ДРАЙВЕРЫ УСТРОЙСТВ

описание

АННОТАЦИЯ

В документе "Руководство пользователя" приводятся назначение, условия применения и особенности использования программного средства "Драйверы устройств", указываются действия, обеспечивающие разработку программного обеспечения с применением поставляемого пакета подпрограмм.

В руководстве основное внимание уделено содержательному описанию подпрограмм "Драйверы устройств", особенностям их использования при разработке программных средств для БПЭВМ "ВЕКТОР-06Ц".

Документ содержит сведения, необходимые пользователю для применения программного средства "Драйверы устройств" при разработке программ на языке ассемблер для БПЭВМ "ВЕКТОР-06Ц".

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ ДРАЙВЕРОВ УСТРОЙСТВ.

Программное обеспечение "Драйверы устройств" представляет собой пакет подпрограмм, выполняющих определенные функции, такие как запись и чтение данных с магнитной ленты, работа с экраном дисплея, программирование цвета и музыки, ввод информации с клавиатуры и т.д..

При программировании на языке ассемблер для БПЭВМ "ВЕКТОР-06Ц" практически всегда необходимо программировать вышеупомянутые функции. Поставляемый пакет подпрограмм (драйверов) позволяет значительно упростить процесс программирования на языке ассемблер.

Пакет можно подразделить на следующие группы подпрограмм (драйверов):

- драйверы записи (чтения) на магнитную ленту;
- драйверы вывода графической информации на дисплей в цветном режиме;
- драйверы клавиатуры и музыкального синтезатора;
- драйверы вывода информации на печатающее устройство;
- драйверы, выполняющие операции над двоичными числами.

Использование драйверов устройств значительно уменьшает трудоемкость программирования на ассемблере. Применение подпрограмм (драйверов) избавляет программиста от разработки собственных программ ввода-вывода на магнитную ленту, опроса клавиатуры, создания знакогенератора, разработки программ, выполняющих арифметические операции и т.д.

Обращаться к драйверам устройств можно в любом месте программы. Параметры задаются в регистрах перед вызовом соответствующего драйвера. Выполнение драйверов устройств обычно не изменяет содержимого регистров.

Описание подпрограмм (драйверов) приводится в разделе 4 настоящего руководства.

В состав поставляемого пакета включены следующие подпрограммы:

@BINDEC	- преобразование двоичного числа в десятичное в виде строки символов в коде КОИ-7;
@BORD	- установка цвета бордюра (границы);
@CIRCLE	- рисование окружности;
@CMPHD	- сравнение двоичных чисел;
@COLOR	- установка цвета отображения;
@CONIN	- ввод символа с клавиатуры;
@CONOUT	- вывод символа на дисплей;
@DECBIN	- преобразование десятичного числа, заданного в виде строки символов в коде КОИ-7, в двоичное число;
@DUMP	- вывод содержимого регистра (А) в шестнадцатеричном виде;
@EADD	- сложение двоичных чисел;
@EDIVM	- деление двоичных чисел;
@EMULT	- умножение двоичных чисел;
@ESIGN	- определение знака двоичного числа;
@ESUB	- вычитание двоичных чисел;
@FON	- установка цвета экрана;
@FPLAY	- определение звучания музыкальных каналов;

@GET	- сохранение графической информации с экрана в оперативной памяти;
@INIT	- инициализация работы пакета;
@INKEY	- ввод символа с клавиатуры без ожидания ввода;
@INTAP	- ввод байта с магнитной ленты;
@KEY	- опрос состояния клавиатуры;
@LINE	- рисование линий и прямоугольников, подготовка вывода символов увеличенного размера;
@LIST	- вывод символа на печать;
@LSCR	- печать экрана в графическом режиме;
@MASK	- установка маски вывода;
@OUTAP	- вывод байта на магнитную ленту;
@PAINT	- закрашивание замкнутых областей;
@PCIRCLE	- установка параметров окружности;
@PGET	- определение цвета текущей точки;
@PLAY	- программирование музыкального синтезатора;
@PLOFF	- прерывание звучания музыкальных каналов;
@PLOT	- рисование точки, установка графического курсора;
@PUT	- вывод на экран графической информации из области оперативной памяти;
@SCOLOR	- установка физических цветов;
@SCONR	- установка скорости обмена информацией с магнитной лентой;
@SCROL	- перемещение экрана по вертикали;
@SLIST	- установка типа принтера;
@SPIC	- вывод сообщения по адресу, определенному регистрами H и L;
@SPLAN	- установка экранных областей;
@TIME	- определение/установка текущего времени.

2. УСЛОВИЯ ПРИМЕНЕНИЯ.

Для разработки программного обеспечения с использованием пакета подпрограмм (драйверов) необходима следующая минимальная конфигурация технических средств:

- БПЭВМ "ВЕКТОР-06Ц";
- телевизионный приемник цветного или черно-белого изображения;
- бытовой кассетный магнитофон.

3. ПОДКЛЮЧЕНИЕ ПАКЕТА ПОДПРОГРАММ (ДРАЙВЕРОВ) К ПРОГРАММЕ ПОЛЬЗОВАТЕЛЯ

Программа пользователя, использующая пакет подпрограмм (драйверов), должна быть написана на языке ассемблер. После подключения пакета подпрограмм полученная рабочая программа может быть выполнена на БПЭВМ "ВЕКТОР-06Ц" автономно.

