2.8. Команды видеопроцессора [89]

Fix Me!

> Кто не способен выдумывать небылицы, у того один выход - рассказывать были.
>
> *—Л.Вовенарг. Размышления и максимы*

[89]

2.8.1. Типы команд

Команды MSX VDP можно использовать для выполнения операторов LINE и PSET MSX BASIC, а также для переноса и копирования части изображения на экране дисплея.

Приведем вначале краткие сведения о командах.

Функция	***Область***		**Единица данных**	**Мнемоника**	**Четыре старших бита регистра номер 46**			
	Куда пересылаются данные (адресат)	**Откуда пересылаются данные (источник)**			**CM3**	**CM2**	**CM1**	**CM0**
Быстрая пересылка	VRAM	CPU	байт	HMMC	1	1	1	1
	VRAM	VRAM	байт	YMMM	1	1	1	0
	VRAM	VRAM	байт	HMMM	1	1	0	1
	VRAM	VDP	байт	HMMV	1	1	0	0
Пересылка с логическим преобразованием	VRAM	CPU	пиксель	LMMC	1	0	1	1
	CPU	VRAM	пиксель	LMCM	1	0	1	0
	VRAM	VRAM	пиксель	LMMM	1	0	0	1
	VRAM	VDP	пиксель	LMMV	1	0	0	0
Построение *линии*	VRAM	VDP	пиксель	LINE	0	1	1	1
Поиск	VRAM	VDP	пиксель	SRCH	0	1	1	0
Установка *точки*	VRAM	VDP	пиксель	PSET	0	1	0	1
Считывание точки	VDP	VRAM	пиксель	POINT	0	1	0	0
Команда отсутствует					0	0	1	1
					0	0	1	0
					0	0	0	1
СТОП (прерывание выполнения *всех* команд)				STOP	0	0	0	0

Перед выполнением команды в регистрах видеопроцессора (VDP) с номерами 32,33,34,...,44,45 Вам нужно задать необходимые *параметры*.

Команды выполняются видеопроцессором после задания кода команды в регистре видеопроцессора с номером 46, который называется *регистром команд* (регистр команд иногда обозначается CMR («CoMmand Register»). Это приводит к установке 1 в нулевом бите регистра *состояния* с номером 2 (имя данного бита - CE). Заметим, что после выполнения команды бит CE устанавливается в 0.

Для прерывания исполняющейся команды выполните команду STOP.

Команды выполняются только в графических режимах SCREEN 5, SCREEN 6, SCREEN 7, SCREEN 8!

Теперь остановимся на понятии *страница*...

Значениями параметров, используемых в командах видеопроцессора (VDP), являются *целые* числа, которые, чаще всего, представляют собой координаты X и Y точки на экране дисплея. Другими словами, VDP работает с областью видеопамяти (VRAM), задействованной в текущем режиме SCREEN.

Страницей будем называть определенный участок видеопамяти.

При отображении на экране видны 192 (или 212) линии текущей страницы (физическое начало изображения на экране устанавливается в соответствии с содержимым регистра видеопроцессора с номером 23). Выбор страницы, предназначенной для вывода на экран, осуществляется путем изменения базового адреса в регистре с номером 2. Содержимое отображаемой на экране дисплея страницы не влияет на выполнение команд видеопроцессора.

«Координатная сетка» и адреса расположения страниц для различных режимов SCREEN приведены ниже.

SCREEN 5		**Адрес**	**SCREEN 6**	
		00000h		
(0,0)	(255,0)		(0,0)	(511,0)
Страница 0			*Страница 0*	
(0,255)	(255,255)		(0,255)	(511,255)
		07FFFh		
		08000h		
(0,256)	(255,256)		(0,256)	(511,256)
Страница 1			*Страница 1*	
(0,511)	(255,511)		(0,511)	(511,511)
		0FFFFh		
		10000h		
(0,512)	(255,512)		(0,512)	(511,512)
Страница 2			*Страница 2*	
(0,767)	(255,767)		(0,767)	(511,767)
		17FFFh		
		18000h		
(0,768)	(255,768)		(0,768)	(511,768)
Страница 3			*Страница 3*	
(0,1023)	(255,1023)		(0,1023)	(511,1023)
		1FFFFh		

SCREEN 7		**Адрес**	**SCREEN 8**	
		00000h		
(0,0)	(511,0)		(0,0)	(255,0)
Страница 0			*Страница 0*	
(0,255)	(511,255)		(0,255)	(255,255)
		0FFFFh		
		10000h		

(0,256)	(511,256)		(0,256)	(255,256)
Страница 1			*Страница 1*	
(0,511)	(255,511)		(0,511)	(255,511)
		1FFFFh		

А теперь поговорим о *логических операциях* ...

Команды видеопроцессора

```
LINE, PSET, LMMC, LMCM, LMMM, LMMV
```

могут выполнять различные логические операции над *цветами*. Выполнение логических операций происходит при установке 4 *младших* битов (с именами LO3,LO2,LO1,LO0) регистра команд номер 46.

Приведем краткие данные о логических операциях. Учтите, что в таблице используются следующие обозначения:

- SC — код цвета *источника*;
- DC — код цвета *адресата*.

