассемблировать. Области данных - это области, скажем, текста, который Вы не желаете представлять как инструкции Z80 - вместо этих областей дизассемблером должны быть сгенерированы дериктивы DEFB.

Если значение байта данных лежит между 32 и 127 (#20 и #7F) включительно, то дается интерпритация байта в коде ASCII, т.е. #41 заменяется на a после DEFB. Когда Вы закончили определение областей данных, или Вы не желаете их определять, то просто нажмите <ENTER> в ответ на обе подсказки. Команда <T> определяет область в конце MONS4 для хранения адресов областей данных и, таким образом, Вы можете определять столько областей данных, сколько позволяет память. Каждая область данных требует 4 байта пространства. Отметим, что использование <T> уничтожает любые точки останова, которые первоначально были установлены (см. команду W).

Байт после выполнения инструкции RST дизассемблируется как DEF N, т.к. этот байт собирается ПЗУ SPECTRUMa и никогда не выполняется.

Экран будет очищен. Если будет запрос на создание текстового файла, то произойдет короткая задержка (в зависимости от размеров секции памяти, которую Вы хотите дизассемблировать) пока создается таблица символов во время первого просмотра, когда это будет сделано, на экране (или принтере) появится листинг дизассемблирования, если только Вы не дизассемблируете на дискету - в этом случае листинг не

производится. Вы можете остановить листинг в конце строки нажатием <ENTER> или <SPACE>, последующее нажатие <EDIT> возвращает к отображению Ф.П., Любой другой ключ продолжает дизассемблирование.

Если встречается неверный код,то он дизассемблируется как <NOP> и отмечается звездочкой после кода в листинге.В конце дизассемблирования изображение будет приостановлено и, если был запрос на создание текстового файла, возникнет сообщение:

END OF THE TEXT XXXXX

Метки генерируются там, где это уместно (т.е. в C30078) в форме xxxxx, где XXXXX - абсолютный 16 - ичный адрес метки, но только если их адреса попадают внутрь границ дизассемблирования. В противном случае метки не генерируются, а просто даются 16-ичные или 10-ичные адреса

Например, если Вы проассеблировали область между #7000 и #8000, то затем инструкция C30078 может быть дизассемблирована как JP L7800; с другой стороны, если Вы дизассемблируете с #9000 по #9800, то эта же инструкция может быть дизассемблирована как JP #7800 или #JL 30720, если используется 10-ичное отображение.

Если особый адрес отнесен к инструкции внутри дизассемблирования, то его метка появится в поле метки (перед мнемоникой) дизассемблированной конструкции этого адреса, но только если листинг направляется в файл.

Например:

```
T
FIRST: 8B     <ENTER>
LAST:  9E     <ENTER>
PRINTER: Y    <ENTER>
TEXT:         <ENTER>
FIRST: 95     <EHTEP>
LAST:  9E     <ENTER>
FIRST:        <ENTER>
LAST:         <ENTER>
```

Может быть проиведено нечто подобное:

```
008B   FE16      CP      #16
008D   3801      JR      C,L0090
008F   23        INC     HL
0090   37        SCF
0091   225D5C    LD      (5C5D),HL
0094   C9        RET
0095   BF524E    DEFB    #BF,"R","N"
0098   C4494E    DEFB    #C4,"I","N"
009B   4B4559    DEFB    "K","E","Y"
009E   A4        DEFB    #A4
```

21. <U>

Используется совместно с командой <O>. Помните, что она обновляет отображение памяти в соответствии с относительным смещением, т.е. показывает эффект инструкции JR или DJNZ. <U> используется для возврата к отображению, которое было при последнем использовании <O>.

Например:

```
  7200    U7          71F3    77
  7201    20          71F4    C9
>7202     F2<        >71F5    F5<
  7203    06          71F6    C5
  картинка 1          картинка 2
```

На Вашем терминале - картинка 1 и Вы хотите узнать, когда будет относительный переход 20 F2. Нажмите клавишу <O> и появится картинка 2 - обновленное отображение памяти.

Итак, Вы нажимаете <o>, и появляется картинка 2. Теперь Вы исследуете код по адресу #71F5 и затем хотите вернуться к кодам, соответствующим первоначальному относительному смещению для того, чтобы увидеть, что происходит, если установлен флаг нуля. Итак, нажмите <U>, и отображение памяти вернется к картинке 1. Отметим, что Вы можете использовать <U> только для возврата к последнему проявлению команды <O>, все предыдущие проявления <O> утеряны.