Пакету подпрограмм (содержащему также программу подключения драйверов к программе пользователя) необходимо 11 Кбайт оперативной памяти. Сам же пакет подпрограмм после подключения к программе пользователя занимает 8 Кбайт оперативной памяти.

Начало программы пользователя должно содержать команды, выполняющие установку адресов программы пользователя и пакета подпрограмм, а также команды, выполняющие инициализацию пакета подпрограмм (подпрограмма @INIT).

В приложении приведен образец программы пользователя. При использовании приведенного образца пакет подпрограмм должен располагаться с адреса 100H, а программа пользователя - с адреса 1F00H.

Подключение пакета драйверов к программе пользователя выполняют в следующем порядке:

- загрузить и запустить программу монитор-отладчик (при выборе адресов для монитора нажать цифру 1);
- по команде R загрузить программу "Драйверы устройств";
- по команде R загрузить программу пользователя с адреса не менее 3000H, так как до адреса 26FFH загружены драйверы с программой настройки пакета;
- по команде G100 запустить программу "Драйверы устройств".

На экран будет выведено:

```
"ДРАЙВЕРЫ УСТРОЙСТВ" V(2.0)
... П/О "СЧЕТМАШ" 1989Г. ...
КОМПОНОВКА ВАШЕЙ ПРОГРАММЫ ВЫПОЛНЕНА? (Y/N) -
```

После ответа, что компоновка вашей программы не выполнена (N) на экран выводится следующее сообщение:

```
ДЛЯ КОМПОНОВКИ ВАШЕЙ ПРОГРАММЫ С ПАКЕТОМ
"ДРАЙВЕРЫ УСТРОЙСТВ" И ЕЕ ПЕРЕСЫЛКИ ПО
РАБОЧЕМУ АДРЕСУ, ВВЕДИТЕ:

НАЧАЛЬНЫЙ АДРЕС ВАШЕЙ ПРОГРАММЫ (ORG ADDR) -
(Для нашего примера - 1F00H.)

КОНЕЧНЫЙ АДРЕС ВАШЕЙ ПРОГРАММЫ (END ADDR) -
(Он равен 2F13H. Это первый адрес, следующий за последним
байтом программы пользователя.)

АДРЕС НАЧАЛА ВАШЕЙ ПРОГРАММЫ (LOAD ADDR) -
(Для данного случая - 3000H. Это адрес с которого находится
в памяти после загрузки по команде R пользовательская прог-
рамма.)

После ответа на эти вопросы монитор - отладчик компонует
вашу программу с подпрограммами и на экране появляется сообщение

КОНЕЦ РАБОТЫ (Y/N) -
```

При положительном ответе (Y) вы возвращаетесь в программу монитор - отладчик. При отрицательном ответе (N) на экране появится сообщение:

ВЫВЕСТИ ПРОГРАММУ НА МАГНИТНУЮ ЛЕНТУ? (Y/N) -

Если вы нажали (Y), то на экране появится меню режимов записи. После выбора режима высветится сообщение спрашивающее о готовности магнитофона. При положительном ответе ваша скомпонованная программа выгружается на ленту. После окончания записи на магнитную ленту на экран выводится сообщение:

ЗАГРУЗИТЬ ПРОГРАММУ ДЛЯ ВЫПОЛНЕНИЯ (Y/N) -

Это же сообщение появляется, если вы отказались выводить программу на магнитную ленту.

Если вы желаете запустить программу на выполнение, то нажмите (Y) и ваша программа начнет выполнятся. Если вы отказываетесь запустить программу на выполнение, т.е. нажали (N), появляется сообщение, спрашивающее желаете ли вы закончить работу.

Процесс настройки заключается в том, чтобы расположить пакет подпрограмм и программу пользователя в оперативной памяти так, что в сформированной программе вначале располагается пакет подпрограмм, а затем - программа пользователя.

Загрузку пакета подпрограмм и программ пользователя в оперативную память и запись сформированной программы на магнитную ленту можно выполнить с помощью монитора-отладчика.

Рекомендуется задавать:

- адрес пакета - 100H;
- адрес загрузки программы - 3000H;
- адрес программы - 2000H.

Необходимо иметь в виду, что с адреса 8000H располагается экранная область. Поэтому при подключении пакета подпрограмм к программе пользователя необходимо располагать программы до адреса экранной области, т.е. до адреса 8000H.

Использование экранной области памяти при подключении драйверов устройств к программе пользователя возможно только с учетом определенных требований (установка экранных областей и цветов).

4. ОПИСАНИЕ ДРАЙВЕРОВ УСТРОЙСТВ

@BINDEC

Назначение: Подпрограмма @BINDEC предназначена для преобразования двоичного числа в десятичное в виде строки символов в коде КОИ-7.

Входные: Регистры (HL) должны содержать двоичное число, предназначенное для преобразования; регистры (DE) должны содержать адрес поля, где будет помещена преобразованная строка символов.

Выходные: Регистры (DE) содержат адрес следующего байта после преобразованной строки символов; регистр (A) содержит число символов в строке.

