

РАДИОКОНСТРУКТОР
**"Бытовое электронно-вычислительное
устройство ЛИК"**

Комплект программного обеспечения

Формуляр

Описание применения

Руководство программиста

Руководство оператора

РАДИОКОНСТРУКТОР

"БЫТОВОЕ ЭЛЕКТРОННО-ВЫЧИСЛИТЕЛЬНОЕ УСТРОЙСТВО ЛИК"

КОМПЛЕКТ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Формуляр

Описание применения

Руководство программиста

Руководство оператора

59 страниц

СОДЕРЖАНИЕ

Формуляр

1. ОБЩИЕ СВЕДЕНИЯ	6
2. ОБЩИЕ УКАЗАНИЯ И ОСНОВНЫЕ ХАРАКТЕРИСТИКИ	6
3. КОМПЛЕКТНОСТЬ	6
4. СВИДЕТЕЛЬСТВО О ПРИЕМКЕ	7
5. ГАРАНТИЙНЫЕ ОБЯЗАТЕЛЬСТВА	7
6. СВЕДЕНИЯ О РЕКЛАМАЦИЯХ	7

Описание применения

1. НАЗНАЧЕНИЕ ПРОГРАММЫ	9
2. УСЛОВИЯ ПРИМЕНЕНИЯ	9
3. ОПИСАНИЕ ПРОГРАММЫ	10

Руководство программиста

1. СТРУКТУРНОЕ ОПИСАНИЕ РАДИОКОНСТРУКТОРА ЛИК	12
1.1. ТЕРМИНОЛОГИЯ	12
1.2. РАБОЧИЕ РЕГИСТРЫ	13
1.3. ПАМЯТЬ	13
1.4. РС ПРОГРАММНЫЙ СЧЕТЧИК	14
1.5. SP УКАЗАТЕЛЬ СТЕКА	14
1.6. ВВОД/ВЫВОД	14
1.7. ПРЕДСТАВЛЕНИЕ ПРОГРАММЫ В ПАМЯТИ	15
1.8. АДРЕСАЦИЯ ПАМЯТИ	16
1.9. БАЙТ СОСТОЯНИЯ	16
2. ОПИСАНИЕ АССЕМБЛЕРА	17
2.1. ОСНОВНЫЕ ЭЛЕМЕНТЫ ЯЗЫКА	17
2.1.1. Символы	17
2.1.2. Числа, константы и символические имена	17
2.2. СИНТАКСИС АССЕМБЛЕРНЫХ ПРЕДЛОЖЕНИЙ	18
2.2.1. Псевдокоманды и директивы	18
2.2.1.1. ORG установить начальное значение счетчика адреса	18
2.2.1.2. DS ЗАРЕЗЕРВИРОВАТЬ ОБЛАСТЬ ПАМЯТИ	18
2.2.1.3. EQU задать значение символического имени	18
2.2.1.4. DB задать байт данных	19
2.2.1.5. DW задать слово данных	19
2.2.1.6. END конец исходной записи	19
2.3. СИСТЕМА КОМАНД	21
2.4. ТРАНСЛЯЦИЯ И ЗАПУСК ПРОГРАММЫ	25
3. ИНТЕРПРЕТАТОР ЯЗЫКА БЕЙСИК	26
3.1. ПРОГРАММА НА ЯЗЫКЕ БЕЙСИК	26
3.2. ОПЕРАНДЫ, ПЕРЕМЕННЫЕ И ВЫРАЖЕНИЯ	26
3.3. ДИРЕКТИВЫ ЯЗЫКА БЕЙСИК	29
3.4. ОПЕРАТОРЫ ЯЗЫКА БЕЙСИК	29
3.4.1. функции в языке БЕЙСИК	32
3.4.2. Сообщение об ошибках	34

Руководство оператора

1. НАЗНАЧЕНИЕ ПРОГРАММ.....	36
2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММ	36
3. ВЫПОЛНЕНИЕ ПРОГРАММ	36
3.1. МОНИТОР.....	37
3.1.1. Директивы МОНИТОРА	37
3.2. РЕДАКТОР ТЕКСТОВ И АССЕМБЛЕР (ПАКЕТ МИКРОН).....	48
3.2.1. Загрузка редактора в ОЗУ.....	48
3.2.2. Запуск редактора.....	48
3.2.3. Директивы редактора.....	48
3.2.3.1. Режим приема директив	48
3.2.3.2. Режим ввода строк.....	49
3.2.3.3. Редактирование текста	49
3.2.3.4. Удаление и перенос строк.....	49
3.2.3.5. Директивы ассемблера.....	49
3.3. ТЕСТ ФУНКЦИОНИРОВАНИЯ	50
3.4. ТЕСТ ОЗУ	50
3.5. ТЕСТ ПЗУ	51
3.6. ТЕСТ МАГНИТОФОНА	51
3.7. МОНИТОР-SP	52
3.8. ИГРОВАЯ ПРОГРАММА "РЕВЕРСИ"	52
3.9. ИГРОВАЯ ПРОГРАММА "ПИТОН"	54
3.10. ИГРОВАЯ ПРОГРАММА "ПОДВОДНАЯ АТАКА"	55
3.11. ИГРОВАЯ ПРОГРАММА "ШАХМАТЫ"	55
3.12. ИГРОВАЯ ПРОГРАММА "КЛАД"	56
3.13. ИГРОВАЯ ПРОГРАММА "TETRIS"	56
3.14. ИГРОВАЯ ПРОГРАММА "ПЕЩЕРА"	57
3.15. ИГРОВАЯ ПРОГРАММА "ШАХМАТЫ2"	57
3.16. ИГРОВАЯ ПРОГРАММА "БИЗНЕС"	58
3.17. ИГРОВАЯ ПРОГРАММА "БУРАН"	59

ФОРМУЛЯР

АННОТАЦИЯ

В формуляре содержится перечень документов, описывающих работу комплекта программного обеспечения радиоконструктора "Бытовое электронно-вычислительное устройство ЛИК" (в дальнейшем радиоконструктор ЛИК).

1. ОБЩИЕ СВЕДЕНИЯ

Комплект программного обеспечения радиоконструктора ЛИК 3.060.004-01,
предприятие-изготовитель _____

2. ОБЩИЕ УКАЗАНИЯ И ОСНОВНЫЕ ХАРАКТЕРИСТИКИ

1) Данный документ содержит перечень программных документов и носителей программного обеспечения, необходимых для эксплуатации радиоконструктора ЛИК.

Поставка комплекта программного обеспечения радиоконструктора ЛИК осуществляется на миникассете 3.060.004-01 12 01.

2) Перед эксплуатацией необходимо ознакомиться с документами программного обеспечения, перечисленными в разд. 3.

3. КОМПЛЕКТНОСТЬ

Обозначение	Наименование	Кол.	Примечание
3.060.004-01 12 01	Радиоконструктор "Бытовое электронно-вычислительное устройство ЛИК".Комплект программного обеспечения.Текст программы. Документация программного обеспечения	1	Миникассета
3.060.004-01 30 01	Радиоконструктор "Бытовое электронно-вычислительное устройство ЛИК".Комплект программного обеспечения. Формуляр.	1	
3.060.004-01 31 01	Радиоконструктор "Бытовое электронно-вычислительное устройство ЛИК".Комплект программного обеспечения. Описание применения.	1	
3.060.004-01 33 01	Радиоконструктор "Бытовое электронно-вычислительное устройство ЛИК". Комплект программного обеспечения. Руководство программиста.	1	
3.060.004-01 34 01	Радиоконструктор "Бытовое электронно-вычислительное устройство ЛИК"- Комплект программного обеспечения. Руководство оператора.	1	

4. СВИДЕТЕЛЬСТВО О ПРИЕМКЕ

Комплект программного обеспечения радиоконструктора ЛИК 3-060-004-01 соответствует техническим условиям ТУ 25.7107.0004-89 и признан годным для эксплуатации.

М.П.

Дата выпуска _____

Представитель службы
технического
контроля _____

5. ГАРАНТИЙНЫЕ ОБЯЗАТЕЛЬСТВА

5.1. Комплект программного обеспечения радиоконструктора ЛИК должен быть принят техническим контролем предприятия-изготовителя.

5.2. Предприятие-изготовитель гарантирует соответствие программного обеспечения требованиям программной документации при соблюдении потребителем условий эксплуатации, транспортирования и хранения.

6. СВЕДЕНИЯ О РЕКЛАМАЦИЯХ

Рекламации должны предъявляться предприятию-изготовителю согласно "Инструкции о порядке приемки продукции производственно-технического назначения и товаров народного потребления по качеству", утвержденной постановлением N П-7 Государственного арбитража при Совете Министров СССР от 25 апреля 1966 года.

Описание применения

АННОТАЦИЯ

В данном документе приведены сведения о составе и назначении комплекта программного обеспечения радиоконструктора "Бытовое электронно-вычислительное устройство ЛИК" (в дальнейшем радиоконструктор ЛИК).

1. НАЗНАЧЕНИЕ ПРОГРАММЫ

Комплект программного обеспечения предназначен для:

Освоения основ программирования на языках "БЕЙСИК" и "АСЕМБЛЕР";

Подготовки, редактирования и выполнения программ, составленных на языке "БЕЙСИК" и "АСЕМБЛЕР";

Считывания и записи программ на магнитную ленту.

Комплект включает в себя следующие программные модули:

программа МОНИТОР-1М;

программа МОНИТОР-SP;

интерпретатор языка БЕЙСИК;

редактор текста и АСЕМБЛЕР (пакет "МИКРОН");

тестовые программы;

игровые программы.

2. УСЛОВИЯ ПРИМЕНЕНИЯ

Комплект программного обеспечения реализуется при наличии следующего состава оборудования:

радиоконструктор ЛИК;

бытовой кассетный магнитофон;

бытовой телевизионный приемник;

источник питания;

соединительные жгуты.

3. ОПИСАНИЕ ПРОГРАММЫ

Комплект программного обеспечения представляет собой ряд программных модулей, записанных на миникассету 3.060.004-01 12 01.

В таблице приведен перечень программных модулей, входящих в состав комплекта.

Номер программы	Имя программы	Тип записи	Тип носителя	Сторона кассеты	Наименование
1	МОНИТОР-1М	A	миникассета	A	Управляющая программа "МОНИТОР-1М"
2	Каталог.A	A	миникассета	A	Каталог программ, поставляемых потребителю (сторона A кассеты)
3	Каталог 0	A	миникассета	A	Чистый каталог для потребительских программ
4	Редактор	A	миникассета	A	Редактор текстов и ассемблер (пакет "МИКРОН")
5	БЕЙСИК	A	миникассета	A	Интерпретатор языка БЕЙСИК
6	МОНИТОР-SP	A	миникассета	A	Управляющая программа "МОНИТОР-SP"
7	ТЕСТФ	A	миникассета	A	Тест функционирования
8	ТЕСТОЗУ	A	миникассета	A	Тест ОЗУ
9	ТЕСТПЗУ	A	миникассета	A	Тест ПЗУ
10	ТЕСТМАГ	A	миникассета	A	Тест магнитофона

Номер программы	Имя программы	Тип записи	Тип носителя	Сторона кассеты	Наименование
11	Каталог . B	A	миникассета	B	Каталог поставляемых программ (сторона B)
12	РЕВЕРСИ	A	миникассета	B	Игровая программа "РЕВЕРСИ"
13	ШАХМАТЫ	A	миникассета	B	Игровая программа "ШАХМАТЫ"
14	ПИТОН	A	миникассета	B	Игровая программа "ПИТОН"
15	АТАКА	A	миникассета	B	Игровая программа "ПОДВОДНАЯ АТАКА"
16	КЛАД	A	миникассета	B	Игровая программа "Клад"
17	TETRIS	A	миникассета	B	Игровая программа "TETRIS"
18	ПЕЩЕРА	A	миникассета	B	Игровая программа "Пещера"
19	ШАХМАТЫ2	A	миникассета	B	Игровая программа "Шахматы2"
20	БИЗНЕС	A	миникассета	B	Игровая программа "Бизнес"
21	БУРАН	A	миникассета	B	Игровая программа "Буран"

В графе "Номер программы" указан порядковый номер программы, записанной на миникассете. В графе "Тип записи" указано:

A - программа в абсолютных кодах, загружаемая и выполняемая командами монитора;

C - программа в символьных кодах, загружаемая и выполняемая командами интерпретатора языка БЕЙСИК.

Перечень программных модулей, приведенных в таблице, записан на стороне A и стороне B миникассеты 3-060.004-01 12 01.

Руководство программиста

АННОТАЦИЯ

Данное руководство содержит сведения о языках программирования АСSEMBЛЕР и БЕЙСИК.

Описаны возможности радиоконструктора "Бытовое электронно-вычислительное устройство ЛИК" (в дальнейшем радиоконструктор ЛИК) используемые АСSEMBЛЕРом и интерпретатором языка БЕЙСИК.