Имя	***Операция***	**LO3**	**LO2**	**LO1**	**LO0**
IMP	DC=SC	0	0	0	0
AND	DC=SC·DC	0	0	0	1
OR	DC=SC+DC	0	0	1	0
EOR	DC=NOT(SC)·DC+SC·NOT(DC)	0	0	1	1
NOT	DC=NOT(SC)	0	1	0	0
		0	1	0	1
		0	1	1	0
		0	1	1	1
TIMP	IF SC=0 THEN DC=DC ELSE DC=SC	1	0	0	0
TAND	IF SC=0 THEN DC=DC ELSE DC=SC·DC	1	0	0	1
TOR	IF SC=0 THEN DC=DC ELSE DC=SC+DC	1	0	1	0
TEOR	IF SC=0 THEN DC=DC ELSE DC=NOT(SC)·DC+SC·NOT(DC)	1	0	1	1
TNOT	IF SC=0 THEN DC=DC ELSE DC=NOT(SC)	1	1	0	1
		1	1	0	1
		1	1	1	0
		1	1	1	1

2.8.2. Состояние регистров после выполнения команд

После того, как некоторая команда VDP выполнена, состояние регистров будет таким, как указано в следующей таблице.

Мнемоника команды	***Номера регистров***									
	32,33	**34,35**	**36,37**	**38,39**	**40,41**	**42,43**	**44**	**46**		**45**
	SX	**SY**	**DX**	**DY**	**NX**	**NY**	**CLR**	**Старш. биты**	**Младш. биты**	**ARG**
HMMC	–	–	–	*	–	#	–	0	–	–
YMMM	–	*	–	*	–	#	–	0	–	–
HMMM	–	*	–	*	–	#	–	0	–	–
HMMV	–	–	–	*	–	#	–	0	–	–

Мнемоника команды	*Номера регистров*									
	32,33	**34,35**	**36,37**	**38,39**	**40,41**	**42,43**	**44**	**46**		**45**
	SX	**SY**	**DX**	**DY**	**NX**	**NY**	**CLR**	**Старш. биты**	**Младш. биты**	**ARG**
LMMC	−	−	−	∗	−	#	−	0	−	−
LMCM	−	∗	−	−	−	#	∗	0	−	−
LMMM	−	∗	−	∗	−	#	−	0	−	−
LMMV	−	−	−	∗	−	#	−	0	−	−
LINE	−	−	−	∗	−	−	−	0	−	−
SRCH	−	−	−	−	−	−	−	0	−	−
PSET	−	−	−	−	−	−	−	0	−	−
POINT	−	−	−	−	−	−	∗	0	−	−

В приведенной таблице:

- α) символ «−» означает, что содержимое регистра не изменяется;
- β) символ «∗» означает, что в регистре находится координата точки в момент завершения выполнения команды или код цвета;
- γ) символ «#» означает, что в регистре находится счетчик (NYB) количества обнаружений границы экрана во время выполнения команды.

Примечание.
Значениями для SY*, DY* и NYB служат точки (или байты в высокоскоростных пересылках данных), вычисляемые подстановкой N в соотношения,приведенные ниже:

```
SY∗=SY+N      DY∗=DY+N      (DIY=0)
SY∗=SY-N      DY∗=DY-N      (DIY=1)
NYB=NY-N
```

Для команд,осуществляющих высокоскоростную пересылку данных (HMMC,HMMM,HMMV,YMMM) N=2 в режимах SCREEN 5 и SCREEN 7, N=4 в режиме SCREEN 6.

Для команды LINE: если MAJ=0, то N=N-1.

Для других команд N=1.

2.8.3. Описание команд

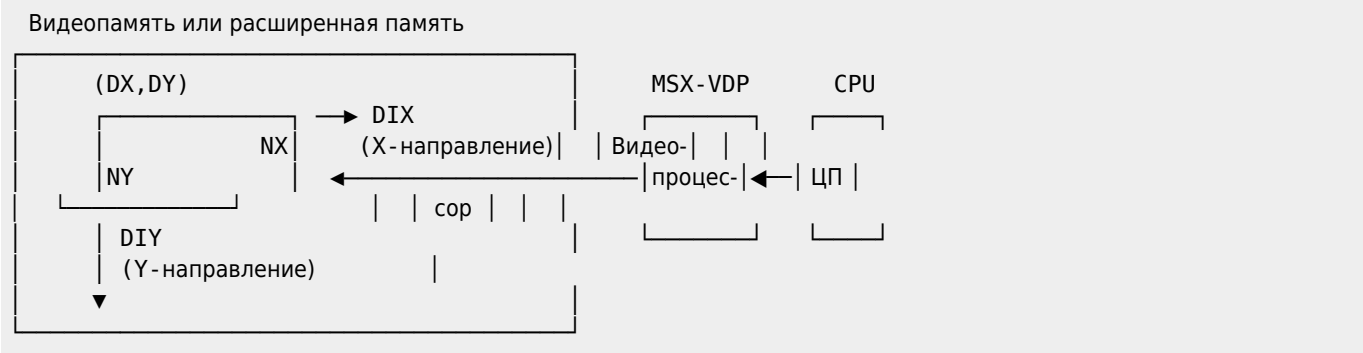
Брось свои иносказанья
И гипотезы пустые!
На проклятые вопросы
Дай ответы нам прямые.

—Г.Гейне

2.8.3.1. Команда HMMC

Команда HMMC пересылает данные от центрального процессора (CPU) в видеопамять (VRAM) или расширенную память в виде прямоугольного блока (размером NX×NY) через регистры видеопроцессора (VDP). Так как переносимые данные организованы побайтно, существует ограничение на величины NX и DX в соответствии с режимом отображения(в режимах SCREEN 5 и SCREEN 7 эти величины должны быть *четными*, а в режиме SCREEN 6 — *кратными* 4).

Взгляните на схему выполнения команды HMMC.



Опишем теперь установку регистров для выполнения команды HMMC и порядок ее выполнения.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:

- α) DX: базовая X-координата адресата (от 0 до 511);

  Отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 — 2 младших бита.