Действие: Подпрограмма @BINDEC преобразует двоичное число (в диапазоне от -32767 до +32767), содержащееся в регистрах H и L в десятичное в виде строки символов, адрес которой задается в регистрах D и E. При преобразовании двоичного числа в строку происходит подавление незначащих нулей. В результате выполнения подпрограммы формируется строка символов (от одного до шести) со знаком, если число отрицательное.

@BORD

Назначение: Подпрограмма @BORD предназначена для установки цвета границы экрана.

Входные: Регистр (A) должен содержать число (0 - 15), определяющее цвет границы.

Действие: Подпрограмма @BORD запоминает в оперативной памяти цвет границы экрана, который задается в регистре (A) в виде двоичного числа.
Следует иметь ввиду, что подпрограмма @BORD только запоминает в памяти цвет границы экрана, а установка цвета границы производится при смене кадра экрана (смена кадра происходит через каждые 20 мсек).

@CIRCLE

Назначение: Подпрограмма @CIRCLE предназначена для рисования окружностей и дуг.

Входные: Регистры (HL) должны содержать число, определяющее координату центра окружности по оси X.
Регистры (DE) должны содержать число, определяющее координату центра окружности по оси Y.
Регистр (A) должен содержать число, определяющее радиус окружности.

Действие: Подпрограмма @CIRCLE производит рисование окружностей и дуг с центром, координаты которого задаются в регистрах (HL) и (DE). Регистр (A) определяет радиус.
Надо иметь ввиду, что подпрограмма @CIRCLE использует параметры окружности, которые устанавливаются подпрограммой @PCIRCLE. Поэтому перед выполнением подпрограммы @CIRCLE необходимо установить параметры окружности, выполнив подпрограмму @PCIRCLE.

@CMPHD

Назначение: Подпрограмма @CMPHD предназначена для сравнения двоичных чисел с установкой флагов условий.

Входные: Регистры (HL) и (DE) должны содержать сравниваемые числа.

Выходные: Устанавливаются флаги условий CY и Z.

Действие: Подпрограмма @CMPHD производит сравнение двоичных чисел, содержащихся в регистрах (HL) и (DE). По результатам сравнения устанавливаются флаги условий CY и Z. Если Z=0 и CY=0 (переноса не было), то двоичные числа в регистрах (HL) и (DE) равны. Если CY=1 (был перенос), то двоичное число в регистрах (HL) меньше двоичного числа в регистрах (DE). Если Z не равно 0 и CY=0 (переноса не было), то двоичное число в регистрах (HL) больше двоичного числа в регистрах (DE).

@COLOR

Назначение: Подпрограмма @COLOR предназначена для установки цвета отображения

Входные: Регистр (A) должен содержать число (0-15), определяющее цвет отображения.

Действие: Подпрограмма @COLOR производит установку цвета отображения. Цвет отображения устанавливается в зависимости от заданного в регистре (A) числа, а также в зависимости от соответствия цветов отображения и физических цветов, установленных подпрограммой @SCOLOR.

@CONIN

Назначение: Назначение @CONIN предназначена для ввода символа с клавиатуры.

Входные: Регистр (A) содержит код нажатой клавиши.

Действие: Подпрограмма @CONIN ждет нажатия клавиши на клавиатуре. Только после нажатия клавиши подпрограмма @CONIN заносит код нажатой клавиши в регистр (A) и заканчивает выполнение.
Следует помнить, что при нажатии клавиши на клавиатуре код нажатой клавиши помещается в буфер клавиатуры. Подпрограмма @CONIN выбирает символы из буфера. Поэтому, если буфер клавиатуры не пустой (содержит уже введенные символы), то будет выбираться символ из буфера, а не тот символ, который был введен последним.

@CONOUT

Назначение: Подпрограмма @CONOUT предназначена для вывода символа на экран.

Входные: Регистр (C) должен содержать код символа, выводимого на экран.

Действие: Подпрограмма @CONOUT осуществляет вывод на экран символа. Код символа должен быть помещен в регистр (C).

@DECBIN

Назначение: Подпрограмма @DECBIN предназначена для преобразования целого десятичного числа, заданного в виде строки символов в коде КОИ-7, в двоичное число.

Входные: Регистры (DE) должны содержать адрес строки символов.

Выходные: Регистры (HL) содержат двоичное число после преобразования строки символов;
Регистры (DE) содержат адрес последнего символа преобразованной строки, которой не является цифрой

Действие: Подпрограмма @DECBIN осуществляет преобразование строки символов, адрес которой помещен в регистры (DE) в двоичное число. После преобразования двоичное число помещается в регистры (HL). Строка символов должна иметь формат целого десятичного числа со знаком или без знака в диапазоне от -32767 до +32767.

@DUMP

Назначение: Подпрограмма @DUMP предназначена для вывода на экран в шестнадцатеричном виде содержимого регистра (A).

Входные: Регистр (A) должен содержать двоичное число, которое должно быть выведено на экран в шестнадцатеричном виде.

Действие: Подпрограмма @DUMP осуществляет вывод на экран в шестнадцатеричном виде содержимого регистра (A).

@EADD

Назначение: Подпрограмма @EADD предназначена для выполнения операции сложения двух чисел.