Дано описание программно-доступных регистров радиоконструктора ЛИК, представление программы в памяти, адресации памяти.

Перечислены элементы АСSEMBЛЕРА и языка программирования БЕЙСИК. Приведена система команд радиоконструктора ЛИК, список сообщений АСSEMBЛЕРА и БЕЙСИКа об ошибках.

1. СТРУКТУРНОЕ ОПИСАНИЕ РАДИОКОНСТРУКТОРА ЛИК

Содержание этой части даст программисту необходимый минимум знаний для изучения всего дальнейшего материала и написания программ.

Для программиста радиоконструктор ЛИК состоит из следующих частей:

- 1) Семь рабочих регистров, в которых могут находиться данные или адреса ячеек памяти. Все действия над данными могут производиться только в регистрах.
- 2) Память, в которой могут находиться команды и (или) данные. Память состоит из адресуемых ячеек.
- 3) Программный счетчик — 16-ти битовый регистр, содержимое которого указывает адрес следующей выполняемой команды.
- 4) Указатель стека - 16-ти битовый регистр, позволяющий адресовать различные участки памяти, используемые как стеки. Это облегчает обращение к подпрограммам и обработку прерываний.
- 5) Ввод/вывод, осуществляющий связь радиоконструктора ЛИК с внешними устройствами.

1.1. Терминология

БИТ - минимальная единица информации. Может принимать значение 1 или 0 (1 двоичный разряд).

БАЙТ - группа из восьми рядом стоящих битов. Минимальная адресуемая ячейка памяти.

СЛОВО - группа из двух рядом стоящих байтов. Адресом слова считается адрес первого по порядку байта.

АДРЕС - 16-ти битовое число — порядковый номер данного байта памяти.

КОМАНДА - элементарная операция, которую может выполнять радиоконструктор ЛИК.

ПРОГРАММА - последовательность команд, выполняя которые радиоконструктор ЛИК решает определенную задачу.

ОБЪЕКТНАЯ ПРОГРАММА - программа, которую можно непосредственно загрузить в оперативную память и начать выполнять.

ИСХОДНАЯ ПРОГРАММА - написанная программистом программа, которую необходимо преобразовать в объектную программу, чтобы ее мог выполнить радиоконструктор ЛИК.

СИСТЕМНАЯ ПРОГРАММА — объектная программа, которая предназначена для помощи пользователю в пропуске его программ на радиоконструкторе ЛИК.

NNNN - число в десятичной системе счисления.

NNNNQ - число в восьмеричной системе счисления.

NNNNH - число в шестнадцатеричной системе счисления.

1.2. Рабочие регистры

Радиоконструктор ЛИК имеет 8-ми битовый аккумулятор и, кроме того, шесть 8-ми битовых регистров. Эти регистры обозначаются числами 0,1,2,3,4,5 и 7. Кроме того, по соглашению эти регистры обозначаются соответственно буквами В, С, D, E, H, L и А (аккумулятор). Некоторые операции можно производить с парами регистров. Эти пары обозначаются буквами В, D, H и PSW. Соответствие пар и регистров следующее:

регистровая пара	регистры
В	В и С (0 и 1)
D	D и E (2 и 3)
H	H и L (4 и 5)
PSW	А и байт состояния

Регистровая пара PSW (PROGRAM STATUS WORD) соответствует аккумулятору А и специальному байту, который отражает текущее состояние машинных флагов. Этот байт подробнее описан ниже.

1.3. Память

Радиоконструктор ЛИК может работать как с памятью только для чтения (ПЗУ), так и с памятью для чтения/записи (ОЗУ). Программа может читать данные из памяти любого типа, а записывать только в ОЗУ.

Для программиста память представляет собой последовательность байтов, где могут находиться данные или команды. Адреса байтов памяти могут быть от 0 до 65535=FFFFH.

В радиоконструкторе ЛИК память распределена следующим образом:

ОЗУ: 0000H-80FFH - рабочая область для программиста;
0000H-1DFFH - область загрузки "BASIC";
0000H-0FFFH - область загрузки "РЕДАКТОРА";
1100H-20FFH - область трансляции;
2100H - начало текстового буфера;
8100H-8B3FH - область загрузки "МОНИТОР-1М" с ленты;
8B40H-8F3FH - область каталога ленты;
9000H-BFFFFH - область экранного ОЗУ.

ПЗУ: C000H-C7FFH - "ЗАГРУЗЧИК";
C800H-D23FH - "МОНИТОР-1М";
D240H-EFFFFH - "BASIC".

Рабочие ячейки ОЗУ:

8FFFFH — константа ввода (начальное значение ЗСН);
8FFEH — константа вывода (начальное значение 28H);
8FFAH, 8FFBH — код цвета экрана после очистки;
00 — черный;
FF — белый;
8FFCH, 8FFDH — координаты курсора (вертикальная и горизонтальная) .

1.4. PC программный счетчик

Программный счетчик (PC) представляет собой 16—ти битовый регистр, который доступен программисту и содержимое, которого, указывает адрес следующей выполняемой команды.

1.5. SP указатель стека

Стек представляет собой область памяти, выделенную программистом, в которую с помощью стековых операций могут заноситься или из которой могут считываться данные или адреса. Наличие стека облегчает обращение к подпрограммам и обработку прерываний. Начальный адрес стека задается программистом с помощью специального 16-ти битового регистра, называемого указателем стека (SP - STACK POINTER).

1.6. Ввод/вывод

К радиоконструктору ЛИК подключаются два устройства ввода/вывода (магнитофон и телевизионный приемник). Эти устройства передают и принимают данные через центральный процессор.

К магнитофону можно обращаться через команды "МОНИТОРА—1М", которые используют стандартные подпрограммы "ЗАГРУЗЧИКА". Вывод информации на телевизионный приемник осуществляется также стандартными подпрограммами "ЗАГРУЗЧИКА".

Основные подпрограммы "ЗАГРУЗЧИКА" приведены в табл.1.

Таблица 1.

Назначение	Адрес вызова	Параметры	
		входные	выходные
Ввод символа с клавиатуры	C337H	нет	A=код символа
Вывод символа на дисплей	C037H	C=код символа	нет
Ввод байта с магнитофона	C377H	A=FF - с поиском синхробайта A=08 - БЕЗ	A=считанный байт
Вывод байта на магнитофон	C3D0H	A=выводимый байт	нет
Ввод блока с магнитофона по адресам, записанным на ленте	C3F9H	нет	Ячейки 8FE3H, 8EE4H- адрес начала, DE=адрес конца блока
Ввод блока с магнитофона	C422H	HL=адрес начала DE=адрес конца области памяти	нет
Сравнение пар HL и DE	C427H	HL, DE	F=результат сравнения
Вывод строки символов	C438H	HL-адрес начала строки символов признак конца строки - 00H	HL-адрес начала
Очистка экрана	C010H	нет	нет

1.7. Представление программы в памяти

Программа состоит из последовательности команд. Каждая команда дает возможность выполнить элементарную операцию, такую как пересылка байта данных, арифметическую или логическую операцию над байтом данных или изменение порядка выполнения команд. Программа будет записана в ПАМЯТЬ как последовательность битов, которые представляют команды программы и которые представлены шестнадцатеричными цифрами. Адрес памяти со следующей выполняемой командой находится в программном счетчике. Непосредственно перед выполнением каждой команды в программный счетчик помещается адрес следующей команды.

Программа выполняется последовательно, если не выполняется команда передачи управления (переход, вызов, или возврат), которая устанавливает новое значение программного счетчика. Изучая содержимое байта памяти, нельзя различить находится ли в этом байте код команды или данные. Только исходя из логики программы, можно достоверно отличить данные от кодов команд. Машинные команды могут требовать 1,2 или 3 байта для кода команды.

В каждом таком случае программный счетчик автоматически увеличивается на соответствующее число до адреса следующей команды. Чтобы избежать ошибок, пользователь должен быть уверен, что за командой не идет байт данных, когда ожидается следующая команда.

Ряд команд, называемых командами передачи управления, вызывает переход к команде, которая может находиться в любой области памяти.

1.8. Адресация памяти

- 1) Прямая адресация. При прямой адресации в команде указан точный адрес ячейки памяти.
- 2) Адресация с помощью пары регистров. Адрес ячейки памяти может быть задан содержимым регистровой пары.
- 3) Адресация с помощью указателя стека.
- 4) Непосредственная адресация. Данные содержатся в самой команде.

1.9. Байт состояния

Этот байт содержит пять булевых переменных, называемых флажками состояния. Они приведены в таблице 2.

Таблица 2.

Номер байта	Буква	Название
7	S	Знак
6	Z	Нуль
4	A	Добавочный перенос
2	P	Четность
0	C	Перенос

Эти флажки изменяются в результате выполнения команд. Для удобства каждый флаг имеет стандартную интерпретацию, в соответствии с которой действует большинство команд.

Z – бит, или бит нуля, устанавливается в “1”, если результат действия команды равен нулю. В противном случае он устанавливается в “0”.

S - бит, или бит знака, устанавливается в “1”, если наибольший значащий бит результата равен единице. В противном случае он устанавливается в “0”.

P - бит, или бит четности, устанавливается в “1”, если результат содержит четное число единиц.

C - бит, или бит переноса, устанавливается в “1”, если после выполнения арифметической операции генерируется либо перенос, либо заем.

A - бит, или бит добавочного переноса. Устанавливается в “1”, если возник перенос из 3-го разряда в 4-й.

2. Описание АССЕМБЛЕРА

Программа написанная на АССЕМБЛЕРЕ представляет собой последовательность строк, каждая из которых содержит или приказ, или команду, или константу, или псевдокоманду.

Программа написанная на АССЕМБЛЕРЕ программистом называется исходной программой. Она заносится в ОЗУ при помощи "Редактора" с магнитофона, или вводится вручную с клавиатуры. АССЕМБЛЕР преобразует исходную программу в эквивалентную объектную программу. Каждая преобразованная команда или константа получает свой адрес, величина которого определяется значением внутренней переменной, называемой счетчиком адреса.

2.1. Основные элементы языка

2.1.1. Символы

Все сложные элементы языка АССЕМБЛЕР формируются из символов. В качестве символов могут использоваться:

буквы латинского алфавита; цифры от 0 до 9;

специальные знаки **+ - , ; : () пробел ;**

в примечаниях любые символы.

2.1.2. Числа, константы и символические имена

Восьмеричные числа образуются из цифр от 0 до 7 и оканчиваются буквой Q или O.

Десятичные числа образуются из цифр от 0 до 9.

Шестнадцатеричные числа образуются из цифр от 0 до 9 и букв от A до F.

Шестнадцатеричное число должно обязательно начинаться с цифры и оканчиваться буквой H.

Все виды чисел могут иметь знак + или -. Отрицательные числа в памяти представляются в дополнительном коде. Поэтому однобайтное число, рассматриваемое как целое со знаком, может быть в диапазоне от -128 до +127. Двухбайтное целое число со знаком будет представлено в диапазоне от -32768 до +32767. Однобайтное число без знака представляется в диапазоне от 0 до 256. Двухбайтное число без знака представляется в диапазоне от 0 до 65535.

Константой называется символ, заключенный в апострофы.

Символическим именем называется последовательность букв и цифр, начинающаяся с букв. Количество букв в имени неограниченно, но идентифицируют имя только первые пять знаков.

2.2. Синтаксис ассемблерных предложений

Ассемблерная программа состоит из последовательности предложений. Каждое предложение принадлежит одной из трех групп: команд, псевдокоманд и директив ассемблера. Каждая команда состоит из четырех отдельных и отличающихся друг от друга полей, расположенных в следующем порядке:

поле имени : поле кода поле операндов ; комментарий

поле имени и комментарий необязательны.

2.2.1. Псевдокоманды и директивы

Псевдокоманды предназначены для указания АССЕМБЛЕРУ и для задания значений имен в ассемблерной программе. Псевдокоманды не генерируются в объектные коды.

2.2.1.1. ORG установить начальное значение счетчика адреса

Чтобы установить значение счетчика адреса, необходимо написать следующее предложение:

name: ORG выражение ; комментарий

где "name" и "комментарий" необязательны, а "выражение" есть число, символическое имя или выражение. Следующая команда будет ассемблироваться с адреса, определяемого величиной "выражение".

2.2.1.2. DS ЗАРЕЗЕРВИРОВАТЬ ОБЛАСТЬ ПАМЯТИ

Чтобы зарезервировать некоторое число байтов памяти, необходимо написать следующее предложение:

LABEL: DS выражение ; комментарий

где "LABEL" и "комментарий" необязательны, а "выражение" есть число, символическое имя или выражение. При ассемблировании в эту область памяти не будут заноситься данные. Следующая команда будет ассемблироваться с адреса, вычисляемого как сумма адреса данной области и величины "выражение".