  DY: базовая Y–координата адресата (от 0 до 1023);

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 базовая X–координата адресата (младшая часть)
Регистр 37	7 6 5 4 3 2 1 0 0 0 0 0 0 0 0 DX8 базовая X–координата адресата
Регистр 38	7 6 5 4 3 2 1 0 DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 базовая Y–координата адресата (младшая часть)
Регистр 39	7 6 5 4 3 2 1 0 0 0 0 0 0 0 DY9 DY8 базовая Y–координата адресата (старшая часть)

- β) NX: «ширина» пересылаемого блока по X–направлению в точках (от 0 до 511);

  Отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 — 2 младших бита.

  NY: «ширина» пересылаемого блока по Y–направлению в точках (от 0 до 1023).

Регистр 40	Номера битов 7 6 5 4 3 2 1 0 NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 количество точек для пересылки по X–координате (младшая часть)
Регистр 41	0 0 0 0 0 0 0 NX8 количество точек для пересылки по X–координате (старшая часть)
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 количество точек для пересылки по Y–координате (младшая часть)

Регистр 43	0	0	0	0	0	0	NY9	NY8

количество точек для пересылки по Y–координате (старшая часть)

- γ)
  - MXD: задание области памяти адресата (0: видеопамять; 1: расширенная память);
  - DIX: направление для NX по X–координате адресата (0: направо; 1:налево);
  - DIY: направления для NY по Y–координате адресата (0:вниз; 1:вверх).

Регистр 45

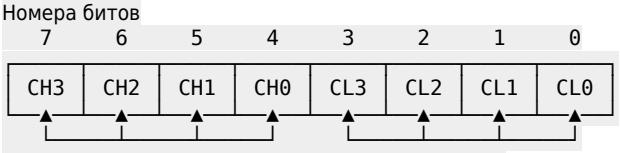
(ориентировка блока относительно точки с координатами (DX,DY))

- δ)

CLR: первый байт данных для переноса.

Для режимов SCREEN 5 и SCREEN 7

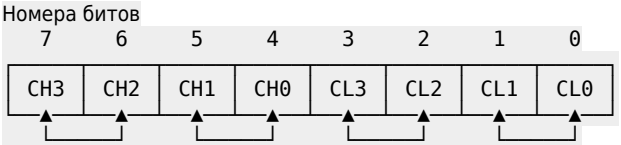
Регистр 44



N=0,1,2,...,127
регистр цвета

Для режима SCREEN 6

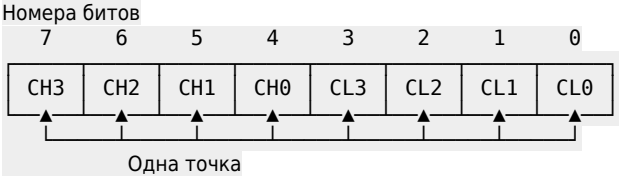
Регистр 44



N=0,1,2,...,127
регистр цвета

Для режима SCREEN 8

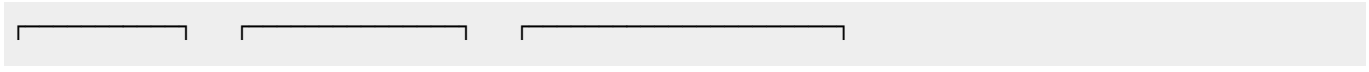
Регистр 44

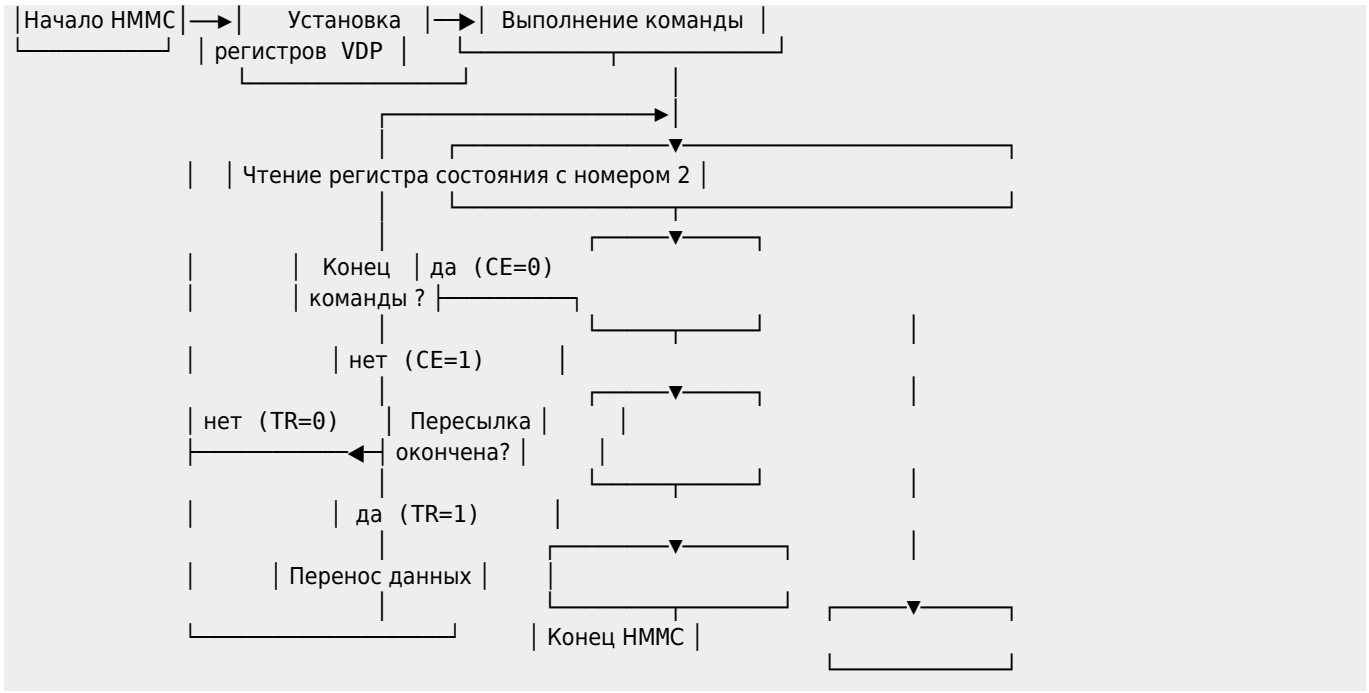


2. После задания значений параметров выполнение команды происходит заданием кода команды 11110000b в регистре команд с номером 46.