Входные: Регистры (HL) и (DE) должны содержать слагаемые в виде целых двоичных чисел в диапазоне от -32767 до +32767;

Выходные: Регистры (HL) содержат сумму заданных слагаемых.

Действие: Подпрограмма @EADD производит сложение чисел заданных в регистрах (HL) и (DE). Результат помещается в регистрах (HL).

@EDIVM

Назначение: Подпрограмма @EDIVM предназначена для выполнения операции деления двух чисел.

Входные: Регистры (HL) должны содержать значение делимого; регистры (DE) должны содержать значение делителя.

Выходные: Регистры (HL) содержат частное, регистры (DE) содержат остаток от деления заданных чисел.

Действие: Подпрограмма @EDIVM производит деление целых двоичных чисел, заданных в регистрах (HL) и (DE). Частное от деления заданных чисел помещается в регистры (HL), а остаток от деления - в регистры (DE). Значения чисел должны быть в диапазоне от -32767 до +32767.

@EMULT

Назначение: Подпрограмма @EMULT предназначена для выполнения операции умножения двух чисел.

Входные: Регистры (HL) и (DE) должны содержать сомножители в виде целых двоичных чисел в диапазоне от -32767 до +32767.

Выходные: Регистры (HL) содержат произведение заданных сомножителей.

Действие: Подпрограмма @EMULT производит умножение чисел, заданных в регистрах (HL) и (DE). Результат помещается в регистрах (HL).

@ESIGN

Назначение: Подпрограмма @ESIGN предназначена для определения знака числа.

Входные: Регистры (HL) должны содержать целое двоичное число.

Выходные: В байте состояния устанавливается флаг знака S в зависимости от знака числа:
0 - число положительное или ноль;
1 - число отрицательное.

Действие: Подпрограмма @ESIGN устанавливает в байте состояния флаг знака S в зависимости от знака заданного числа в регистрах (HL).

@ESUB

Назначение: Подпрограмма @ESUB предназначена для выполнения операции вычитания двух чисел.

Входные: Регистры (HL) должны содержать значение уменьшаемого;
регистры (DE) должны содержать значение вычитаемого.

Выходные: Регистры (HL) содержат разность заданных чисел.

Действие: Подпрограмма @ESUB производит вычитание целых чисел, заданных в регистрах (HL) и (DE). Результат вычитания помещается в регистры (HL). Значения чисел должны быть в диапазоне от -32767 до +32767.

@FON

Назначение: Подпрограмма @FON предназначена для установки цвета фона экрана.

Входные: Регистр (A) должен содержать число (0 - 255), определяющее цвет фона.

Действие: Подпрограмма @FON запоминает в оперативной памяти цвет фона, который задается в регистре (A) в виде двоичного числа.
Следует иметь ввиду, что подпрограмма @FON только запоминает в памяти цвет экрана, а установка цвета экрана производится при смене кадра (смена кадра происходит через каждые 20 мсек).

@FPLAY

Назначение: Подпрограмма @FPLAY предназначена для определения включенных музыкальных каналов.

Входные: Регистр (A) содержит байт, определяющий состояние музыкальных каналов.

Действие: Подпрограмма @FPLAY определяет состояние музыкальных каналов и заносит в регистр (A) байт состояния каналов:

```
                  (A) = 00000XXX
                                |||
0 - Канал выкл.                 ||+-- нулевой канал
1 - Канал вкл.                  |+---  первый канал
                                +----  второй канал
```

@GET

Назначение: Подпрограмма @GET предназначена для запоминания графической информации с экрана в ОЗУ.

Входные: Регистры (B) и (C) должны содержать числовые значения, которые определяют размер прямоугольника, ограничивающий графическую информацию на экране;
регистры (DE) и (HL) должны содержать адрес начала и адрес конца области оперативной памяти, в которую будет помещена графическая информация с экрана.

Действие: Подпрограмма @GET запоминает в области оперативной памяти графическую информацию с экрана, ограниченную прямоугольником, левый нижний угол которого определяется текущим положением графического курсора, а размеры сторон - содержимым регистров (B) и (C) по оси Y и по оси X соответственно.
Адрес начала области оперативной памяти помещается в регистрах (DE), адрес конца - в регистрах (HL).
Адрес конца области оперативной памяти определяется по формуле:

```
ADRK = ADRN + Y*X/2+2,
```

где

```
ADRN - адрес начала;
ADRK - адрес конца;
Y    - размер стороны прямоугольника по оси Y;
X    - размер стороны прямоугольника по оси X;
```

@INIT

Назначение: Подпрограмма @INIT предназначена для инициализации работы пакета подпрограмм (драйверов).

Действие: Подпрограмма @INIT заносит по адресу 38H код команды JMP и адрес перехода (39H, 3AH) на подпрограмму обработки внутренних прерываний.
Подпрограмма @INIT формирует также таблицу для вычисления адресов элементов графических изображений (512 байт).
Для инициализации пакета подпрограмм необходимо в начале программы пользователя предусмотреть выполнение подпрограммы @INIT.

@INKEY

Назначение: Подпрограмма @INKEY предназначена для ввода символов с клавиатуры без ожидания ввода.

Выходные: Регистр (A) содержит код нажатой клавиши или 0FFH.