2.2.1.3. EQU задать значение символического имени

Чтобы ранее неопределенному имени присвоить некоторое значение, применяется псевдокоманда EQU:

NAME : EQU выражение ; комментарий

АССЕМБЛЕР присваивает имени "NAME" значение "выражение". Каждый раз, когда имя "NAME" встречается при дальнейшем ассемблировании, будет использоваться присвоенное значение.

2.2.1.4. DB задать байт данных

Для того, чтобы задать последовательность однобайтовых элементов данных, необходимо написать предложение:

NAME : DB список ; комментарий

где "NAME" и "комментарий" необязательны, а "список" является списком:

- 1) чисел, имен и выражений, которые преобразуются в восьмибитовые элементы данных;
- 2) знаков КОИ-7, заключенных в апострофы. Различные элементы списка должны разделяться запятыми. Восьмибитовые значения, вырабатываемые списком данных, будут последовательно запоминаться в памяти, начиная с байта, адресованного именем "NAME". Знаки КОИ-7 приведены в таблице 3.

2.2.1.5. DW задать слово данных

Для того, чтобы задать последовательность двухбайтовых элементов данных (слов данных), необходимо написать предложение

NAME: DW список ; комментарий

где "NAME" и "комментарий" необязательны, а "список" является списком чисел, имен и выражений, которые преобразуются в шестнадцатибитовые элементы данных.

2.2.1.6 END конец исходной записи

Предложение END сообщает АССЕМБЛЕРУ, что достигнут конец одной физической записи программы и что генерацию программы из этой записи следует закончить. Предложение имеет вид:

NAME : END ; комментарий

Одна запись программы по определению есть последовательность строк, заканчивающаяся предложением END. Ассемблерная программа может состоять из нескольких записей, каждая из которых должна оканчиваться предложением END.

Таблица 3.

	2	3	4	5	6	7
0		0	@	P	Ю	П
1	!	1	A	Q	А	Я
2	"	2	B	R	Б	Р
3	#	3	C	S	Ц	С
4	\$	4	D	T	Д	Т
5	%	5	E	U	Е	У
6	&	6	F	V	Ф	Ж
7	'	7	G	W	Г	В
8	(	8	H	X	Х	Ь
9	)	9	I	Y	И	Ы
A	*	:	J	Z	Й	З
B	+	;	K	[	К	Ш
C	,	<	L	\	Л	Э
D	-	=	M	]	М	Щ
E	.	>	N	^	Н	Ч
F	/	?	O	_	О	■

2.3. Система команд

Ниже в таблицах представлены команды АССЕМБЛЕРА. Каждая команда приводится вместе с ее форматом, функциональным описанием и воздействием на флаги.

В таблице 4. приведены символы, применяемые при описании команд.

Таблица 4.

Символ	Значение
=	Замена
<=>	Обмен операндов
⋈	Логическое объединение элементов
<AND>	Поразрядная операция "И"
<OR>	Поразрядная операция "ИЛИ"
<XOR>	Поразрядная операция "ИСКЛЮЧАЮЩЕЕ ИЛИ"
<NOT>	Дополнение до единицы
[X]	Однобайтовый операнд в порте ввода/вывода
(X)	Однобайтовый операнд в ячейке памяти
(X, Y)	Двухбайтовый операнд в ячейках X и Y (X старший байт)
M	Операнд в ячейке памяти, адрес которой задан в паре регистров HL
+	флаг изменяется в соответствии с результатом
N	Однобайтовое целое число без знака
NN	Двухбайтовое целое число без знака
D	Однобайтовое целое число со знаком
CC	C, Z, A, P, S (Мнемоническое значение флагов)
R	A, B, C, D, E, H или L регистры
R'	То же
RM	A, B, C, D, E, H, L или M
RR	B, D, H, или SR (пары регистров)
SS	B, D, H или PSW (пары регистров)
IE	Триггер разрешения прерывания

Команды загрузки и обмена приведены в табл.5.

Таблица 5.

формат команды	функциональное описание
LDA NN	A= (NN)
LDAX B	A= (BC)
LDAX D	A= (DE)
MVI R,N	R=N
MOV R,R'	R=R'
MOV R,M	R= (HL)
STA NN	(NN) =A
STAX B	(BC) =A
STAX D	(DE) =A
MVI M,N	(HL) =N
NOV M,R	(HL) =R
LXI RR,NN	RR=NN
LHLD NN	HL= (NN+1, NN)
SHLD NN	(NN+1, NN) =HL
SPHL	SP=HL
PUSH SS	(SP-1, SP-2) =SS, SP=SP-2
POP SS	SS= (SP+1, SP) , SP=SP+2
XCHG	DE <=> HL
XTHL	HL <=> (SP+1, SP)

Команды арифметики приведены в табл.6.

Таблица 6.

формат команды	функциональное описание	Условия				
		C	Z	P	A	S
ADI N	A=A+N	+	+	+	+	+
ADD RM	A=A+RM	+	+	+	+	+
ACI N	A=A+N+C	+	+	+	+	+
ADC RM	A=A+RM+C	+	+	+	+	+
SUI N	A=A-N	+	+	+	+	+
SUB RM	A=A-RM	+	+	+	+	+
SBI N	A=A-N-C	+	+	+	+	+
SBB RM	A=A-RM-C	+	+	+	+	+
DAA	Коррекция A	+	+	+	+	+
INR RM	RM=RM+1	-	+	+	+	+
DCR RM	RM=RM-1	-	+	+	+	+
DAD RR	HL=HL+RR	+	-	-	-	-
INX RR	RR=RR+1	-	-	-	-	-
DCX RR	RR=RR-1	-	-	-	-	-

Логические команды приведены в табл.7.

Таблица 7.

формат команды	функциональное описание	Условия				
		C	Z	P	A	S
ANI N	A=A<AND>N	+	+	+	-	+
ANA RM	A=A<AND>RM	+	+	+	-	+
ORI N	A=A<OR>N	+	+	+	-	+
ORA RM	A=A<OR>RM	+	+	+	-	+
XRI N	A=A<XOR>N	+	+	+	-	+
XRA RM	A=A<XOR>RM	+	+	+	-	+
CPI N	A $\neq$ N	+	+	+	+	+
CMP RM	A $\neq$ RM	+	+	+	-	+
CMA	A=<NOT>A	-	-	-	-	-
CMC	C=<NOT>C	+	-	-	-	-
STC	C=1	+	-	-	-	-

Команды управления микропроцессором приведены в табл. 8.

Таблица 8.

формат команды	функциональное описание
NOP	нет операции
NLT	останов микропроцессора
DI	IE=0
EI	IE=1

Команды циклического сдвига приведены в табл.9.

Таблица 9.

формат команды	функциональное описание	Условия				
		C	Z	P	A	S
RLC	Циклический сдвиг A влево	+	-	-	-	-
RAL	Циклический сдвиг C-A влево	+	-	-	-	-
RRC	Циклический сдвиг A вправо	+	-	-	-	-
RAR	Циклический сдвиг A-C вправо	+	-	-	-	-

Команды перехода приведены в табл.10.

Таблица 10.

формат команды	функциональное описание
JMP NN	PC=NN
JCC NN	PC=NN, если значение CC "истина"
PCHL	PC=HL

Команды вызова и возврата приведены в табл. 11.

Таблица 11.

формат команды	функциональное описание
CALL NN	(SP-1, SP-2)=PC, SP=SP-2, PC=NN
CCC NN	вызов, если значение CC «истинно»
RET	PC=(SP+1, SP) SP=SP+2
RCC	возврат, если значение CC «истинно»
RST P	(SP-1, SP-2)=PC, SP=SP-2, PC=P

Команды ввода – вывода приведены в табл.12.

Таблица 12.

формат команды	функциональное описание
IN N	A= [N]
OUT N	[N]=A

2.4. Трансляция и запуск программы

Транслятор с языка АССЕМБЛЕР входит в состав пакета "МИКРОН". После ввода исходной программы в текстовый буфер редактора, ее необходимо оттранслировать. Транслятор вызывается нажатием клавиши "  ". Для начала трансляции необходимо нажать на клавишу с номером режима трансляции.

Режимы трансляции:

1- трансляция с отображением протокола;

2- трансляция с отображением таблицы меток;

3- трансляция с выводом количества ошибок и адресов оттранслированной программы.

Запуск оттранслированной программы осуществляется директивой "МОНИТОРА-1М" J1100 (если в программе отсутствует псевдокоманда ORG).

Во время трансляции могут возникать ошибки. Расшифровка кодов ошибок во время трансляции следующая:

01* — двойное определение метки;

02* — метка неопределенна;

04* — несуществующая мнемоника;

08* — неправильный операнд;

A* - недопустимый символ в метке.

Для более детального изучения языка программирования АССЕМБЛЕР необходимо пользоваться дополнительной литературой.

3. Интерпретатор языка БЕЙСИК

БЕЙСИК является одним из самых популярных языков для персональных микро-ЭВМ. Широкое применение БЕЙСИКА в компьютерах такого класса объясняется тем, что для его работы не требуется большого объема оперативной памяти машины, так как БЕЙСИК является интерпретатором, а это значит, что каждая команда (оператор) языка транслируется (интерпретируется) в машинные коды в процессе выполнения программы. Это значительно экономит объем памяти компьютера, так как в ОЗУ одновременно находятся только транслятор языка, текст программы и переменные, используемые программой".

Недостатком интерпретатора является низкая скорость выполнения программ, но в работе интерпретатор значительно удобнее. Он позволяет быстро проверить только что написанную программу или ее отдельные части, и, при необходимости, оперативно ввести исправления в программу. Кроме того, возможен непосредственный режим — строка команд выполняется сразу после ее введения и нажатия клавиши <ВК>.

3.1. Программа на языке БЕЙСИК

Программа на БЕЙСИКЕ состоит из последовательности пронумерованных строк. Им принято присваивать номера с интервалом равным 10. Это может оказаться полезным, если в процессе отладки программы потребуется вставить несколько промежуточных строк. Номера могут быть любыми от 0 до 65529. Каждая строка может состоять из одного или нескольких операторов, предписывающих интерпретатору определенные действия. Если операторов несколько, их отделяют друг от друга символом " : " (двоеточие).

3.2. Операнды, переменные и выражения

В качестве операндов выступают выражения, составленные из констант и переменных. В БЕЙСИКЕ существует два типа констант - числовые и символьные. Числовые константы — это любые десятичные числа в интервале от -1.7E38 до +1.7E38, символьные - последовательность любых символов, заключенная в кавычки. Например, "значение", "введите X", "F=" — символьные константы, 3.14, 180, -10, 2.33E-1, 7E23 - числовые. Две последние константы заданы в экспоненциальной форме, на что указывает знак E, за которым следует знак и величина порядка (знак + можно опускать). Точность задания констант - 6 значащих цифр.

Переменные в программах используются также двух типов: числовые и символьные. Обращаются к переменным по имени, которое состоит из одного или двух символов. Первый из них обязательно должен быть буквой латинского алфавита, второй - буквой того же алфавита или цифрой. Символьные переменные после имени содержат знак \$. Например:

A, X, X1 - допустимые имена числовых переменных,

A\$, NM\$, F\$ - символьных.

Группе переменных одного типа может быть присвоено общее имя. Их в этом случае называют переменными с индексами или массивами. Для обращения к каждой переменной в массиве используют один или несколько индексов (индекс - номер переменной в массиве). Наименьшее значение индекса равно 0, а наибольшее определяется размерами массива. Если индекс один то массив называется одномерным, два — двухмерным и т.д. Индексы необходимо указывать в круглых скобках после имени массива. Например:

A(3) - третий элемент одномерного массива A

RS(2,3) - элемент, стоящий на пересечении 2-ой строки и 3-го столбца двухмерного массива RS

NM\$ (6) - шестой элемент одномерного массива NM\$

При работе в памяти компьютера под каждый массив отводится определенная область, поэтому перед использованием массива нужно задать его размер оператором DIM.

Переменные и константы образуют выражения. Кроме них, в выражения входят знаки операций, скобки и имена функций. Все операции можно разделить на 3 типа.

1) арифметические: сложение (+), вычитание (—), умножение (*), деление (/), возведение в степень (^);

2) операции отношения: больше (>), меньше (<), равно (=), не равно (<>), больше или равно (>=), меньше или равно (<=);

3) логические: отрицание "не" (NOT), логическое умножение "и" (AND), логическое сложение "или" (OR).

Для символьных переменных и констант разрешены только операции отношения и конкатенации (слияния), обозначаемые знаком " + ".

При вычислении результата выражения все операции имеют определенный приоритет. Ниже показан приоритет операций:

1) ^

2) * , /

3) + , -

4) = , <> , > , < , >= , <=

5) NOT

6) AND

7) OR

Самый высокий приоритет имеет операция возведения в степень (^). Изменить порядок вычислений можно с помощью скобок.