Второй и все последующие байты посылаются в регистр номер 44 после проверки TR-бита и CE-бита в регистре состояния с номером 2.

В заключение пункта приведем схему алгоритма выполнения команды HMMC:





Fix Me! *Пример*. Иллюстрация работы команды HMMC

208-01.bas

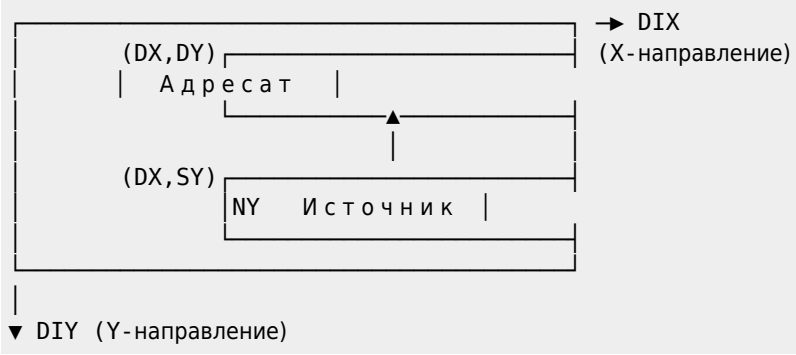
200-01.bas

```
10 DATA 3E,02,F7,87,31,01,32,00,E0,C9 :' Подпрограмма  в машинных кодах,
20 FOR I=0 TO 9:READ A$               :' позволяющая  прочесть  содержи-
30 POKE &HD000+I,VAL("&h"+A$):NEXT I  :' мое регистра статуса с номером
40 DEFUSR=&HD000                      :' 2  видеопроцессора
50 I=&HA000                           :' С этого адреса "берем" данные
60 SCREEN 8                           :' Возможны  SCREEN 5 ÷ SCREEN 8
61  VDP(36+1)=10:VDP(37+1)=0          :' X,Y-координаты блока
70  VDP(38+1)=10:VDP(39+1)=0          :' Длины сторон блока по осям  OX
80  VDP(40+1)=30:VDP(41+1)=0          :' и OY
81  VDP(42+1)=30:VDP(43+1)=0          :'
90  VDP(44+1)=PEEK(I)                 :' Цвет берем из  RAM
100 VDP(45+1)=0                       :' Ориентируем блок
110 VDP(46+1)=&B11110000              :' Выполняем команду  HMMC
120 A=USR(0):A=PEEK(&HE000)           :' Читаем 2-й регистр статуса◄┐┐
130 CE=(A AND &B00000001)             :' Выделяем бит  CE            ||
140 IF CE=0 THEN 190                  :' Если он равен 0, то ────────||┐
150 TR=(A AND &B10000000)             :' иначе выделяем бит  TR       |||
160 IF TR=0 THEN 120                  :' Если он равен 0, то ────────┘||
170 I=I+1:VDP(44+1)=PEEK(I)           :' иначе задаем новый цвет    ||
180 GOTO 120                          :' ──────────────────────────────┘|
190 A$=INPUT$(1)                      :' К о н е ц ◄──────────────────┘
```

2.8.3.2. Команда YMMM

> Тщательно пережевывайте пищу, этим вы помогаете обществу.
>
> *—И.Ильф и Е.Петров*

Команда YMMM переносит данные из области, определяемой величинами DX, SY, NY, DIX, DIY, и правой или левой «границей» видеопамяти в Y-направлении, задаваемом параметром DY.



Опишем порядок выполнения команды YMMM.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
    - α)
        - MXD: задать область памяти для адресата:
            - (0: видеопамять; 1: расширенная память);
        - DIX: направление по X-координате от точки источника в сторону правого или левого края экрана
            - (0: направо; 1: налево);
        - DIY: направление для NY
            - (0: вниз; 1: вверх) .

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 MXC MXD MXS DIY DIX EQ MAJ Направление пересылаемых данных (X) Выбор области памяти Направление пересылаемых данных (Y) (регистр аргумента)

    - β)
        - DX: базовая X-координата *источника* (от 0 до 511)
          (отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 - 2 младших бита);
        - SY: базовая Y-координата *источника* (от 0 до 1023).

Регистр 34	Номера битов 7 6 5 4 3 2 1 0 SY7 SY6 SY5 SY4 SY3 SY2 SY1 SY0 (базовая координата Y пересылаемого блока (младшая часть))
Регистр 35	0 0 0 0 0 0 SY9 SY8 (базовая координата Y пересылаемого блока (старшая часть))
Регистр 36	DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X пересылаемого блока (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X пересылаемого блока (старшая часть))

    - γ)
        - DY: базовая Y-координата *адресата* (от 0 до 1023)
          (отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 - 2 младших бита);
        - NY: «ширина» пересылаемого блока по Y-направлению в точках (от 0 до 1023).