22. <V>

Используется совместно с командой <X> и подобна команде <U> по эффекту, с той разницей, что она обновляет отображение памяти на картинку, которая была перед последней командой <X>.

Например:

```
  8702    AF          842D    18
  8703    CD          842E    A2
>8704     2F<        >842F    E5<
  8705    44          8430    21
  картинка 1          картинка 2
```

На Вашем экране - картинка 1 и Вы хотите посмотреть подпрограмму с адреса #842F. Для этого Вы нажимаете <X>. При данной картинке отображение памяти обновляется на картинку 2.

Вы просматриваете эту подпрограмму, и затем хотите вернуться к кодам первоначального вызова подпрограммы. Для этого нажмите <V> и вновь появится картинка 1.

Как и в случае с <U>, используя эту команду, Вы можете достичь только адресов,при которых была введена последняя команда <X>.Все предыдущие адреса, в которых использовалась команда <X>, будут утеряны.

23. <W>

Установка точки останова. Точка останова для MONS4 - это просто инструкция вызова внутри MONS4 подпрограммы, которая отображает Ф.П. Так, что программисту дается возможность остановить выполнение программы и проверить регистры Z80, флаги и любые значащие ячейки памяти.

Таким образом, если Вы хотите остановить выполнение программы в #9876, допустим, то используйте команду <M> для установки MP на #9876 и затем используйте <W>для установки точки останова по этому адресу, 3 байта кода, которые были первоначально по адресу #9876, заменяются инструкцией CALL, которая останавливает выполнение, когда на нее выходят. Когда достигается эта инструкция CALL, то она

восстанавливает 3 первоначальные байта по адресу #9876 и высвечивает Ф.П. Со всеми регистрами и флагами в состоянии,в которой они находились перед выполнением точки останова. Теперь Вы можете использовать любые возможности MONS4 обычным способом.

Замечания по использованию точек останова:

Когда встречается точка останова, MONS4 будет выдавать звуковой сигнал и ожидать Вашего нажатия клавиши перед возвратом к Ф.П. MONS4 использует область в конце самого себя, которая первоначально содержит перемещаемые адреса для сохранения информации о точках останова. Это означает, что Вы можете установить столько точек останова, сколько позволит память. Каждая точка останова требует 5 байтов памяти. Когда останов произошел, MONS4 автоматически восстанавливает значения памяти, которые существовали до этой точки останова.

Отметим, что команда <T> использует эту область, поэтому при работе команды <T> все точки останова потеряются. Точки останова могут быть установлены только в ОЗУ. Т.к. точка останова состоит из трехбайтовой инструкции, то в определенных случаях необходимо проявить предусмотрительность,

например: предполагаемый код:

8000	3E	8008	00
8001	01	8009	00
8002	18	800A	06
8003	06	800B	02
>8004	AF<	>800C	18
8005	0E	800D	F7
8006	FF	800E	06
8007	01	800F	44

Если Вы установили точку останова по #8004 и затем начали выполнение с ячейки #8000, то регистр A будет загружен значением 1, выполнение переместится на #800A, регистр B загрузится значением 2 и выполнение переместится на #8005. Но в #8005 помещается младший байт вызова точки останова и, таким образом, мы испортим код и появятся непредсказуемые результаты. Эта ситуация нетипична, но Вы должны оградить себя от нее - в этом случае единственный выход - использовать пошаговый проход (см. команду "^Z" ниже с подробным описанием пошагового прохода).

24. <X>

Используется для обновления MP по назначению абсолютного вызова подпрограммы или по инструкции JP. <X> берет 16-разрядный адрес, определяемый байтом, на который указывает MP, и байтом MR+1, и затем обновляет отображение памяти на экране таким образом, что отображаются адреса,близлежащие к данному адресу. Помните, что младшая половина адреса определяется первым байтом, а старшая - вторым (формат INTEL).

Например, Вы хотите посмотреть подпрограмму, которая

вызывается кодом CD0563. Установите MP (используя <M>) так, что он адресует к коду 05 внутри инструкции вызова и нажмите <X>. Отображение памяти будет обновлено и отобразятся ячейки с адресами, близкими к #6305 (см. также команду <V>).

25. <Y>

Ввод ASCII. <Y> дает Вам новую строку, которую Вы можете ввести символами ASCII прямо с клавиатуры. Эти символы попадают на эхо-печать и их 16-тиричные эквиваленты вводятся в память, начиная с текущего значения MP. Строка символов должна заканчиваться <EDIT>. <DELETE> (<CAPS SHIFT> 0) может быть использована для удаления символов из строки.