Одним из основных операторов во всех алгоритмических языках является оператор присваивания. В БЕЙСИКЕ это - оператор LET. Так, например, запись

```
LET A=4
```

означает, что переменной A присваивается значение 4. В данной версии языка оператор LET не используется, поэтому нужно писать просто

```
A=4
```

Интерпретатор языка БЕЙСИК совмещает в себе функцию редактора текста и отладчика. Перед вводом новой программы необходимо сначала стереть старую директивой NEW, а затем набрать с клавиатуры программу, ставя перед каждой новой строкой ее номер. Впрочем, номера строк можно проставлять автоматически, воспользовавшись директивой AUTO, вызываемой последовательным нажатием клавиш " ␣ " и "A". При этом на экран выводится сообщение

```
AUTO*
```

и номер строки. Теперь нужно ввести строку и по окончании ввода нажать клавишу "BK". Строка занесется в память, а на дисплей будет выведен номер следующей строки, равный предыдущему +10. При необходимости номер строки можно изменить, пользуясь клавишами "←" и "→". При вводе можно исправлять ошибки, двигая курсор по строке клавишами "←", "→". Выйти из автонумерации можно двукратным нажатием клавиши "␣".

Просмотреть текст программы можно при помощи директивы LIST. Для удаления строки достаточно набрать на клавиатуре ее номер и нажать <BK>, а чтобы вставить в текст новую, нужно просто ввести ее с клавиатуры. Если строка с таким номером уже существует, то новая становится на место старой. Если надо исправить строку, необходимо воспользоваться директивой EDIT, она вызывается последовательным нажатием клавиш "↵" и "E", затем, после вывода сообщения

EDIT*

ввести номер строки и нажать "BK". Изменения можно вносить пользуясь клавишами "←" и "→". Удалить правую от курсора часть строки можно нажатием клавиш "↵" и "BK", удалить всю строку - "↵" и "PC". После нажатия клавиши "BK" строка вновь выводится на повторную проверку. Если теперь нажать клавишу "BK", то на редакцию будет вызвана очередная строка. Выйти из режима редактирования можно клавишей "↵".

При редактировании можно раздвигать строку клавишей "↵" и удалять стоящий слева от курсора символ клавишей "ЗБ", но визуально оценить результат этих операций можно после нажатия на клавишу "BK", когда строка выведется на повторную проверку.

Если во время редактирования изменить номер строки, то строка с прежним номером останется неизменной, а в память занесется строка с новым номером после нажатия "BK". Таким образом можно перенумеровать строки программы.

Во время ввода строки можно вводить операторы и имена функций последовательным нажатием двух клавиш "↵" и клавиши в ниже приведенном соответствии:

J - GOTO	C – CUR	U - USR	K - CLEAR
E - EDIT	N – NXT	G - GOSUB	[- LEFT\$
] - RIGHT\$	Z - SPC(	H - CHR\$	F - FOR
Y - POKE	W – RND	A - AUTO	P - PLOT
R - RETURN	O – MSAVE	L - LINE	D - DATA
V - VAL	Q – READ	S - STEP	M - MID\$
I - INPUT	T – THEN	X - PEEK	B - TAB(
@ - MLOAD	; - REM	1 - DIM	2 - CLS
3 - STOP	4 – DEF	5 - INT	6 - AND
7 - ABS	8 – SQR	9 - EXP	! - LIST
" - RUN	# - CONT	\$ - NEW	% - RESTORE
& - DPL	ПРОБЕЛ – PRINT		

Отладка программы на БЕЙСИКЕ производится также при помощи интерпретатора. Для этой цели предусмотрен перевод интерпретатора из режима выполнения программы в непосредственный при выполнении оператора STOP или двукратном нажатии клавиши "↵". После этого в непосредственном режиме можно просмотреть и изменить содержимое переменных, распечатать программу на экране по директиве LIST и затем продолжить выполнение программы с того места, где она была прервана директивой CONT. Нельзя только модифицировать текст программы, так как в этом случае продолжение работы по директиве CONT уже запрещено.

3.3. Директивы языка БЕЙСИК

Рассмотрим теперь системные директивы БЕЙСИКА. Далее по тексту скобки будут означать, что данные параметры в директиве могут отсутствовать. В этом случае используют соглашения по умолчанию - то есть, в случае отсутствия параметра директива использует его определенное значение для этой директивы.

NEW - стирание старой программы и подготовка для ввода новой. Обязательна при первом запуске интерпретатора.

RUN(N) - запуск программы со строки с номером N. Если номер не указан, программа запускается сначала.

LIST(N) - распечатка программы со строки с номером N. Приостановить работу директивы можно нажатием клавиши "STOP", продолжить - клавишей "BK". Прекратить распечатку можно клавишей "⏏".

EDIT - редактирование строки.

CONT - продолжение выполнения программы после ее останова оператором STOP или с клавиатуры.

AUTO - включение автонумерации строк. Выход из режима автонумерации - клавиша "⏏".

MSAVE (имя) - вывод текста программы на магнитофон. Имя должно состоять из одного символа, впрочем, его можно не указывать, тогда и при последующей загрузке программы имя указывать не нужно.

MLOAD (имя) - ввод программы с магнитной ленты. Если имя указано, производится поиск программы с этим именем. Если нет - загружается первая попавшаяся программа, записанная без имени.

3.4. Операторы языка БЕЙСИК

REM - вставка в текст комментариев. Все, что записано после оператора REM интерпретатором игнорируется.

DIM - оператор описания массивов. После оператора располагается список имен массивов с указанными максимальными индексами, то есть задается размер и размерность массива. Массив можно не описывать, если его размер не превышает 10.

DATA - описание в тексте данных, значения которых присваиваются переменным оператором READ. Операторы DATA могут быть размещены в любом месте программы, независимо от расположения операторов READ. Оператором DATA могут быть описаны как числовые, так и символьные данные. Если строка символов содержит разделители - запятую, двоеточие - ее нужно заключить в кавычки. Данные, описанные оператором DATA, образуют блок данных, значения которых можно последовательно считывать из блока оператором READ. После считывания каждого значения указатель в блоке смещается на следующую позицию.

READ - оператор чтения данных из блока, описанного оператором DATA.

RESTORE - перемещение указателя в блоке данных на первую позицию. Это позволяет многократно считывать данные из блока.

PRINT - предназначен для вывода на экран значений переменных, результатов вычислений и текстовых сообщений. Если оператор использован без операнда, то просто переводится строка. При вводе вместо слова PRINT можно писать знак "?". Операндов, стоящих вслед за оператором PRINT может быть несколько тогда их отделяют друг от друга разделителями "," и ";", причем при использовании первого следующий результат печатается в ближайшей справа позиции, кратной 14, а второго - вслед за предыдущим напечатанным результатом.

INPUT - оператор ввода данных с клавиатуры во время выполнения программы. Значения введенных данных (как числовых, так и символьных) присваиваются переменным, следующим за оператором. При выполнении оператора на экран выводится знак "?". В ответ нужно ввести данные, разделяя, если их несколько, запятой и нажать <BK>. Перед списком переменных может стоять строка символов, заключенная в кавычки, которая при выполнении оператора будет выведена перед знаком "?".

CUR X,Y - установка курсора в позицию на экране, левый нижний угол, который имеет координаты X,Y. Отсчет ведется от левого нижнего угла экрана. Дискретность сдвига позиции по вертикали — одна точка, по горизонтали — две. Диапазон изменения координат: X=0...191, Y=0...255. Если после оператора CUR сразу выполняется PRINT, то печатать начнется с позиции с координатами X, Y.

GOTO N — безусловный переход на строку с номером N.

GOSUB N — вызов подпрограммы, начинающейся со строки N. Для возврата из подпрограммы в точку вызова служит оператор RETURN. Вложенность подпрограммы ограничена только объемом свободной памяти.

ON X GOTO N1,N2,...NM - в зависимости от вычисления результата выражения X управление передается на строку, номер которой указан в списке, и состоит в нем по счету, равному целой части значения выражения X. То есть, если X=1, то будет переход на строку N1, X=2 - строку N2 и т. д. Если результат выражения больше или меньше количества строк в списке, то управление передается следующему оператору.

ON X GOSUB N1,N2,...NM - вызов подпрограммы по результату выражения X. Условия такие же, как в операторе ON X GOTO.

STOP - оператор останова программы. Продолжить выполнение программы можно директивой CONT.

FOR X=A1 to A2 STEP A3:.....:NEXT X - организация цикла по переменной X, которая будет принимать значение от 1 до A2 с шагом A3. Оператор NEXT - конец цикла. После оператора NEXT может быть указан список переменных, по которым в этом месте необходимо закончить цикл. Все операторы, стоящие между заголовками цикла FOR...TO...STEP и оператором NEXT называются телом цикла и выполняются в цикле, пока переменная X не достигнет конечного значения A2. A1 - первоначальное значение переменной. Каждое следующее значение X вычисляется по формуле $X=X+A3$, затем проверяется не превышает ли X значение A2 и все повторяется с новым значением X до тех пор, пока $X \leq A2$. Затем управление передается операторам следующим за оператором NEXT. Если шаг равен 1, то оператор STEP можно не писать.

IF X THEN Y - условный оператор. Работа его заключается в следующем: проверяется условие X, если (в переводе с английского IF — если) X — истинно то выполняются операторы Y, стоящие после слова THEN (тогда). В противном случае управление передается следующей строке программы. Выражение X считается ложным, если его значение равно 0, иначе оно считается истинным.

Выражение X может включать проверку различных условий с использованием операций отношения и логических операций. Если сразу после THEN стоит оператор GOTO, можно писать вместо THEN GOTO N просто THEN N.

CLEAR (X) очистка памяти от переменных. Всем числовым переменным присваивается значение 0, всем символьным - значение "пустая строка". Если параметр X указан, то изменяется размер буфера и памяти, отведенного для символьных переменных. В этом случае его длина будет равна X байт. Этот буфер предназначен для хранения символьных переменных, введенных с клавиатуры или образованных в результате слияния, а также для проведения различных операций с символьными переменными. Переменные описанные в тексте программы непосредственно, в буфер не попадают. При запуске интерпретатора размер буфера устанавливается равным 50. Максимально допустимая длина символьной переменной — 255 знаков, поэтому для операций с такими переменными буфер нужно расширить как минимум до 500 байт.

POKE X,Y позволяет записать в ячейку памяти с десятичным адресом, заданным выражением X, результат выражения Y. Y может быть равно 0...255, X=-32768...+32767. Адрес, старший (15-й) бит которого равен 1, в десятичном виде представляется отрицательным числом. То есть, если при переводе шестнадцатеричного числа X3X2X1X0 в десятичное YD по формуле

$$YD=X3*4096+X2*256+X1*16+X0$$

Получился результат больше 32767, то получить искомый адрес можно по формуле

$$Y=YD-65536.$$

Следующие операторы предназначены для формирования на экране графических изображений.

CLS N - очистка экрана цветом N. Если N=0, то цвет экрана не меняется, при N=1 или N=2 экран окрашивается соответственно в черный и белый цвета.

PLOT X,Y,N — вывод на экран точки с координатами X=0...283 и Y=0...255 цвета N. Цвет N означает те же цвета, что и в операторе CLS, но к ним добавляется еще значение N=3 - инверсный цвет. Это означает, что цвет точки в заданных координатах X,Y инвертируется — превращается из черного в белый и наоборот. Координатам X=0, Y=0 соответствует точка в левом нижнем углу экрана.

LINE X,Y - построение отрезка прямой линии из последней выведенной на экран точки операторами PLOT , LINE или DPL в точку с координатами X,Y.

Цвет рисуемой линии определяется последним выполненным оператором PLOT.

DPL DX,DY — построение отрезка из последней выведенной графическими операторами точки, координаты которой, например, равны X, Y в точку с координатами X+DX,Y+DY. Цвет определяется оператором PLOT.

Значения DX и DY должны находиться в пределах DX=-383...+383,DY=-255...+255

3.4.1. функции в языке БЕЙСИК.

В языке БЕЙСИК имеется ряд встроенных функций, которые позволяют значительно упростить написание некоторых программ. Название "встроенная" означает, что в интерпретаторе есть программа обработки данной функции. Существуют функции, работающие с числовыми и символьными данными, функции преобразования типов данных и функции обращения к машинным подпрограммам и ячейкам памяти.

В выражениях обращение к функции должно стоять в первой части, например,

$A = \text{COS}(X) + \text{SIN}(X/2)$.

Таким образом, для вызова функции достаточно указывать ее имя в выражении. В круглых скобках указывается аргумент. В качестве аргумента могут быть константы, имена переменных или выражения, которые, в свою очередь, могут содержать обращения к встроенным функциям.

функция	Результат работы
SQR(X)	корень квадратный из X
EXP(X)	экспоненциальная функция X
LOG(X)	натуральный логарифм от X
ABS(X)	абсолютная величина X
	$\text{ABS}(X) = \begin{cases} X, & \text{если } X > 0 \\ 0, & \text{если } X = 0 \\ -X, & \text{если } X < 0 \end{cases}$
SGN(X)	знак X
	$\text{SGN}(X) = \begin{cases} 1, & \text{если } X > 0 \\ 0, & \text{если } X = 0 \\ -1, & \text{если } X < 0 \end{cases}$
SIN(X)	синус от X (X - в радианах)
COS(X)	косинус от X (X - в радианах)
TAN(X)	тангенс от X (X - в радианах)
ATN(X)	арктангенс от X, результат в радианах
INT(X)	целая часть от X
RND(X)	случайное число в диапазоне от 0 до 1

RND(X) — функция генерации случайного числа. Аргумент функции всегда равен 1. Результат работы - случайное число в диапазоне от 0 до 1.