Регистр 38	Номера битов 7 6 5 4 3 2 1 0 DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))

Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек по Y-координате (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек по Y-координате (старшая часть))

2. Выполнение команды происходит после помещения числа 11110000b в регистр команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 1 1 1 0 0 0 0 0 Код команды YMMM (регистр команды)

3. При выполнении команды YMMM CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Иллюстрация действия команды YMMM

208-02.bas

208-02.bas

```
10 SCREEN 8                     ' Возможны режимы  SCREEN 5 ÷ SCREEN 8
20 FOR I=15 TO 240 STEP 15      ' ┐ Цепочка разноцветных окружностей,  ко-
30   CIRCLE(I,15),15,I          ' │ торую мы будем копировать
40 NEXT                         ' ┘
50 A$=INPUT$(1)                 ' Подождем до нажатия любой клавиши...
60 VDP(34+1)=0: VDP(36+1)=0     ' Y,X  -  координаты пересылаемого блока
70 VDP(38+1)=181                ' Так как копирование ведется только в
80 :'                           ' Y-направлении, то указываем   Y-координа-
90 :'             'ту адресата  (X-координата остается преж-
95 :'             'ней)
100 VDP(42+1)=31                ' "Высота" (длина по оси  Y)  пересылаемого
105 :'             'блока
110 VDP(45+1)=&B00000000        '
111 :'      ▲▲▲     ' Пересылаем
120 :'      │ │└──────'н а п р а в о  и
130 :'      │ └──────'  в н и з
140 :'      └────────'Пересылаем во  VRAM
150 VDP(46+1)=&B11100000        ' К о д  команды  YMMM
170 A$=INPUT$(1)                ' К о н е ц
```

2.8.3.3. Команда HMMM

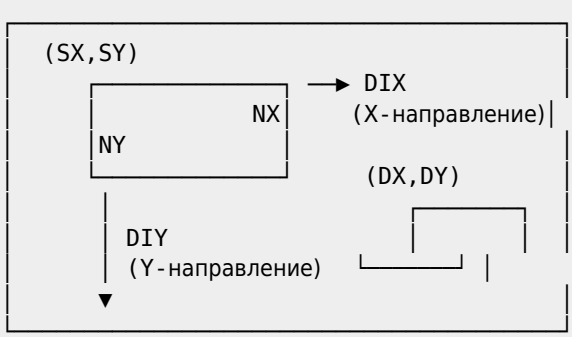
Fix Me!

> Стокгольм - город контрастов. Рядом с красивыми новыми домами стоят еще более новые и красивые.
>
> *—Н.Богословский*

Команда HMMM пересылает данные в виде специального прямоугольного блока из *видеопамяти* или *расширенной* памяти в *видеопамять* или *расширенную* память. Так как данные для пересылки организованы побайтно, существует

ограничение на значение параметра X в соответствии с режимом отображения.

Видеопамять или *расширенная память*



Опишем порядок выполнения команды HMMM.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
   - α)
     - MXS: Выбрать область памяти для *источника*;
     - MXD: Выбрать область памяти для *адресата*
       (0: видеопамять; 1: расширенная память);
     - DIX: Направления для NX от точки источника
       (0: направо; 1: налево);
     - DIY: Направления для NY от точки источника
       (0: вниз; 1: вверх).

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 MXC MXD MXS DIY DIX EQ MAJ Направление пересылаемых данных (X) Направление пересылаемых данных (Y) Выбор области памяти для источника Выбор области памяти для адресата (регистр аргумента)

   - β)
     - SX: X-координата *источника* (от 0 до 511)
       (отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6— 2 младших бита SX, DX и NX);
     - SY: Y-координата *источника* (от 0 до 1023).

Регистр 32	Номера битов 7 6 5 4 3 2 1 0 SX7 SX6 SX5 SX4 SX3 SX2 SX1 SX0 (базовая координата X источника (младшая часть))
Регистр 33	0 0 0 0 0 0 0 SX8 (базовая координата X источника (старшая часть))
Регистр 34	SY7 SY6 SY5 SY4 SY3 SY2 SY1 SY0 (базовая координата Y источника (младшая часть))
Регистр 35	0 0 0 0 0 0 SY9 SY8 (базовая координата Y источника (старшая часть))

   - γ)
     - DX: базовая X-координата *адресата* (от 0 до 511)
       (отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 - 2 младших бита SX, DX и NX);

- DY: базовая Y-координата *адресата* (от 0 до 1023).
  NX: «ширина» пересылаемого блока по X-направлению в точках (от 0 до 511) (отметим, что в режимах SCREEN 5 и SCREEN 7 теряется один младший бит, а в режиме SCREEN 6 - 2 младших бита SX, DX и NX);
- NY: «ширина» пересылаемого блока по Y-направлению в точках (от 0 до 1023).

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))
Регистр 40	NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек для пересылки в X-направлении (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек для пересылки в X-направлении (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек для пересылки в Y-направлении (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек для пересылки в Y-направлении (старшая часть))

2. Выполнение команды происходит заданием кода команды 11010000b в регистре команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 1 1 0 1 0 0 0 0 (регистр команды)