Когда Вы закончите ввод символов ASCII (и введете <EDIT>), то экран обновится и MP спозиционируется на конец строки, введенной в память.

26. <SIMBOL SHIFT> Z

Перед использованием "^Z" (или "^T") PC должен быть установлен на адрес инструкции, которую Вы хотите выполнить. "^Z" просто выполняет текущую инструкцию и обновляет Ф.П., чтобы отобразить изменения, вызванные выполнением инструкции.

Отметим, что пошаговый режим возможен в любом месте карты памяти (и в ОЗУ, и в ПЗУ) но не возможен во внешнем ПЗУ.

Далее следует расширенный пример, который поясним действие многих команд отладчика MONS4. Вы должны их внимательно изучить и постараться выполнить самостоятельно.

Р а б о ч и й п р и м е р:

Давайт предположим, что у нас в машине есть 3 секции кода, приведенного ниже, первая секция - главная программа, которая загружает HL и DE числами и затем вызывает подпрограмму для их перемножения (вторая секция) с результатом в HL и затем дважды вызывает подпрограмму для вывода результата умножения на экран (3-я секция). Текст секций:

```
7080    2A0072          LD      HL,(#7200)
7083    ED5B0272        LD      DE,(#7202)
7087    CD0071          CALL    MULT
708A    7C              LD      A,H
708B    CD1D71          CALL    AOUT
708E    7D              LD      A,L
708F    CD1D71          CALL    AOUT
7092    210000          LD      HL,0

7100    AF      MULT    XOR     A
7101    ED52            SWC     HL,DE
7103    19              ADD     HL,DE
7104    3001            JR      NC,MU1
7106    EB              EX      DE,HL
```

```
7107    B2      MU1     OR      D
7108    37              SCF
7109    C0              RET     NZ

710A    B3              OR      E
710B    5A              LD      E,D
710C    2007            JR      NZ,MU4
710E    EB              EX      DE,HL
710F    C9              RET

7110    EB      MU2     EX      DE,HL
7111    19              ADD     HL,DE
7112    EB              EX      DE,HL

7113    29      MU3     ADD     HL,HL
7114    D8              RET     C

7115    1F      MU4     RRA
7116    30FB            JR      NC,MU3
7118    B7              OR      A
7119    20F5            JR      NZ,MU2
711B    19              ADD     HL,DE
711C    C9              RET

711D    F5      AOUT    PUSH    AF
711F    0F              RRCA
711F    0F              RRCA
7120    0F              RRCA
7121    0F              RRCA
7122    CD671           CALL    NIBBLE
7125    F1              POP     AF

7126    E60F    NIBBLE  AND     %1111
7128    C690            ADD     A,#90
712A    27              DAA
712B    CE40            ADC     A,#40
712D    27              DAA
712E    FD213A5C        LD      IY,#5C3A
7132    D7              RST     #10
7133    C9              RET
7200    1B2A            DEFW    10779
7202    033A            DEFW    3
```

Теперь мы хотим исследовать вышеприведенный код. Сделаем это со следующим набором команд. Отметим, что пошаговый режим - это просто один из способов, он не достаточно эффективен, но весьма иллюстративен:

```
M: 7080 <ENTER>         установить MP на #7080
7080                    установить PC на #7080
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
```

```
                        следует вызов CALL
M:7115 <ENTER>          пропуск предподготовки чисел
W                       установка точки останова
^K                      продолжить с #7100 до останова
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
^Z                      возврат из подпрограммы умножения
^Z                      простой шаг
^Z                      следует вызов CALL

M:7128 <ENTER>          установка MP на нужный байт
W                       установка точки останова
^K                      продолжить с #711D до останова
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
^Z                      простой шаг
                        просмотр адреса возврата
W                       установка там точки останова
^K                      продолжение
^Z
W
^K
^Z
^T
```

Пожайлуйста, проработайте этот пример, сначала введите код подпрограммы (см. модификация памяти) - можно использовать GENS4, и затем выполните команды, приведенные выше. Вы высоко оцените пример в качестве иллюстрации того, как пройти по программе.

27. <SIMBOL SHIFT> P

Эта команда аналогична LIST, но вывод вместо экрана идет на принтер. Помните, что в конце страницы Вы нажимаете <EDIT> для возврата к Ф.П. Или любую другую клавишу для получения другой страницы.