Если необходимо получить число в другом диапазоне, например, от 10 до 100, достаточно написать:

$A = \text{RND}(1) * 91 + 10$

а чтобы сделать его целым — $A = \text{INT}(\text{RND}(1) * 91) + 10$

Рассмотрим встроенные функции для работы с символьной информацией.

LEN(X\$) - результат ее работы равен количеству символов в переменной X\$. Например, если X\$="АВТОМАТ", то LEN(X\$)=7.

LEFT\$(X\$,N) позволяет выделить левую часть переменной X\$ длиной N символов. Например, LEFT\$(X\$, 4) равно "АВТО".

RIGHT\$(X\$,N) выделяет правую часть переменной X\$ длиной N символов. Например, RIGHT\$(X\$,3) равно "MAT".

MID\$(X\$,P,N) "вырезает" из переменной X\$ строку символов длиной N, начиная с позиции P в переменной. Например, значение MID\$(X\$,4,2) будет равно "OM".

В программах часто возникает необходимость перевода данных из числового вида в символьный и наоборот. Этой цели служат две специальные функции.

STR\$(X) - преобразует результат арифметического выражения X в строку символов, представляющими этот результат.

VAL(X\$) - преобразование данных из символьного вида в числовой, начиная с крайнего левого символа. Как только встречается недопустимый символ (не цифра и не знак числа), преобразование прекращается.

ASC(X\$) - код первого символа X\$. Например, ASC("F") равно 70.

CHR\$(X) - получение символа, код которого равен X. Аргумент не должен превышать 255. Например, если X=70, то CHR\$(X) равно "F".

Для управления форматом печати на экране служат следующие три функции.

POS(I) - номер позиции последнего выведенного на экран символа.

SPC(X) — используется совместно с PRINT. Результат работы — печать X пробелов.

TAB(X) - горизонтальная табуляция. Позволяет переместить курсор в позицию X в строке.

Функция FRE(0) возвращает своим значением число байт свободной памяти для хранения программ и данных.

PEEK(X) - содержимое ячейки памяти с адресом X.

INP(A) - возвращает своим значением код нажатой клавиши. При обращении к этой функции на экране появляется курсор, и программа ждет нажатия клавиши.

Функция USR(X) вызывает подпрограмму в машинных кодах, начинающуюся с адреса X. Подпрограмма должна обязательно заканчиваться командой RET.

Своим значением функция USR(X) возвращает содержимое аккумулятора (регистра A процессора) в момент возврата из подпрограммы.

Пример использования функций USR и POKE :

```
10 X=-16016:REM адрес подпрограммы генерации звука
20 FS=-28687:REM адрес ячейки-частота звука
30 DL=-28686:REM ячейка-длительность звучания
40 POKE DL,20
50 FOR I=10 TO 100 STEP 5
60 POKE FS,I
70 A=USR(X)
80 NEXT
```

Кроме встроенных функций, в интерпретаторе имеется возможность определять новые, назначение которых программируется пользователем. Определить такую функцию можно в любом месте программы оператором DEF. Имена таких Функций должны начинаться с префикса FN, за которым могут следовать один или два символа имени.

```
10 DEF FNLG(X)=LOG(X)/LOG(10):REM десятичный логарифм
20 A=FNLG(100):B=FNLG(A*10)
```

Параметр X является формальным, необходимым для обозначения функциональной зависимости.

3.4.2. Сообщение об ошибках

Интерпретатор языка БЕЙСИК позволяет в процессе выполнения программы обнаруживать и анализировать ошибки. Сообщение выводится на экран в виде

XX ошибка в N

N — номер строки программы, где была обнаружена ошибка, XX — код ошибки:

Ошибка 01 - в программе встретился оператор NEXT, для которого не был выполнен оператор FOR.

Ошибка 02 - неверный синтаксис

Ошибка 03 - встретился оператор RETURN без предварительного выполнения GOSUB.

Ошибка 04 - не хватает данных для оператора READ.

Ошибка 05 - аргумент функции не соответствует области определения.

Ошибка 06 - переполнение при выполнении арифметических операций.

Результат любой операции не может быть меньше -1.7E38 или больше 1.7E38.

Ошибка 07 - недостаточен объем памяти.

Ошибка 08 - нет строки с данным номером. Возникает при выполнении операторов перехода.

Ошибка 09 - индекс не соответствует размеру массива.

Ошибка 10 - повторное выполнение операторов DIM или DEF.

Ошибка 11 - деление на 0.

Ошибка 12 - попытка выполнить INPUT или DEF в непосредственном режиме.

Ошибка 13 - несоответствие типов данных.

Ошибка 14 - переполнение символьного буфера.

Ошибка 15 - длина символьной переменной превышает 255.

Ошибка 16 - выражение, содержащее символьные переменные слишком сложно для интерпретатора.

Ошибка 17 - невозможность продолжения программы по директиве CONT.

Ошибка 18 - обращение к функции, не описанной оператором DEF.

Кроме описанных, интерпретатор выдает еще три сообщения об ошибке при выполнении оператора INPUT;

?ПОВТОРИТЕ ВВОД - означает ошибку в наборе данных - вместо числа введена строка символов.

?ЛИШНИЕ ДАННЫЕ - данных набрано больше, чем переменных в операторе INPUT. Лишние данные просто игнорируются.

?? - данных наоборот меньше, чем задано переменных.

Необходимо ввести недостающие данные.

Руководство оператора

АННОТАЦИЯ

В данном руководстве приведено описание действий оператора по загрузке и запуску программ, входящих в комплект программного обеспечения радиоконструктора "Бытовое электронно-вычислительное устройство ЛИК" (в дальнейшем радиоконструктор ЛИК).

1. Назначение программ

МОНИТОР-1М - это ядро системы программного обеспечения радиоконструктора ЛИК. Программа МОНИТОР-1М является основной управляющей программой и представляет простые и удобные средства для работы непосредственно в машинных кодах.

Программа РЕДАКТОР служит для внесения изменений в символьные записи.

МОНИТОР и РЕДАКТОР предоставляют возможность пользователю проверять и/или изменять содержимое памяти, загружать программы или данные с магнитофонной ленты и начинать выполнение программы, а также управлять периферийными устройствами (бытовой телевизионный приемник, бытовой магнитофон).

Интерпретатор языка БЕЙСИК обеспечивает подготовку и выполнение программ, написанных на алгоритмическом языке БЕЙСИК. Язык программирования БЕЙСИК ориентирован на выполнение программ в режиме диалога.

Связь оператора с МОНИТОРОМ, РЕДАКТОРОМ и интерпретатором языка БЕЙСИК осуществляется посредством нажатия клавиш радиоконструктора ЛИК.

Игровые программы способствуют существенному повышению сообразительности, увеличению реакции, а также предназначен для формирования навыков самостоятельного исследования, интуиции и развития логического мышления.

2. Условия выполнения программ

Для работы комплекта программ радиоконструктора ЛИК необходим следующий состав оборудования:

- бытовой кассетный магнитофон;
- бытовой телевизионный приемник;
- радиоконструктор ЛИК;
- источник питания;
- соединительные жгуты.

3. Выполнение программ

Программное обеспечение радиоконструктора ЛИК состоит из МОНИТОРА, интерпретатора БЕЙСИК, РЕДАКТОРА, каталога, тестовых и игровых программ.

3.1. МОНИТОР

Вызов МОНИТОРА, находящегося в ПЗУ, можно произвести нажатием клавиш <СБР>, <STOP>, <BK>.

На экране телевизора под надписью *RUN "COM:" должно появиться сообщение "МОНИТОР - 1М" с курсором в виде сплошного квадратика и стрелки под надписью. МОНИТОР готов к приему директив.

Запись МОНИТОРА в память радиоконструктора ЛИК с магнитофона кассеты:

установить кассету с программой "МОНИТОР" в кассетоприемник магнитофона;

перемотать ленту к началу записи;

установить счетчик магнитофона в ноль;

нажать клавишу <СБР>;

пустить ход ленты в режиме чтения;

по началу сигнала на ленте, обнаруживаемому по звуку или индикатору сигнала, отпустить клавишу <СБР> и ждать окончания сигнала на ленте.

При нормальной загрузке, после окончания сигнала на магнитофонной ленте, на экране под надписью *RUN "COM:" должна появиться надпись *МОНИТОР - 1М* с курсором в виде сплошного квадратика и стрелки под надписью. МОНИТОР готов к приему директив.

Если МОНИТОР был загружен с магнитофона, он повторно вызывается последовательностью <СБР> , <STOP> , <ПС> .

3.1.1. Директивы МОНИТОРА

D ADR - распечатка и редактирование содержимого памяти в шестнадцатеричном и символьном виде с адреса ADR. Печатается таблица размером в 129 байт, начиная с ADR. После печати таблицы выводится приглашение ==> , означающее ожидание ввода команды редактирования:

<BK> - распечатка следующего блока

[←] - редактирование шестнадцатеричной таблицы ,

[→] - редактирование символьной таблицы

<ПВТ> - повторение директивы

<BK> - выход из редактирования

<пробел> - выход в МОНИТОР

Пример. Распечатать содержимое памяти с адреса 0000H

Действие оператора: D0<BK>

Сообщение МОНИТОРА:

	0	1	2	3	4	5	6	7	01234567
	8	9	A	B	C	D	E	F	89ABCDEF
0000	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
0008	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
0010	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
0018	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
0020	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
0028	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY
	...	...	...	...	...	...	...	...	
0078	XX	XX	XX	XX	XX	XX	XX	XX	YYYYYYYY

где: 0000, 0008, 0010,...,0078 - адреса памяти;

0 1 2 3 4 5 6 7

8 9 A B C D E F - поправка к адресу;

XX XX XX XX XX XX XX - содержимое памяти в шестнадцатеричном виде;

YYYYYY - содержимое памяти в символьном виде.

Примечание. При вводе адреса незначащие нули можно опускать.

Пример. Отредактировать содержимое памяти: по адресу 0000H записать F1,

а по адресу 0001H записать CC. После чего выйти в МОНИТОР.

Действие оператора:

D0<BK>

[←]

F1CC

<BK>

нажать <пробел>

сообщение МОНИТОРА:

после первой команды МОНИТОР выводит на экран таблицу, после ввода [←] курсор занимает исходное место в верхнем левом углу шестнадцатеричной таблицы. После ввода F1CC содержимое адресов 000H и 001H меняют свое значение на F1 и CC соответственно. Одновременно меняется по этим же адресам значение символьной информации. После нажатия <BK> происходит выход из редактирования. После нажатия <пробела> происходит завершение редактирования.

M ADR,XX - запись кода XX в ячейку с адресом ADR.

Пример. Записать код 31H в ячейку 8B40H.

Действие оператора:

M8B40,31<BK>

Сообщение МОНИТОРА:

* МОНИТОР - 1М *

===>

F ADR1,ADR2,XX - заполнение области памяти с ADR1 по ADR2 кодом XX.

Пример. Заполнить область памяти с адреса 000H по адрес 1FFFH кодом 76H

Действие оператора;

F0,1FFF,76<BK>

Сообщение МОНИТОРА:

* МОНИТОР - 1М *

===>

T ADR 1 , ADR2 ,ADR3 - пересылка содержимого области памяти ADR1 - ADR2 по адресу ADR3. Копировать можно и перекрывающиеся области.

Пример. Переслать содержимое памяти с области 0000H - 01EFH в области 0010H - 01FFH.

Действие оператора:

T0,1EF,10<BK>

Сообщение МОНИТОРА:

* МОНИТОР - 1М *

===>

S ADR1,ADR2,XX - поиск в области памяти ADR1 - ADR2 кода XX.

При этом выводится адрес ячейки, содержащий искомый код и содержимое трех ячеек, начиная с найденного адреса.

Пример. Произвести поиск кода CDH в области памяти с адреса 0000H по адрес 1FFFFH.

Действие оператора:

S,1FFF,CD ,<BK>

Сообщение МОНИТОРА:

015E CD 37 C0

03CA CD A1 04

13E5 CD EF 10

* МОНИТОР - 1M *

====>

Монитор печатает все адреса (015E,03CA,13E5), по которым он нашел код CD. Дополнительно печатается содержимое ещё двух ячеек, которые следуют за ячейкой с кодом CD.

C ADR1,ADR2,ADR3 - сравнение области памяти ADR1 - ADR2 с областью, начинающейся с адреса ADR3. На экран выводятся адреса и содержимое несовпадающих ячеек.