3. 3. При выполнении команды HMMM CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Пример использования команды HMMM
208-03.bas
208-03.bas

```
20 SCREEN 5:CIRCLE(40,40),40,8:PAINT STEP(0,0),8
30 FOR T=32 TO 46:VDP(T+1)=0:NEXT :'Очистка
40 VDP(32+1)=0:VDP(33+1)=0         :'Источник (координата X)
50 VDP(34+1)=0:VDP(35+1)=0         :'Источник (координата Y)
60 VDP(36+1)=80:VDP(37+1)=0        :'Приемник (координата X)
70 VDP(38+1)=80:VDP(39+1)=0        :'Приемник (координата Y)
80 VDP(40+1)=80:VDP(41+1)=0        :'Длина по координате X
90 VDP(42+1)=80:VDP(43+1)=0        :'Длина по координате Y
100 VDP(45+1)=&B00000000           :'Байт аргументов:
110 '       |||└──────────:' пересылать направо;
120 '       ||└───────────:' пересылать вниз;
130 '       |└────────────:' пересылать из VRAM;
140 '       └─────────────:' принимать во VRAM;
```

```
150 VDP(46+1)=&B11010000          :'Выполнение команды  HMMM
160 '           └▲┘               :'
180'      └─────────────── :'Код команды  HMMM
190 A$=INPUT$(1)                  :'К о н е ц
```

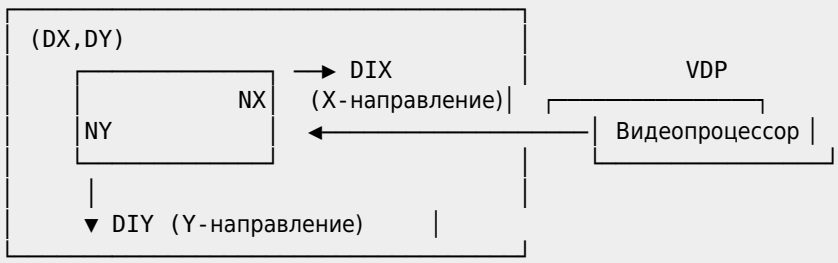
2.8.3.4. Команда HMMV

Каждая копировальная машина приводит
в негодность оригинал.

—Закон Мэрфи

Команда HMMV используется для *закраски* указанной прямоугольной области видеопамяти или расширенной памяти. Так как пересылаемые данные организованы побайтно, то существует ограничение на величину X в соответствии с режимом отображения.

Видеопамять или расширенная память



Опишем последовательность выполнения команды HMMV.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
    - α)
        - MXD: задать область памяти *адресата*
          (0: видеопамять; 1: расширенная память);
        - DIX: направление для NX от точки *источника*
          (0:направо; 1:налево);
        - DIY: направление для NY от точки *источника*
          (0:вниз; 1:вверх);

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 MXC MXD MXS DIY DIX EQ MAJ Направление пересылаемых данных (X) Направление пересылаемых данных (Y) Выбор области памяти для адресата (регистр аргумента)
Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))

Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))
Регистр 40	NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек для пересылки в X-направлении (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек дляпересылки в X-направлении (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек для пересылки в Y-направлении (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек для пересылки в Y-направлении (старшая часть))

- γ)
  - CLR: данные для *цветового кода.*

| *Для режимов* SCREEN 5 и SCREEN 7||

Регистр 44	Номера битов 7 6 5 4 3 2 1 0 CH3 CH2 CH1 CH0 CL3 CL2 CL1 CL0 X=2·N (четные точки) X=2·N+1 (нечетные точки) N=0,1,2,...,127 (регистр цвета)
Для режима SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 CH3 CH2 CH1 CH0 CL3 CL2 CL1 CL0 X=4·N X=4·N+1 X=4·N+2 X=4·N+3 N=0,1,2,...,127 Четные точки Нечетные точки (регистр цвета)
Для режима SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 CH3 CH2 CH1 CH0 CL3 CL2 CL1 CL0 О д н а т о ч к а (регистр цвета)

2. Выполнение команды происходит после помещения числа 11000000b в регистр команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 1 1 0 0 0 0 0 0 (регистр команд)

3. При выполнении команды HMMV CE-бит регистра состояния номер 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Иллюстрация работы команды HMMV
208-04.bas
208-04.bas

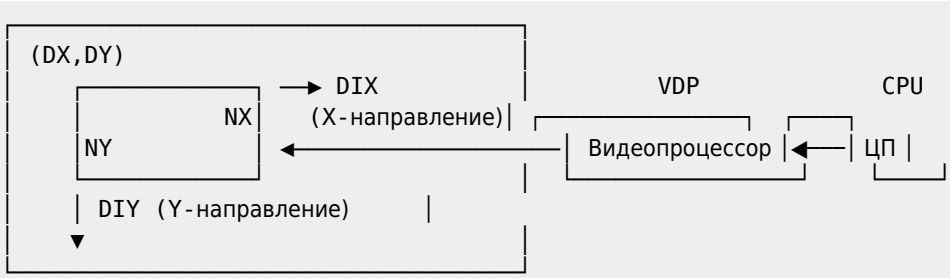
```
10 SCREEN 5                    ' Возможны режимы  SCREEN 5 ÷ SCREEN 8
30 VDP(36+1)=0: VDP(37+1)=0   ' X-координата прямоугольника
40 VDP(38+1)=0: VDP(39+1)=0   ' Y-координата прямоугольника
50 VDP(40+1)=25:VDP(41+1)=0   ' Длина по оси  X
60 VDP(42+1)=19:VDP(43+1)=0   ' Длина по оси  Y
70 VDP(44+1)=&B11011111   ' Чтобы прямоугольник был ровно окрашен, а не
71'      └▲┘└▲┘ 'был "полосатым",значение младшего полубайта
80'       │  │  'регистра цвета должно  быть  равно значению
85'       │  └──'старшего полубайта
90'       └─────'(в  SCREEN 8  это правило может не выполнять-
100'          'ся, так как в этом режиме 256 цветов)
120 VDP(45+1)=&b00000000 ' Ориентация прямоугольника  и  выбор области
121'          'видеопамяти
130 VDP(46+1)=&B11000000 ' Код команды  HMMV
140 A$=INPUT$(1)          ' К о н е ц
```