Действие: Подпрограмма @INKEY заносит в регистр (A) код нажатой клавиши или 0FFH, если при обращении к подпрограмме ни одна из клавиш не была нажата (буфер клавиатуры пуст).

@INTAP

Назначение: Подпрограмма @INTAP предназначена для ввода байта с магнитной ленты.

Входные: Регистр (A) должен содержать следующие значения:
FF - чтение одного байта после байта синхронизации (0E6H);
08 - чтение одного байта без поиска байта синхронизации.

Выходные: Регистр (A) содержит значение введенного байта.

Действие: Подпрограмма @INTAP вводит один байт с магнитной ленты и заносит его значение в регистр (A). Скорость считывания информации с магнитной ленты устанавливается подпрограммой @SCONR.

@KEY

Назначение: Подпрограмма @KEY предназначена для опроса состояния клавиатуры.

Выходные: Регистр (A) содержит код состояния буфера клавиатуры:
00H - нет символа в буфере;
0FFH - есть символ в буфере;

Действие: Подпрограмма @KEY производит проверку буфера клавиатуры на наличие в нем введенных символов. Результат выполнения возвращается в регистр (A). Если в буфере клавиатуры есть символы, или производилось нажатие клавиши УС, то в регистре (A) возвращается значение 0FFH.
Если в буфере клавиатуры нет символов, то в регистре (A) возвращается значение 00H.

@LINE

Назначение: Подпрограмма @LINE предназначена для рисования линий, прямоугольников, а также для подготовки вывода символов увеличенного размера.

Входные: Регистры (B) и (C) должны содержать числовые значения, которые определяют координаты (соответственно по оси Y и по оси X) начала линии или левого угла прямоугольника или размеры символов увеличенного размера в зависимости от заданного режима;
регистр (A) указывает режим:
0 - линия;
1 - прямоугольник;
2 - закрашенный прямоугольник;
3 - подготовка вывода символов увеличенного размера.

Действие: Подпрограмма @LINE рисует линии, вычерчивает прямоугольники или закрашенные прямоугольники в зависимости от заданного режима, а также подготавливает вывод символов увеличенного размера.

@LIST

Назначение: Подпрограмма @LIST предназначена для вывода на печать одного символа.

Входные: Регистр (C) должен содержать код символа, выводимого на печать.

Действие: Подпрограмма @LIST осуществляет вывод на печать одного символа для следующих типов принтеров:
- ROBOTRON - CM 6329
- EPSON FX-85, RAVI-8010M.

@LSCR

Назначение: Подпрограмма @LSCR предназначена для печати экрана в графическом режиме

Входные: Регистр (A) должен содержать номер графического режима в диапазоне от 0 до 7.

Действие: Подпрограмма @LSCR производит печать экрана в графическом режиме. Номер графического режима задается в регистре (A).

@MASC

Назначение: Подпрограмма @MASC предназначена для установки маски вывода.

Входные: В регистры (DE) необходимо поместить адрес таблицы, содержащей 8 байт маски.

Действие: Подпрограмма @MASC устанавливает маску вывода в соответствии с заданной таблицей, адрес которой определяется регистрами (DE).
Маска вывода представляет собой матрицу 8x8 (8 байт на 8 бит), которая описывает некоторое графическое изображение (образец). Каждый байт соответствует горизонтальной линии в образце, а значение бита маски равно единице, то соответствующая точка на экране будет изменяться. Если значение бита маски равно нулю, то соответствующая точка на экране не будет изменяться.

Установленная подпрограммой @MASC маска вывода используется в дальнейшем подпрограммой @LINE для вывода на экран изображений, соответствующих заданному образцу.

Например, если задать маску в виде 055H 0AAH 055H 0AAH 055H 0AAH 055H 0AAH, то изображение, полученное при выводе подпрограммы @LINE в режиме 2 (рисование закрашенного прямоугольника) представляет собой мелкоструктурную сетку.

@OUTAP

Назначение: Подпрограмма @OUTAP предназначена для вывода байта на магнитную ленту.

Входные: Регистр (C) должен содержать выводимый байт.

Действие: Подпрограмма @OUTAP выводит на магнитную ленту байт данных из регистра (C).
Скорость записи информации на магнитную ленту устанавливается подпрограммой @SCONR.

@PAINT

Назначение: Подпрограмма @PAINT предназначена для закрашивания замкнутых областей.

Входные: Регистры (B) и (C) должны содержать числовые значения, определяющие координаты (соответственно по оси Y и по оси X) точки на экране, с которой будет начинаться закрашивание:
регистр (D) должен содержать число, определяющее цвет заполнения;
регистр (E) должен содержать число, определяющее цвет границы.

Действие: Подпрограмма @PAINT закрашивает произвольные замкнутые области на экране.
Область для закрашивания должна быть ограничена цветом границы или цветом закрашивания.

@PCIRCLE

Назначение: Подпрограмма @PCIRCLE предназначена для установки параметров окружности.

Входные: Регистр (A) должен содержать масштаб по оси Y
регистр (C) - масштаб по оси X (0-255);
регстры (DE) - значение начального угла (0-2047);
регистры (HL) - значение конечного угла (0-2047).