Пример. Область памяти с 0000H по 1FFFFH адреса заполнить кодом 34H. В ячейку 035EH записать 2FH. Сравнить область, памяти 0000H - 0FFFFH с областью памяти 1000 - 1FFF.

Действие оператора:

F0,1FFF,34 <BK>

M35E,2F <BK>

C0,FFF,1000 <BK>

Сообщение МОНИТОРА:

035E 2F 135E 34

* MONITOR-1M *

====>

2FH - код записан по адресу 035EH

34H - код записан по адресу 135EH

I=ИМЯ,ADR1,ADR2,XX - ввод программы с магнитофона. XX – константа записи (необязательна). Имена найденных на ленте файлов, имена которых не совпадают с указанным, будут печататься на экране. Знак <== после распечатанного имени означает, что файл найден и загружается. Если адреса не указывать, то файл будет загружен по адресам, указанным на ленте, иначе по адресам ADR1+4 - ADR2+4, а в ячейках ADR1+3 будут находиться адреса файла, записанные на ленте.

Если файл есть в каталоге, то на экране будет выведена строка подсказка из каталога, содержащая данные о файле, в том числе номер по счетчику, где находится данный Файл.

По окончании загрузки на экран выводятся: начальный и конечный адреса программы, контрольная сумма и сумма сформированная в результате, загрузки. Если контрольные суммы не совпадают, то будет выведено сообщение об ошибке.

Пример. Ввести с магнитофона каталог поставляемых программ.

Действия оператора:

Перемотать магнитофонную ленту на начало 2-го файла.

ввести команду: I=КАТАЛОГ .A

включить магнитофон в режиме чтения

нажать клавишу <BK>

ждать окончания ввода информации с магнитофона

выключить магнитофон.

Сообщение МОНИТОРА:

```
КАТАЛОГ. A <==  
8B40  
8F3F  
2DC9  
2DC9  
* МОНИТОР -1M *  
==>
```

Примечание:

КАТАЛОГ. A <== - это означает, что программа найдена и загружается.

8B40 - начальный адрес.

8F3F - конечный адрес.

2CD9 - контрольная сумма.

2CD9 - контрольная сумма после загрузки.

G=ИМЯ - загрузка программы с магнитофонной ленты и ее запуск с именем ИМЯ. При вводе с ошибкой запуск программы не производится.

Пример. Ввести в ОЗУ ЛИКа с магнитофонной ленты каталог поставляемых программ и распечатать его.

Действия оператора:

Перемотать магнитофонную ленту на начало 2-го файла (сторона А).

ввести команду: G=КАТАЛОГ. А

включить магнитофон в режиме чтение

нажать клавишу <ВК>

ждать окончания ввода информации с магнитофона

выключить магнитофон

Сообщение МОНИТОРА:

КАТАЛОГ. А <==

8B40

8F3F

2DC9

2DC9

0001	КАТАЛОГ. А	8B40 - 8F3F
0002	КАТАЛОГ0	8B40 - 8F3F
0003	РЕДАКТОР	0000 - 0FFF
0004	BASIC	0000 - 1DBF
0005	ТЕСТФ	0000 - 004D
0006	ТЕСТОЗУ	0003 - 05C4
0007	ТЕСТПЗУ	0000 - 01E1
0008	ТЕСТМАГ	0000 - 01D1
0009	МОНИТОР-SP	8000 - 882F
0010	FREE	
*МОНИТОР - 1M *		
==>		

На экране распечатан каталог поставляемых программ, где в первой колонке указан номер файла. Во второй колонке указаны названия программ, кроме первой программы МОНИТОР, которая не имеет названия на ленте и загружается по нажатию клавиши <СБР>. В третьей колонке указаны адреса размещения данных программ в оперативной памяти ЛИКа после их загрузки.

О=ИМЯ,ADR1,ADR2,XX - вывод области ADR1 - ADR2 на магнитофон, XX - константа записи (необязательна). Имя файла может состоять не более, чем из 15 символов. Если в памяти находится КАТАЛОГ, необходимо предварительно воспользоваться директивой **Р**. Если файл уже есть в КАТАЛОГЕ и выводится повторно, то адреса (если они остались прежними) можно не указывать.

После нажатия на клавишу <BK> на экран будет выведена контрольная сумма.

Пример. Вывести на магнитофонную ленту КАТАЛОГ, под именем FILES, расположенный по адресам 8B40H - 8F3FH.

Действие оператора:

вставить магнитофонную ленту в кассетоприемник и перемотать ее на начало свободного файла.

ввести команду: O=FILES,8B40,8F3F

включить магнитофон в режим записи

нажать клавишу <BK>

ждать окончания вывода информации на магнитофонную ленту.

выключить магнитофон.

Сообщение МОНИТОРА:

F5A1

* МОНИТОР -1M *

==>

Примечание: F5A1 - контрольная-сумма

V - сравнение содержимого ОЗУ с записью на ленте. Используется для контроля правильности только что записанной на ленту программы, для чего после записи перематывают ленту к началу записанной программы, набирают директиву V и по началу звука нажимают клавишу <BK>.

Если по окончании проверки будет выведена надпись "ОШИБКА", значит запись необходимо повторить.

X - вывод на экран констант ввода-вывода.

Пример. Вывести константы ввода-вывода на экран. Действия оператора:

X <BK>

Сообщение МОНИТОРА:

3C28

* МОНИТОР - 1M *

==>

Примечание: В сообщении 3C28 - 3CH является константой ввода, 28H является константой вывода.

J ADR - запуск программы с адреса ADR, ранее загруженной в ОЗУ.

K ADR1,ADR2 - подсчет контрольной суммы блока ADR1 - ADR2.

Пример. Подсчитать контрольную сумму информации, что находится с адреса 0000H по адрес 01FFH.

Действие оператора:

K0,1FF <BK>

Сообщение МОНИТОРА:

FF00

* МОНИТОР - 1M *

==>

B - загрузка в ОЗУ и запуск БЕЙСИКА.

Пример. Загрузить БЕЙСИК в оперативную память радиоконструктора ЛИК и подготовить его к приему информации.

Действие оператора:

B <BK>

Сообщение МОНИТОРА:

* BASIC ЛИК V2 *

OK

Примечание: При отсутствии интерпретатора БЕЙСИК в ПЗУ загрузку его произвести с миникассеты.

Выйти из программы БЕЙСИК в МОНИТОР можно нажав клавиши:

а) <СБР>, <STOP>, <BK> - для МОНИТОРА в ПЗУ

б) <СБР>, <STOP>, <ПС> - для МОНИТОРА, загруженного с магнитофонной ленты.

A ADR - дизассемблирование программы с адреса ADR.

Пример. Если в оперативной памяти с адреса 0000H по адрес 01FFH находится программа в машинных кодах, то произвести ее дизассемблирование можно следующим образом:

Действие оператора:

A0 <BK>

Сообщение МОНИТОРА:

0000 31 FF 3F 1.. LXI SP,7FFF

0003 21 00 C8 !.. LXI H ,0000

0006 22 E1 8F "... SHLD 8FE1

где в первой колонке указан адрес, во 2-й, 3-й и 4-й - шестнадцатеричный код, в 5-й колонке - метка команды, в 6-й - команда ассемблера.

P - распечатка каталога файлов. Если каталог в ОЗУ не загружен, то при попытке выполнить директиву P будет выведена сообщение об ошибке. Если загружен "чистый" каталог, то перед началом работы необходимо выполнить директивы M8B40,31<BK>. "Чистый" каталог находится на магнитофонной ленте, поставляемой потребителю - файл 3, имя файла КАТАЛОГ 0.

Он используется для записи и хранения перечня программ потребителя, их начальный и конечный адреса размещения в ОЗУ после загрузки и номера счетчика магнитофонной ленты.

Пример. Загрузить "чистый" каталог и подготовить его к приему информации.

Действие оператора:

Вставить поставляемую магнитофонную ленту в кассетоприемник и перемотать ее на начало 3-го файла (сторона А)
ввести команду: I=КАТАЛОГ 0
включить магнитофон в режиме чтения
нажать клавишу <BK>
ждать окончания ввода информации с магнитофона
выключить магнитофон

Сообщение МОНИТОРА:

```
КАТАЛОГ 0 <==  
8B40  
8F3F  
8AA1  
8AA1  
* МОНИТОР - 1М *  
===>
```

Действие оператора:

M8B40,31<BK>

Сообщение МОНИТОРА:

```
* МОНИТОР - 1М *.  
===>
```

Пример. Распечатать каталог файлов. Действие оператора:

P<BK>

Сообщение МОНИТОРА:

```
0010 FILES      8B40 - 8F3F  
0030 FREE
```

P1,N - ввод номера N счетчика ленты, куда будет выведена очередная программа - директива, выполняется перед записью программы на магнитофон.

Пример. Записать область памяти с адреса 0000H по адрес 01FFH на магнитофонную ленту. Счетчик ленты - 30. Название выводимой программы - PROGRAM 1.

Действие оператора:

P1,30<BK>

Сообщение МОНИТОРА:

```
* МОНИТОР - 1М *  
===>
```

Действие оператора:

Вставить магнитофонную ленту в кассетоприемник и перемотать ее на начало.
Сбросить счетчик ленты в ноль.
Выключить перемотку ленты вправо (>>) и наблюдать за счетчиком ленты.
Выключить магнитофон при показании счетчика 30.
Нажать на магнитофоне клавишу "ожидание" и "запись".
Ввести директиву O=PROGRAM1,0,1FF
Отжать магнитофонную клавишу "ожидание".
Нажать <BK>.
Ждать окончания вывода информации на ленту.

Выключить магнитофон.

Распечатать состояние КАТАЛОГ 0 можно следующим образом.

Действие оператора:

P<BK>

Сообщение МОНИТОРА:

```
0010 FILES          8B40 - 8F3F
0030 PROGRAM1       0000 - 01FF
0030 FREE
```

P2,N - ввод номера и счетчика ленты, куда можно (будет потом записать очередную программу). Этот номер означает свободное место на кассете, и при распечатке каталога директивой P будет выведен в конце: N - FREE.

Пример. Изменить признак начала следующей записи.

В предыдущем примере мы записали на ленту программу PROGRAM1. Счетчик магнитофонной ленты в конце записи был равен 80. Значит следующую информацию будем записывать по счетчику равному 90.

Действие оператора:

P2,90<BK>

Сообщение МОНИТОРА:

```
* МОНИТОР - 1M "
===>
```

Примечание. Если директивой P <BK>, распечатаем каталог, то он будет выглядеть следующим образом:

```
0010 FILES          8BA0 - 8F3F
0030 PROGRAM1       0000 - 01FF
0090 FREE
```

P3,N - удаление строки каталога с номером N.

Пример. Удалить в каталоге строку с номером 30.

Действие оператора:

P3,30 <BK>

Сообщение МОНИТОРА:

```
* МОНИТОР - 1M *
===>
```

P4 - подготовка для записи программы на свободное место (с номером по счетчику FREE)

Пример. Записать на свободное место магнитофонной ленты программу PROGRAM2 (с номером по счетчику FREE).

Действие оператора:

P4<BK>

Сообщение МОНИТОРА:

```
* МОНИТОР - 1M *
===>
```

Действие оператора:

O=PROGRAM2,200,3FF

Установить магнитофонную ленту на начало свободного файла и включить в режим записи.

Нажать клавишу <BK>

Сообщение МОНИТОРА:

7101 - контрольная сумма данной программы

Если задать директиву P<BK>, то на экране появится содержимое каталога:

```
0010    FILES      8B40 - 8F3F
0090    PROGRAM2   0200 - 03FF
0090    FREE
```

Выполнение любой директивы, кроме вывода на ленту, можно прервать нажатием клавиши <STOP>. Продолжить выполнение можно нажатием любой клавиши. Окончательно прерывается работа одновременным нажатием клавиши <STOP> и <ПРОБЕЛ>. Прервать ввод с магнитофона можно нажатием последовательности <STOP> и <ПС>.

При вводе директив нулевые параметры можно не указывать, но запятая обязательна. Незначащие нули параметров, можно не писать. Вводимую строку можно редактировать клавишами [←] и [→]. Крайний справа символ удаляется клавишей <ЗБ>. Директива выполняется после нажатия клавиши <BK> независимо от расположения при этом курсора.

Предыдущую директиву можно повторить, если вместо ввода новой нажать <BK>.

3.2. Редактор текстов и АССЕМБЛЕР (пакет МИКРОН).

Пакет "МИКРОН" состоит из редактора текстов и АССЕМБЛЕРА, он поставляется потребителю на магнитофонной ленте, где записан 4-м файлом.

3.2.1. Загрузка редактора в ОЗУ.

Действие оператора:

Вставить поставляемую миникассету в кассетоприемник.

Ввести команду: I= редактор.

Включить магнитофон в режиме "чтение".

Нажать клавишу <ВК>.

Ждать окончания ввода информации с магнитофона.

Выключить магнитофон.