2.8.3.5. Команда LMMC

> Историю цивилизации можно выразить в шести словах:чем больше знаешь, тем больше можешь.
>
> *—Э.Абу*

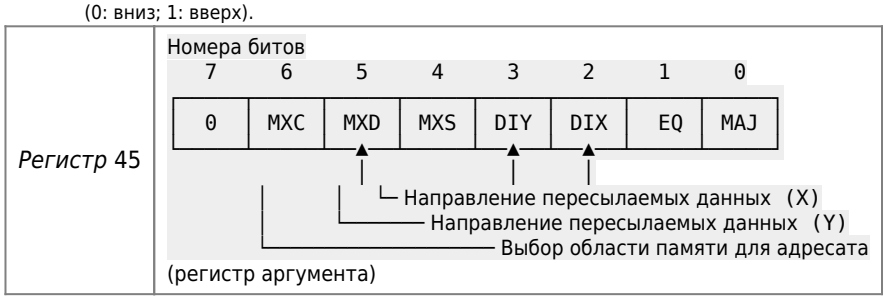
Команда LMMC передает данные из центрального процессора (CPU) в видеопамять или расширенную память в указанную прямоугольную область через видеопроцессор. Так как данные для переноса организованы поточечно, то над точками адресата могут быть выполнены логические операции.

Видеопамять или *расширенная память*



Опишем последовательность выполнения команды LMMC.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
    - α)
        - MXD: задать область памяти *адресата*
          (0: видеопамять; 1: расширенная память);
        - DIX: направление для NX от X-координаты адресата
          (0: направо; 1: налево);
        - DIY: направления для NY от Y-координаты адресата
          (0: вниз; 1: вверх).



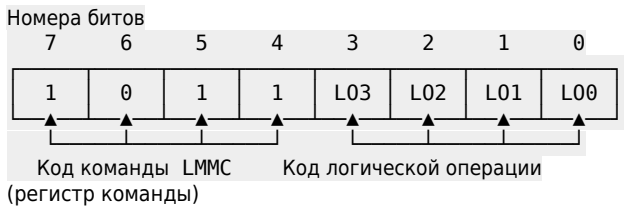
- β)
  - DX: Базовая X-координата *адресата* (от 0 до 511);
  - DY: Базовая Y-координата *адресата* (от 0 до 1023);
  - NX: «ширина» пересылаемого блока по X-направлению (от 0 до 511);
  - NY: «ширина» пересылаемого блока по Y-направлению (от 0 до 1023).

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))
Регистр 40	NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек для пересылки в X-направлении (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек для пересылки в X-направлении (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек для пересылки в Y-направлении (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек для пересылки в Y-направлении (старшая часть))

- γ) CLR: первый байт данных для пересылки.

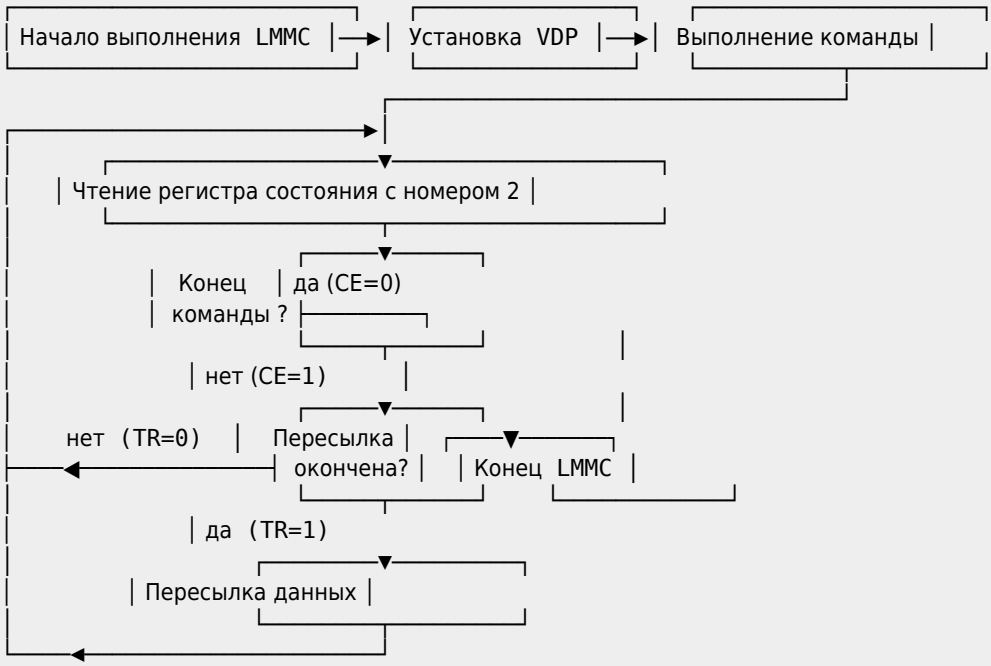
Режимы SCREEN 5 и SCREEN 7	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 CR3 CR2 CR1 CR0 (регистр цвета)
Режим SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 CR1 CR0 (регистр цвета)
Режим SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 CR7 CR6 CR5 CR4 CR3 CR2 CR1 CR0 (регистр цвета)

2. Выполнение команды LMMC происходит после помещения числа 1011b в 4 *старших* битах регистра команд с номером 46 и помещения кода логической операции в 4 *младших* битах регистра с номером 46.

Регистр 46	

3. Передавать второй байт и все последующие байты в 44-й регистр следует с постоянной проверкой значений битов TR и CE в регистре состояния с номером 2.

Приведем схему алгоритма выполнения команды LMMC.