Действие: Подпрограмма @PCIRCLE запоминает в оперативной памяти заданные параметры окружности, которые используются подпрограммой @CIRCLE для рисования окружности на экране. Меняя масштабы по оси X и по оси Y можно рисовать вытянутые вдоль этих осей окружности (овалы). При совпадении масштабов по оси X и по оси Y подпрограммой @CIRCLE рисуется окружность.

@PGET

Назначение: Подпрограмма @PGET предназначена для определения цвета текущей точки.

Выходные: Регистр (A) возвращается номер цвета точки с координатами текущего положения графического курсора.

Действие: Подпрограмма @PGET заносит в регистр (A) номер цвета точки.

@PLAY

Назначение: Подпрограмма @PLAY предназначена для программирования музыкального синтезатора.

Входные: Регистры (BC), (DE) и (HL) должны содержать адреса областей памяти, содержащих музыкальные предложения на музыкальном макроязыке (ММЯ) соответственно для нулевого, первого и второго каналов аппаратного звукосинтезатора.

Действие: Подпрограмма @PLAY запускает музыкальные каналы в соответствии с заданными предложениями на ММЯ. Можно запретить запуск любого из каналов. Для этого необходимо занести ноль в соответствующие регистры. Предложение на ММЯ для любого из трех независимых звуковых каналов состоит из символов, после которых может следовать числовой параметр.

C	D	E	F	G	A	B
до	ре	ми	фа	соль	ля	си

Эти символы вызывают исполнение соответствующей ноты в текущей октаве и текущей длительности. Ноты можно модифицировать добавлением:

, + , - , . , <длительность>.

Знаки "#" и "+" указывают диез, знак "-" обозначает бемоль. После этих добавлений может быть указана длительность ноты, если она отличается от стандартной (заданной "L") по следующему правилу: 1 - целая, 2 - половинная, 4 - четвертная и т.д. Далее можно поставить одну или несколько точек, каждая из которых увеличивает длительность ноты наполовину.

O<число> - определяет выбор октавы от 0 до 8. Октава остается неизменной до следующего "O". Если в начале строки не задано "O", то при запуске подпрограммы @PLAY по умолчанию устанавливается O4. O4 соответствует первой октаве.

L<длительность> - устанавливает стандартную длительность, т.е. длительность тех нот, для которых она не указана явно. Длительность задается от 0 до 64, а действительная длительность определяется величиной "1/<длительность>". При каждом запуске подпрограммы @PLAY устанавливается по умолчанию L4.

P<длительность> - специальная молчащая нота (пауза).

R задается в конце предложения, если необходимо запустить мелодию снова.

Предложение на ММЯ должно заканчиваться кодом 00H.

@PLOFF

Назначение: Подпрограмма @PLOFF предназначена для выключения всех трех каналов музыкального синтезатора.

Действие: Подпрограмма @PLOFF осуществляет прерывание звучания всех трех каналов музыкального синтезатора.

@PLOT

Назначение: Подпрограмма @PLOT предназначена для установки в цвет фона или текущий цвет рисования точки на экране и для перемещения графического курсора.

Входные: Регистры (B) и (C) должны содержать числовые значения, которые определяют координаты точки для установки графического курсора (соответственно по оси Y и по оси X);
регистр (A) указывает режим:
0 - стирание;
1 - рисование;
2 - установка графического курсора.

Действие: Подпрограмма @PLOT производит гашение или высвечивание точки на экране с координатами заданными в регистре (A), равен 1, то на экране точка высвечивается, 0 - гаснет.
Если режим равен 2, то происходит установка графического курсора в точку с координатами, заданными в регистрах (B) и (C), без каких-либо действий над точкой на экране.

@PUT

Назначение: Подпрограмма @PUT предназначена для вывода на экран изображения, запомненного подпрограммой @GET.

Входные: Регистры (DE) и (BC) должны содержать числовые значения, определяющие координаты точки (соответственно по оси Y и по оси X) на экране, куда будет выводиться изображение, запомненное подпрограммой @GET;
регистр (HL) должен содержать адрес оперативной памяти с изображением, помещенным подпрограммой @GET;
регистр (A) должен содержать режим:
0 - сброс точек;
1 - установка точек;
2 - точная копия.

Действие: Подпрограмма @PUT выводит на экран изображение, запомненное подпрограммой @GET. Регистры (DE) и (BC) должны содержать значения координат левого нижнего угла прямоугольника, размеры которого определены подпрограммой @GET при занесении графического изображения в оперативную память.
Регистр (A) должен содержать режим наложения изображений.
0 - сброс точек. На экран выводится изображение, цвет которого определяется цветом фона.
1 - установка точек. На экран выводится изображение с тем же цветом, с которым оно было занесено в оперативную память подпрограммой @GET.
2 - точная копия. На экран выводится копия прямоугольника, занесенного в оперативную память подпрограммой @GET.
Подпрограмма @PUT обычно применяется в сочетании с подпрограммой @GET. Однако, можно занести в область оперативной памяти изображение, не используя подпрограмму @GET, а затем вывести на экран это изображение подпрограммой @PUT. Для этого необходимо занести в область памяти точки изображения по следующим правилам:
- в левые два байта выделенной области занести размеры выводимого прямоугольника по оси X и по оси Y соответственно;
- в каждый следующий байт занести номера цветов двух точек (по 4 разряда на точку).
  Вывод точек на экран подпрограммой @PUT будет производиться слева направо и снизу вверх, начиная с текущего положения графического курсора.