Сообщение МОНИТОРА:

```
Редактор <==  
0000  
0FFF  
C6FA  
C6FA  
* МОНИТОР - IM *  
===>
```

3.2.2. Запуск редактора

Запуск редактора производится директивой J МОНИТОРА.

Действие оператора:

J<ВК>

Сообщение:

```
ED.*МИКРОН*  
*
```

В этом случае редактор готов к приему директив.

3.2.3. Директивы редактора.

3.2.3.1. Режим приема директив

<ВК> - очистка текстового буфера

<ПС> - выход в монитор

< > - выход в АССЕМБЛЕР

<О> - вывод текста на магнитофон.

При нажатии клавиши <О> на экране появляется: имя =

Необходимо включить магнитофон в режиме "запись". Ввести имя выводимого файла и нажать <ВК>.

<I> - ввод текста ленты. По запросу редактора, имя =

Ввести имя файла и нажать клавишу <ВК>.

Имена файлов, несовпадающих с требуемым, будут выведены на экран.

После загрузки искомого файла редактор перейдет в режим ввода директив.

<V> - сравнение текста на ленте с содержанием ОЗУ.

Перемотать ленту на начало файла. По началу сигнала на ленте, обнаруживаемому по звуку или индикатору сигнала, нажать клавишу <V>.

- <S>** - поиск группы символов.
После нажатия клавиши на экран выводится знак
" = ", после чего нужно ввести искомую последовательность и нажать <BK>. Если такая последовательность в тексте обнаружена, то на экран будет выведен фрагмент текста, первая строка которого содержит данную последовательность. Продолжить поиск можно нажатием клавиши "S" в режиме редактирования.
- <F>** - определение верхней границы текста.
После нажатия клавиши <F> на экране появляется четырехзначное шестнадцатеричное число, определяющее адрес верхней границы текста.
<↑> - распечатка и редактирование первого фрагмента текста.
<↓> - распечатка следующего фрагмента текста.
<→> - распечатка текущего фрагмента текста.
- <A>** - переход в режим ввода строк.
При нажатии клавиши <A> на экране появляется >

3.2.3.2. Режим ввода строк

- <ПС>** - начало ввода.
При вводе этой команды на экране появляется >?
после чего можно ввести строку текста. По окончании ввода строки нажать <BK>.
- <↵>** - переход в режим приема директив.

3.2.3.3. Редактирование текста

- <↓>, <↑> - перемещение курсора к редактируемой строке.
<→> - вход в редактирование строки.
<ПС> - табуляция.
<↵> - включение/выключение режима сдвижки/раздвижки строки.
При включенном режиме клавиши <←>, <→> соответственно сдвигают и раздвигают строку.
<BK> - ввод строки в память. При этом положение курсора не имеет значения.
<↵> - выход в режим приема директив.
<. > - вставка строк в текст (переход в режим ввода строк).

3.2.3.4. Удаление и перенос строк

- D** - метка первой строки.
- E** - метка последней строки, после чего на вопрос "Delete or transmit?" ответить "D", если группу строк необходимо удалить, или "T", если запомнить для последующей вставки.
- <T>** - вставка в текст отмеченного фрагмента.

3.2.3.5. Директивы ассемблера

- <1>** - трансляция программы с выдачей листинга на экран дисплея.
<2> - трансляция программы с выдачей на экран дисплея меток программы и их адресов.
<3> - трансляция программы.
В конце выполнения каждой из директив выдается количество ошибок и конечный адрес программы.

3.3. Тест функционирования

Тест функционирования используется для проверки функционирования и наладки радиоконструктора ЛИК.

Тест функционирования записан на стороне А поставляемой кассеты седьмым файлом-Загружается монитором: I=ТЕСТф. Запуск теста производится при помощи директивы J. Наблюдайте выполнение теста функционирования, который производит следующие операции в циклическом режиме:

- подает сигнал на динамик;
- подает сигнал записи на магнитофон (при подключенном магнитофоне наблюдайте индикацию импульсных сигналов на индикаторе уровня записи);
- записывает нули в ОЗУ;
- записывает нули в экранное ОЗУ (при подключенном телевизионном приемнике наблюдайте окраску экрана в темный цвет);
- записывает "1" во все ячейки ОЗУ;
- записывает "1" во все ячейки экранного ОЗУ (при подключенном телевизионном приемнике наблюдайте окраску экране в светлый цвет).

Прохождение всех сигналов вы можете проверить по схеме электрической принципиальной Рис.2 4-075-005 РЗ.

3.4. Тест ОЗУ

Тест ОЗУ представляет возможность проверять работоспособность микросхем КР565РУ6 на правильность записи в них информации, проверять быстродействие цикла записи/считывания информации, а также производить обнаружение адресных ошибок.

Тест ОЗУ записан на стороне А поставляемой кассеты восьмым файлом. Загружается монитором: I=ТЕСТОЗУ. Запуск теста: J,3. Наблюдайте выполнение теста ОЗУ, который производит в циклическом режиме следующие действия:

- 1) подает один звуковой сигнал на динамик, который обозначает начало работы теста ОЗУ;
- 2) записывает и проверяет правильность записи информации в микросхемы КР565РУ6.

В случае обнаружения ошибки тест ОЗУ подает на динамик последовательность звуковых сигналов- Количество сигналов высокого тона обозначает линейку, а количество сигналов низкого тона обозначает номер микросхемы в данной линейке.

3) проверяет наличие адресных ошибок. В случае обнаружения ошибки тест ОЗУ подает на динамик последовательность звуковых сигналов высокого тона.

4) проверяет время цикла записи информации в микросхему и считывания ее с микросхемы. Если время цикла записи/считывания превышает допустимое значение, то тест ОЗУ подает на динамик последовательность звуковых сигналов. Количество сигналов высокого тона обозначает линейку, а количество сигналов низкого тона обозначает порядковый номер микросхемы в линейке.

При обнаружении любой ошибки тест ОЗУ выдает сообщение об ошибке на динамик и повторно проверяет бракованную микросхему. В том случае если ошибка не обнаружена тест продолжает свою работу, в противном случае тест опять выдает сообщение на динамик.

Примечание:

Микросхемы DS17—DS24 соответствуют 1—й линейке, DS9—DS10 соответствуют 2-й линейке, DS1-DS8 соответствуют 3-й линейке (см. схему расположения элементов 4.075.005 РЗ). Номер микросхемы в линейке считывается слева направо.

3.5. Тест ПЗУ

Тест ПЗУ проверяет правильность считывания с ПЗУ. Тест ПЗУ работает в диалоговом режиме.

Тест ПЗУ вводится в ОЗУ с миникассеты по директиве I=ТЕСТПЗУ.

Запуск теста происходит по директиве J. На экране появляется сообщение:

ТЕСТ ПЗУ

При появлении на экране запроса:

Введите номер микросхемы:

Необходимо указать номер проверяемой микросхемы ПЗУ (нумерация начинается с 1, что соответствует микросхеме DS1 на схеме расположения элементов в документе 4.075.005 РЭ).

При исправной работе микросхемы выводится сообщение:

Ошибок нет

При неисправности выводится сообщение:

Ошибка ! ! !

3.6. Тест магнитофона

Тест магнитофона проверяет программы записи на магнитофон и чтения с магнитофона, а также проверяет магнитофон в режимах записи и чтения с радиоконструктором ЛИК. Тест магнитофона работает в диалоговом режиме.

Тест магнитофона вводится в ОЗУ с миникассеты по директиве I=ТЕСТМАГ. Запуск теста производится по директиве J.

На экране появляется надпись:

Тест магнитофона

При появлении на экране указания:

Включите магнитофон на запись, нажмите <ВК>

Необходимо вставить в магнитофон чистую кассету и включить его в режим "Запись", после чего нажать <ВК>.

При появлении на экране указания:

Перемотайте ленту на начало, включите магнитофон на воспроизведение, нажмите <ВК>

Необходимо перемотать ленту на начало записи, включить магнитофон на воспроизведение и нажать "ВК".

При нормальном завершении работы теста на экран выводится сообщение:

Конец работы теста

Если во время считывания будет обнаружена ошибка, то на экран выводится сообщение:

Ошибка !!!

3.7. МОНИТОР-SP

Программа Монитор-SP представляет собой монитор радиоконструктора "Специалист" (см. журнал "Моделист-конструктор" 1987г.), адаптированный к основным подпрограммам загрузчика радиоконструктора ЛИК.

Загрузка программы осуществляется по директиве: I=МОНИТОР-SP <BK>. Программа размещается в ОЗУ по адресам 8000H-882FH, поэтому работать эта программа может, если МОНИТОР—1М находится в ПЗУ.

После загрузки программы "МОНИТОР-SP" вы можете загрузить игровую программу написанную для радиоконструктора "Специалист". Для примера на кассете поставляется игра "Буря".

Игра загружается при помощи директивы I МОНИТОР-1М и запускается директивой J.

Если игровая программа записана в формате радиоконструктора "Специалист", то она загружается при помощи директив монитора радиоконструктора "Специалист" (см. журнал "Моделист-конструктор") после чего ее необходимо вызвать директивой J8000.

3.8. Игровая программа "РЕВЕРСИ"

Правила игры.

Игра "Реверси" (от латинского to reverse - обращать) завоевала широкую популярность во многих странах. Ее привлекательность объясняется простыми правилами и удивительной динамичностью.

Игра способствует развитию логического мышления.

Вашим соперником в игре будет компьютер. Загрузка программы в ОЗУ осуществляется при помощи директивы программы МОНИТОР: I=РЕВЕРСИ. Запуск по директиве J.

После демонстрации рекламной заставки компьютер предлагает нажать "пробел". На запрос компьютера выбрать цвет, клавишами <↑> или <↓> необходимо заставить мигать фишку на экране того цвета, которым вы решили играть. Выбрав цвет, нажмите <пробел>. Аналогично выбираем уровень сложности игры.

Передвижение фишек по полю осуществляется клавишами <←>, <↑>, <→> и <↓>.

Фиксация хода осуществляется нажатием клавиши <пробел>.

Компьютер ответный ход делает автоматически.

Каждый ваш ход и ход компьютера сопровождается музыкальным фрагментом.

Если вы сделали неверный ход, компьютер возвращает фишку, в исходное положение (верхний левый угол поля).

Ознакомившись с методическими указаниями по игре "РЕВЕРСИ" с радиоконструктором ЛИК можно приступить и к самой игре.

Начинают черные. Ходят по очереди, выставляя фишки на Свободные клетки поля обязательно рядом с фишкой противоположного цвета по горизонтали, вертикали или диагонали так, чтобы "замкнуть" эту фишку.

На рис.1 крестиками отмечены все возможные варианты первого хода черных.

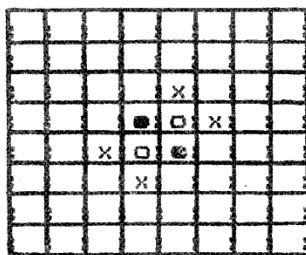
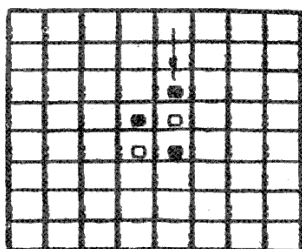
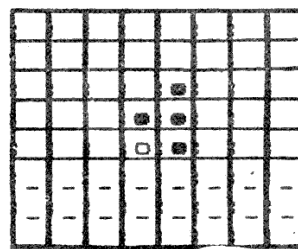


Рис. 1

Замкнутая ходом фишка, т.е. фишка, расположенная между фишками противоположного цвета автоматически меняет цвет. (Рис. 2).



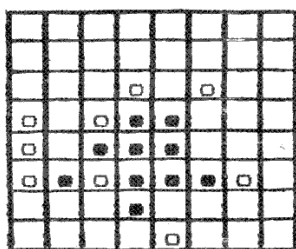
Ход



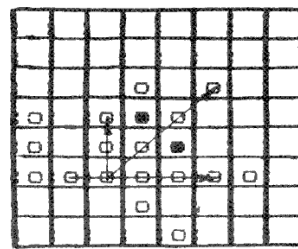
Результат

Рис. 2

В дальнейшем, ставя фишки на свободное поле, замыкание может происходить одновременно по нескольким линиям (рис.3). В этом случае меняют цвет все цепочки фишек соперника.



Ход



Результат

Рис.3

Если один из играющих не может сделать ход по правилам, он его пропускает, а право продолжать игру получает соперник (Рис.4 у белых хода нет).

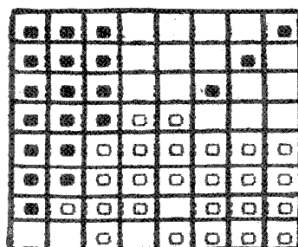


Рис. 4

Игра продолжается до тех пор, пока ни один из играющих не сможет сделать правильный ход, либо все игровое поле не будет заполнено фишками.

Побеждает тот, у кого на игровом поле будет больше своего цвета.