28. Модификация памяти

Значение ячеек по адресам, которые дает MP, могут быть модифицированы вводом 16-ичного числа, за которым следует ограничитель (см. раздел 1). Две последние 16-ичные цифры (если одна, то она дополняется нулем) вводятся в ячейку, на которую указывает MP, и затем команда (если она есть), определенная ограничителем, выполняется. Если ограничивается незначащая команда, то она игнорируется.
Например:

```
F2 <ENTER>              #F2 вводится и MP:=MP+1
123 <CAPS SHIFT> 8      #23 вводится и MP:=MP+8
EM:E00_                 #E0 вводится в текущую ячейку
                         и затем MP:=#E0. Пробел огра-
                         ничивает команду "M"
```

8CO	#8C вводится и MP обновляется (т.к. использована команда "O" для назначения относительного смещения #8C)
2A5D_	#5D вводится и MP не меняется, т.к. ограничитель - пробел, а не команда.

Если 16-ичное число вводится в ответ на подсказку и ограничивается точкой, то введенное число будет занесено в текущий регистр Z80, адресуемый стрелкой.

На входе MONS4 стрелка ">" указывает на PC, и использование ".", Как ограничителя 16-ичного числа, будет модифицировать PC. Если Вы хотите изменить другие регистры, то используйте точку саму, а не как ограничитель числа. указатель ">" будет циклически указывать регистры от PC до AF. Отметим, что невозможно адресовать и изменять SP или IY регистры.

Например, предположим, что ">" первоначально указывает на PC:

.	Указывает на IY
	Указывает на IX
0.	Устанавливает IX в 0
	Указывает на HL
123.	HL:=#123
.	Указывает на DE
	Указывает на BC
E2A7	BC:=#E2A7
.	Указывает на AF
FF00	AF:=#FF и сбрасывает все флаги
.	Указывает на PC
8000.	PC:=#8000

Отметим, что точка также может исползоваться для модификации адреса альтернативного набора регистров, если он отобрахается. Используйте команду "Q" для переключения набора регистров.

Пример отображения фронтальной панели

```
710C      2007     JR    NZ,#7115

>PC   710C   20 07 EB C9 EB 19 EB
 SP   D0AF   8A 70 06 03 0A 03 0D
 IY   CF6A   0D 11 0C 0F 09 18 18
 IX   D09F   04 03 04 00 00 00 1B
 HL   2A1B   DF FE 29 28 02 CF 02
 DE   0000   F3 AF 11 FF FF C3 CB
 BC   0004   FF C3 CB 11 2A 5D 5C
 AF   0304   >
 IR   3F7C   ON

 7100   AF     7108  37     7110  EB
 7101   ED     7109  C0     7111  19
 7102   52     710A  B3     7112  EB
 7103   19     710B  5A     7113  29
 7104   30    >710C  20<    7114  D8
 7105   01     710D  07     7115  1F
 7106   EB     710F  C9     7117  FB
 7107   B2     710F  C9     7117  FB
>
```

Выше показано типичное изображение Ф.П. (одно из изображений, полученных при пошаговом проходе по п/п MULT в примере использования команды Z.

Первые 9 строк изображения содержат 9 регистров, сначала - имя регистра <от PC до IR>, затем <для PC...BC> - значения содержимого регистров, и затем - содержимое 7 ячеек памяти,начиная с адреса, находящегося в регистре.

Регистр флагов декодируется, чтобы показать текущие значения битов флагов, установленных так, как они используются внутри регистра. Если регистр флагов установлен в #FF, то за отображением имени AF будет следовать:

```
00FF    C2   H   UNS ,
```

т.е. знак, ноль полуноситель, паритет/переполнение, сложить/вычесть и будут установлены все флаги носителей. Справа от регистров IR находятся слова ON или OFF, которые показывают текущее состояние внешних прерываний. Указатель регистров ">" указывает на регистр, который адресуется в данный момент (см. раздел 2).

24 Байта памяти отображаются ниже отображения регистров. Отображение состоит из адреса (2 байта,4 символа),за которым следует значение (1 байт, 2 символа) памяти по этому адресу. Отображение размещено вокруг текущего значения MP, указываемого ">".

Команды вводятся вверху экрана в ответ на подсказку ">". Отображение обновляется после каждой отработанной команды.

Орел, Тип «Труд». Заказ №783 Тир. 20000 экз.