@SCOLOR

Назначение: Подпрограмма @SCOLOR предназначена для установки физических цветов.

Входные: Регистры (DE) должны содержать адрес таблицы в оперативной памяти, содержащей 16 байт физических цветов.

Действие: Подпрограмма @SCOLOR присваивает значение физических цветов (из 256 цветов палитры) соответствующим номерам цветов. Значения физических цветов должны быть занесены в шестнадцатибайтовую таблицу, адрес которой задается в регистрах (DE). Номер цвета определяется относительным смещением от начала таблицы (начиная с нуля).
Например, первый байт таблицы соответствует номеру цвета 0, второй - номеру цвета 1 и т.д. Таким образом, значение физического цвета в первом байте таблицы соответствует номеру цвета 0, значение физического цвета во втором байте таблицы соответствует номеру цвета 1 и т.д.
Коды физических цветов определяют цвет в соответствии с таблицей 1:

Таблица 1

Перв. цвета	синий	зеленый	красный
Вес разряда	1/2 1/4	1/2 1/4 1/8	1/2 1/4 1/8
Код физич. цвета	128 64	32 16 8	4 2 1

Например, желтый цвет средней яркости получается из 1/2 красного и 1/2 зеленого цветов. Тогда код этого цвета равен 4+32=36.
При инициализации пакета подпрограмм устанавливается следующее соответствие номеров цветов и значений физических цветов (таблица 2):

Номер цвета	Код физ. (дес.)	Табл.цветн. (двоичн.)	Цвет
0	64	01000000	темно синий
1	128	10000000	синий
2	16	00010000	зеленый
3	208	11010000	голубой
4	6	00000110	красный
5	134	10000110	малиновый
6	22	00010110	кирпичный

Продолжение таблицы 2

Номер цвета	Код физ. (дес.)	Табл.цветн. (двоичн.)	Цвет
7	54	00110110	желтый
8	0	00000000	черный
9	197	11000101	фиолетовый
10	34	00100010	ярко-зеленый
11	192	11000000	ярко-синий
12	2	00000010	темно-красный
13	152	10011000	бирюзовый
14	82	01010010	серый
15	173	10101101	белый

@SCONR

Назначение: Подпрограмма @SCONR предназначена для установки скорости обмена информацией с магнитной лентой.

Входные: Регистр (A) должен содержать значение скорости обмена информацией с магнитной лентой в БОД/10.

Действие: Подпрограмма @SCONR производит установку скорости обмена информацией с магнитной лентой. Значение скорости задается в регистре (A).
При инициализации пакета устанавливается первоначальная скорость записи 1100 БОД.

@SCROL

Назначение: Подпрограмма @SCROL предназначена для перемещения экрана по вертикали.

Входные: Регистр (A) должен содержать число, определяющее величину смещения экрана по вертикали.

Действие: Подпрограмма @SCROL производит перемещение экрана по вертикали. Величина смещения по вертикали задается в регистре (A).
Если заданная величина смещения больше нуля, то экран перемещается по вертикали вверх.
Если заданная величина смещения меньше нуля, то экран перемещается по вертикали вниз.
Если заданная величина смещения равна нулю, то перемещение экрана не происходит.

@SLIST

Назначение: Подпрограмма @SLIST предназначена для установки типа принтера.

Входные: Регистр (A) должен содержать:
0 - тип принтера ROBOTRON-CM 6329;
1 - тип принтера EPSON FX-85, RAVI-8010M.

Действие: Подпрограмма @SLIST устанавливает тип принтера, заданный в регистре (A).

@SPIC

Назначение: Подпрограмма @SPIC предназначена для вывода на дисплей сообщения из области оперативной памяти.

Входные: Регистры (HL) должны содержать адрес выводимого на дисплей сообщения.

Действие: Подпрограмма @SPIC осуществляет вывод на дисплей сообщения, расположенного в оперативной памяти по адресу в регистрах (HL). Сообщение должно заканчиваться кодом 00H (признак конца сообщения)

@SPLAN

Назначение: Подпрограмма @SPLAN предназначена для установки (включения/выключения) экранных областей.

Входные: Регистр (A) должен содержать код обращения к экранным областям. Код должен принимать значения от 00H до 0FH.

Действие: Подпрограмма @SPLAN устанавливает режим обращения к экранным областям.
Если рассматривать код как двоичное число, то его можно представить в следующем виде:

```
X X X X
| | | |
| | | ------ управление областью 1
| | -------- управление областью 2
| ---------- управление областью 3
------------ управление областью 4
```

Причем 0 - запрещает обращение к области экрана;
1 - разрешает обращение к области экрана;
Таким образом, применяя подпрограмму @SPLAN, можно создавать до 4 независимых планов изображения.

@TIME

Назначение: Подпрограмма @TIME предназначена для определения или установки текущего времени.

Входные: Регистр (A) должен содержать 0 (число, отличное от нуля), если необходимо определить (установить) текущее время;
регистры (HL) должны содержать время, которое необходимо установить.