НЕСКОЛЬКО СОВЕТОВ ПО ТАКТИКЕ ИГРЫ.

1. фишку соперника, расположенную в центре поля, захватить проще, чем на краю, поэтому надо стремиться занимать края поля и препятствовать в этом сопернику. Особенно выгоден захват угловых клеток поля. Попавшие на них фишки "замкнуть" нельзя.
2. В дебюте надо стараться вести борьбу в центре поля - в квадрате, выделенном точками, чтобы раньше времени не допустить соперника на крайние клетки поля.
3. Правила игры просты и позволяют научиться играть очень быстро. Но для совершенствования ее тактических возможностей требуется много времени.

ЖЕЛАЕМ УДАЧИ!

3.9. Игровая программа "ПИТОН"

Правила игры

Загрузка программы в ОЗУ осуществляется при помощи директивы программы "МОНИТОР":
I=ПИТОН. Загрузка по директиве J.

После демонстрации заставки "ПИТОН", нужно нажать <ПРОБЕЛ>. На запрос компьютера "ВЫБРАТЬ СКОРОСТЬ", нужно нажать клавишу <↑>, при этом заданная скорость будет изменяться. Установите необходимую скорость и нажмите <ПРОБЕЛ>.

На игровом поле появляется "ПИТОН", который "ест" яблоки и кроликов.

Передвижение "ПИТОНА" по полю осуществляется клавишами

<↑>, <↓>, <→>, <←>

При этом нельзя задавать противоположные движения. Например, "ПИТОН" движется вправо, а вы заставляете его двигаться влево.

Количество звеньев у "ПИТОНА" равно К умноженное на М, где К-количество съеденных кроликов, а М - количество сведенных яблок. Количество сведенных яблок высвечивается ниже игрового поля.

На пути движения "ПИТОНА" появляются преграды в виде заштрихованных четырехугольников и "мангуст". Встреча с ними ведет к проигрышу. На игровом поле появляются "амфоры" с "противоядием" от "мангуст". Если "ПИТОН" съел "амфору", то встреча с "мангустой" грозит ему потерей только 13 звеньев.

Ваша задача вырастить "ПИТОНА" как можно большей длины. Но при этом необходимо помнить, что к проигрышу ведет:

1. Задание противоположного движения.
2. Встреча с препятствиями.
3. Встреча с "мангустом", если у "ПИТОНА" недостаточно противоядия.
4. Столкновение со стенкой игрового поля.
5. Столкновение "ПИТОНА" с самим собой.

ЖЕЛАЕМ ВАМ УСПЕХА!

3.10. Игровая программа "ПОДВОДНАЯ АТАКА"

Программа написана на языке БЕЙСИК.

Загрузка программы производится при помощи МОНИТОРА: I=АТАКА <ВК>

Запуск программы:

J<ВК>

RUN <ВК>

После вывода на экран заставки в сопровождении музыкального фрагмента мигает сообщение "Нажмите <пробел>!"

После нажатия клавиши <пробел> выводится на экран вопрос: "Нужны ли правила (Y/N)?" Нажимаете клавишу "Y" для печати правил, в противном случае нажимаете клавишу "N".

Правила игры. Вы - командир подводной лодки А736-2. Ваша задача уничтожить надводный флот противника. С помощью спутниковой квантовой фотолокации и бортовых компьютеров получена карта дислокации кораблей противника. Для защиты от вражеской разведки данные на карте закодированы одним из трех способов:

(А) - симметричным отражением относительно вертикальной прямой, проходящей через середину карты;

(В) - таким же отражением, но относительно горизонтальной линии;

(С) — обоими способами (А) и (В) сразу.

Карта разделена на квадраты, координаты левого нижнего квадрата равны 1,1.

Чтобы выстрелить достаточно ввести координаты цели X,Y через запятую и нажать <ВК>. О результатах выстрела сообщит бортовой компьютер вашей лодки.

На поле боя расположены два миноносца (по одному квадрату), два крейсера (по два квадрата), два линкора (по три квадрата) и два авианосца (по четыре квадрата каждый).

Чтобы уничтожить корабль, необходимо попасть во все квадраты, на которых он находится.

Для досрочного окончания игры введите координаты выстрела 0,0.

СЧАСТЛИВОГО ПЛАВАНИЯ !!!

3.11. Игровая программа "ШАХМАТЫ"

Загрузка программы осуществляется при помощи директивы программы МОНИТОР: I=ШАХМАТЫ. Запуск по директиве J.

Правила игры.

Сначала необходимо выбрать уровень сложности игры.

На вопрос: "LEVEL 0 OR 1 ?" - отвечаете 0 или 1.

На следующий вопрос: "COMPUTER TO PLAY WHITE (0) OR BLACK (1)?" - ответьте <0> - если вы предоставляете компьютеру играть белыми фигурами, и <1> - если черными.

Далее делать хода согласно правилам игры в шахматы, указывая координаты начального положения фигуры и через запятую - координаты конечного положения фигуры.

ЖЕЛАЕМ ВАМ УСПЕХА!

3.12. Игровая программа "КЛАД"

Увлекательная игра. Развивает скорость реакции и сообразительность.

Игра "Клад" записана на стороне В поставляемой кассеты пятым файлом. В память загружается директивой I МОНИТОРА, запуск по директиве J.

После заставки "Клад" появляется картинка с бегающим человечком. С помощью клавиш: <↑>, <↓>, <→>, <←> человек может передвигаться. Выбираем скорость, с которой он будет бегать, и оказываемся на первом уровне.

Цель у человечка такова: ему необходимо найти клад с ключиком. С помощью этого ключика он может открыть все дверцы, которые ему мешают перебежать на следующий уровень. Но это сделать не просто, т.к. его преследуют бандиты. Поэтому вам надо не только найти клад, но и убежать от бандитов.

Если же они его догонят, то придется начинать сначала. Но вы не огорчайтесь, у вас есть пять попыток. Так же есть еще клад не только с ключиком, но и с попытками. Если вы найдете этот клад у вас увеличится число попыток. Если понадобится пройти через стенку, то это не так сложно.

Человечек может стрелять при нажатых одновременно клавиш <пробел> и <→> или <←> (смотря в какую сторону нужно стрелять). Существуют мостики, через которые человек может перебежать, но мостик ломается если на него прыгать.

ЖЕЛАЕМ УДАЧИ!

3.13. Игровая программа "TETRIS"

Правила игры:

Загрузка программы в ОЗУ осуществляется при помощи директивы программы МОНИТОР: I=TETRIS <BK>, запуск по директиве J.

После заставки "TETRIS" появляется таблица. Набираем имя игрока. Затем нажимаем клавишу <BK>, после исчезновения курсора нажимаем <пробел> и на экран высвечивается игра: начинают падать разные фигурки. Их необходимо уложить так, чтобы не оставалось пустых клеточек.

Клавишами <→> и <←> фигурки можно передвигать вправо или влево. С помощью клавиши <↑> фигурки можно переворачивать. Вверху таблицы написана цель. Там указано количество очков, которые нужно набрать, чтобы перейти на другой уровень. Каждая фигурка имеет свое количество очков, поэтому т.к. с каждым уровнем фигурки усложняются, то меняется цель игры. Если долго играть на данном уровне, то будет увеличиваться скорость падения фигуры.

3.14. Игровая программа "ПЕЩЕРА"

Представьте себе, что вы попали в таинственную пещеру, где водится нечистая сила. В пещере хранится видимо-невидимо сокровищ, и вам необходимо собрать их как можно больше. Для защиты от нечистой силы, у вас имеется оружие. Применяйте его по назначению, но будьте осторожны: пуля может от ricochet от стены и попасть в вас. К тому же следует помнить, что уничтоженные "черти" имеют способность заново материализоваться. Если вы сумеете собрать здесь все сокровища, то сразу же попадете в другую пещеру, где "черти" много проворнее. Там от вас потребуется большая расторопность. Если же удача покинет вас и вы попадете в лапы нечистой силы, не отчаивайтесь - попытайте счастье еще раз (всего предоставляется 5 попыток). Запуск программы производится с "нулевого" адреса. На экране появится лабиринт пещеры. Игра начнется после нажатия цифры от 1 до 3, определяющих выбранную вами сложность (1 — максимальная). Управляют движением следующими клавишами:

- < 4 > - влево
- < 5 > - вниз
- < 6 > - вправо
- < 7 > - выстрел влево
- < 8 > - вверх
- < 9 > - выстрел вправо
- < 0 > - стоп
- < - > - повторный запуск игры.

3.15. Игровая программа "ШАХМАТЫ2"

Загрузка программы осуществляется при помощи директивы программы МОНИТОР:
I=ШАХМАТЫ2.

Запуск по директиве J

После демонстрации рекламной заставки на экране появляется шахматная доска с расставленными фигурами. Компьютер играет черными фигурами. В левом нижнем углу появляется запрос:

Уровень :

На него необходимо ответить выбором уровня (цифры от 0 до 9). После чего в правом верхнем углу доски появляется мерцающий квадрат. Управляя квадратом при помощи стрелок его подводят к нужной фигуре и отмечают эту фигуру <пробелом>. Далее необходимо перевести квадрат на ту позицию куда вы хотите сделать ход и опять нажать <пробел>. Через некоторое время компьютер отвечает вам своим ходом. Рокировки осуществляются при помощи клавиш <3> и <4>. Начать игру сначала можно при помощи клавиши <1>. Игра проходит по правилам игры в шахматы.

ЖЕЛАЕМ ПОБЕДЫ !

3.16. Игровая программа "БИЗНЕС"

Игра "Бизнес" загружается с магнитофона вместе с интерпретатором языка БЕЙСИК:
I=БИЗНЕС <BK>.

Запуск игры производится следующим образом: по директиве J МОНИТОРА осуществляется выход в БЕЙСИК, после чего необходимо дать команду RUN.

Игра "Бизнес" является аналогом игры "Менеджер" в машинном варианте. Эта игра развивает у играющих коммерческую смекалку и умение ориентироваться в мире бизнеса. В игру могут играть от одного до четырех человек.

В состав играющих может быть включена сама ЭВМ. Для этого на запрос машины:

введите имя игрока?

Необходимо ввести ЭВМ N, где N изменяется от одного до четырех.

Цель игры заключается в том, что каждый из играющих должен скупить как можно больше фирм и получить как можно больший доход.

Игра происходит следующим образом. Каждый из играющих едет на автомобиле. Для того, чтобы фишка сдвинулась с места нужно нажать <BK>. Под таблицей дана стоимость бензина, но она не постоянна. В течении игры стоимость бензина будет то увеличиваться, то уменьшаться. Если бензин закончится в пути, то он будет стоить в три раза дороже. Заправлять можно не более 16 литров. После нажатия количества литров, необходимо нажать <BK>.

Если игрок попал в критическое положение, т.е. у него закончились все деньги, то он имеет право одолжить нажав на клавишу <?>. У него появится долг, который на протяжении игры будет увеличиваться, т.е. растут проценты. Если же этот игрок поправил свое материальное положение и у него появились деньги, то погасить долг можно нажав клавишу <J>.

Вы попали в KAZINO. Машина выдает вопрос:

Какова ваша ставка?

Делаете ставку и нажимаете <BK>. После каждого загаданного числа, также нажимаете <BK>, кроме последнего, и терпеливо, ждете результат вашего выигрыша.

Вы попали в банк. У вас есть возможность или положить деньги в банк, или взять. Если вы решили взять нажимаете для этого <—> (минус)A, где A - количество денег.

Если вы хотите купить фирму нажимаете <пробел>, если нет - <—>(минус).

В пути вы можете натолкнуться на неприятности. Вас могут оштрафовать, конфисковать какую-либо фирму или отправить в гостиницу на 2 или 4 суток. После отдыха в гостинице вы опять отправляетесь в путь.

ЖЕЛАЕМ БОЛЬШИХ ФИНАНСОВЫХ УСПЕХОВ !

3.17. Игровая программа "БУРАН"

Игра "Буран" написана для радиоконструктора "Специалист" и записана на кассете в формате радиоконструктора ЛИК.

Перед загрузкой игры необходимо загрузить "МОНИТОР-SP". После чего загрузить игру директивой I=БУРАН <BK>.

Запуск игры по директиве J программы "МОНИТОР-1M".

После запуска на экране появляется заставка с вопросом:

Инструкция нужна? (1,0)

Если вам нужна инструкция нажмите 1, в противном случае 0.

Вы можете выбрать уровень сложности цифрой от 1 до 3.

Правила игры:

Представьте себе, что вы в кабине космического корабля типа "Буран".

На стороне противника маневренные спутники-истребители с лазерным вооружением малой дальности. Ваша задача - уничтожить вражеские аппараты раньше, чем они приблизятся на расстояние выстрела. На корабле имеется неограниченный запас управляемых ракет. Пуск ракеты - клавиша <↑>. Управление полетом ракеты - клавиши < ← > и < → >.

ЖЕЛАЕМ ПОБЕДЫ !