Выходные: Регистры (HL) содержат текущее время (содержимое регистра (A) равно нулю).

Действие: Для определения текущего времени необходимо в регистр (A) занести 0. Тогда в регистрах (HL) будет возвращено значение текущего времени.
Если в регистр (A) занести число, отличное от нуля, а в регистры (HL) значение времени, то подпрограмма @TIME установит время, значение которого было помещено в регистрах (HL).
Текущее время измеряется в "тиках" (через каждые двадцать миллисекунд текущее время увеличивается на единицу). Таким образом, 50 "тиков" определяют производительность времени в одну секунду. Поэтому определение действительного времени возложено на программу пользователя.

ПРИЛОЖЕНИЕ

ОБРАЗЕЦ НАЧАЛА ПРОГРАММЫ ПОЛЬЗОВАТЕЛЯ

ТЕКСТ ФРАГМЕНТА ПРОГРАММЫ, ОПРЕДЕЛЯЮЩЕГО АДРЕСА ПОДПРОГРАММ (ДРАЙВЕРОВ УСТРОЙСТВ)

```
;
        TITLE   'GAME V(1.1)'
;
        $-MACRO             ;упр. включ. в листинг макрорасширений
;
        ORG     2000H
;
LF      EQU     0AH         ;перевод строки
CR      EQU     0DH         ;возврат каретки
ESC     EQU     1BH         ;
BASE    EQU     120H        ;начальный адрес пакета подпрограмм
;                            (драйверов)
;
@INIT   EQU     BASE        ;инициализация работы пакета
@KEY    EQU     BASE+3      ;опрос состояния клавиатуры
@INTAP  EQU     BASE+6      ;ввод байта с магнитной ленты
@CONOUT EQU     BASE+9      ;вывод символа на дисплей
@OUTAP  EQU     BASE+12     ;вывод байта на магнитную ленту
@LIST   EQU     BASE+15     ;вывод символа на печать
@CONIN  EQU     BASE+16     ;ввод символа с клавиатуры
@DUMP   EQU     BASE+21     ;вывод содержимого регистра (A) в
;                            шестнадцатеричном виде.
@SPIC   EQU     BASE+24     ;вывод сообщения по адресу, определен-
;                            ному регистрами H и L
@INKEY  EQU     BASE+27     ;ввод символа с клавиатуры без ожидания
;                            ввода
@PLOT   EQU     BASE+30     ;рисование точки, установка графическо-
;                            го курсора
@LINE   EQU     BASE+33     ;рисование линий и прямоугольников,
;                            подготовка вывода символов увеличенно-
;                            го размера
@CIRCLE EQU     BASE+36     ;рисование окружности
@PCIRCLE EQU    BASE+39     ;установка параметров окружности
@GET    EQU     BASE+42     ;сохранение графической информации с
;                            экрана в оперативной памяти
@PUT    EQU     BASE+45     ;вывод на экран графической информации
;                            из области оперативной памяти
@PAINT  EQU     BASE+48     ;закрашивание замкнутых областей
@PGET   EQU     BASE+51     ;определение цвета текущей точки
@COLOR  EQU     BASE+54     ;установка цвета отображения
@FON    EQU     BASE+57     ;установка цвета экрана
@BORD   EQU     BASE+60     ;установка цвета бордюра (границы)
@SCROL  EQU     BASE+63     ;перемещение экрана по вертикали
@PLAY   EQU     BASE+66     ;программирование музыкального синтеза-
;                            тора
@FPLAY  EQU     BASE+69     ;определение звучания музыкальных
;                            каналов
@PLOFF  EQU     BASE+72     ;прерывание звучания музыкальных
;                            каналов
@TIME   EQU     BASE+75     ;определение/установка текущего времени

@MASC   EQU     BASE+78     ;установка маски вывода
```

```
@EADD    EQU      BASE+81     ;сложение двоичных чисел
@ESUB    EQU      BASE+84     ;вычитание двоичных чисел
@EMULT   EQU      BASE+87     ;умножение двоичных чисел
@EDIVM   EQU      BASE+90     ;дуление двоичных чисел
@ESIGN   EQU      BASE+93     ;определение знака двоичного числа
@DECBIN  EQU      BASE+96     ;преобразование десятичного числа,
;                              заданного в виде строки символов в
;                              коде КОИ-7, в двоичное число
@BINDEC  EQU      BASE+99     ;преобразование двоичного числа в
;                              десятичное в виде строки символов в
;                              коде КОИ-7
@CMPHD   EQU      BASE+102    ;сравнение двоичных чисел
@SCOLOR  EQU      BASE+105    ;установка физических цветов
@SPLAN   EQU      BASE+108    ;установка экранных областей
@SCONR   EQU      BASE+111    ;установка скорости обмена информацией
;                              с магнитной лентой
@SLIST   EQU      BASE+114    ;установка типа принтера
@LSCR    EQU      BASE+117    ;печать экрана в графическом режиме
;
         LXI      SP,7000H
BB       EQU      1
BF       EQU      2
BS       EQU      3
;
;*****************************
;*** ПРОГРАММА ПОЛЬЗОВАТЕЛЯ ***
;*****************************
;
         END
```