Fix Me! *Пример*. Иллюстрация использования команды LMMC

208-05.bas

208-05.bas

```
10:'Команда LMMC отличается от команды HMMC только тем, что в команде
20:'LMMC можно использовать л о г и ч е с к и е о п е р а ц и и.
30 DATA 3E,02,F7,87,31,01,32,00,E0,C9 :' Подпрограмма в машинных кодах,
40 FOR I=0 TO 9:READ A$               :' позволяющая "прочесть" содер-
50 POKE &HD000+I,VAL("&h"+A$):NEXT I  :' жимое регистра статуса видео-
60 DEFUSR=&HD000: I=0                 :' процессора с номером 2
70 SCREEN 8                           :' Возможны SCREEN 5 ÷ SCREEN 8
80 VDP(36+1)=10:VDP(37+1)=0           :' X-координата блока
90 VDP(38+1)=10:VDP(39+1)=0           :' Y-координата блока
100 VDP(40+1)=30:VDP(41+1)=0          :' Длины сторон блока по осям OX
101 VDP(42+1)=30:VDP(43+1)=0          :' и OY
110 VDP(44+1)=PEEK(0)                 :' Цвет берем из RAM
120 VDP(45+1)=0                       :' Ориентируем блок
130 VDP(46+1)=&B10110100              :' Подаем команду LMMC
140 A=USR(0):A=PEEK(&HE000)           :' Читаем регистр стат. 2 ◄┐
150 CE=(A AND &B00000001)             :' Выделяем бит CE         │
160 IF CE=0 THEN 210                  :' Если он равен 0, то ────┤┐
170 TR=(A AND &B10000000)             :' иначе выделяем бит TR   ││
180 IF TR=0 THEN 140                  :' Если он равен 0, то ────┤│
190 I=I+1:VDP(44+1)=PEEK(I)           :' иначе задаем новый цвет ││
200 GOTO 140                          :' ────────────────────────┘│
210 A$=INPUT$(1)                      :' К о н е ц ◄──────────────┘
```

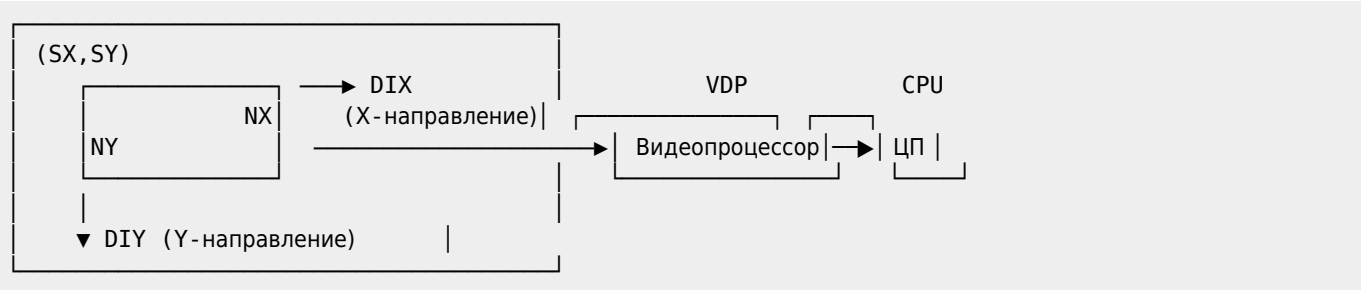
2.8.3.6 Команда LMCM

Когда я молод был, все тайны бытия,
Казалось, я раскрыл.
Ах, ошибался я!

—О.Хайям. Рубайят

Команда LMCM пересылает данные из видеопамяти или расширенной памяти в центральный процессор в виде заданной прямоугольной области (в X-Y координатах) через видеопроцессор. Данные пересылаются *поточечно*.

Видеопамять или *расширенная память*



Сейчас мы расскажем вам о порядке выполнения команды LMCM.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:

   - α)

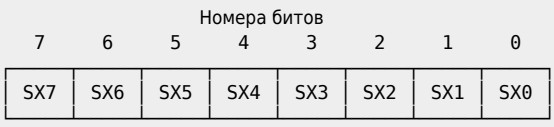
     - MXS: выбор области памяти для *источника*
       (0: видеопамять (VRAM); 1: расширенная память);
     - DIX: направление для NX от X-координаты точки источника
       (0: направо; 1: налево);
     - DIY: направление для NY от Y-координаты точки источника
       (0: вниз; 1: вверх);

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 | MXC | MXD | MXS | DIY | DIX | EQ | MAJ └ Направление пересылаемых данных (X) └ Направление пересылаемых данных (Y) └ Выбор области памяти для источника (регистр аргумента)

   - β)

     - SX: базовая X-координата точек *источника* (от 0 до 511);
     - SY: базовая Y-координата точек *источника* (от 0 до 1023).

   |*Регистр* 32|



   (базовая координата X источника (младшая часть))|

Регистр 33	0 | 0 | 0 | 0 | 0 | 0 | 0 | SX8 (базовая координата X источника (старшая часть))

Регистр 34	SY7 SY6 SY5 SY4 SY3 SY2 SY1 SY0 (базовая координата Y источника (младшая часть))
Регистр 35	0 0 0 0 0 0 SY9 SY8 (базовая координата Y источника (старшая часть))

- γ)
  - NX: «ширина» пересылаемого блока по X-направлению в точках (от 0 до 511);
  - NY: «ширина» пересылаемого блока по Y-направлению в точках (от 0 до 1023).

Регистр 40	Номера битов 7 6 5 4 3 2 1 0 NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек для пересылки в X-направлении (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек для пересылки в X-направлении (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек для пересылки в Y-направлении (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек для пересылки в Y-направлении (старшая часть))

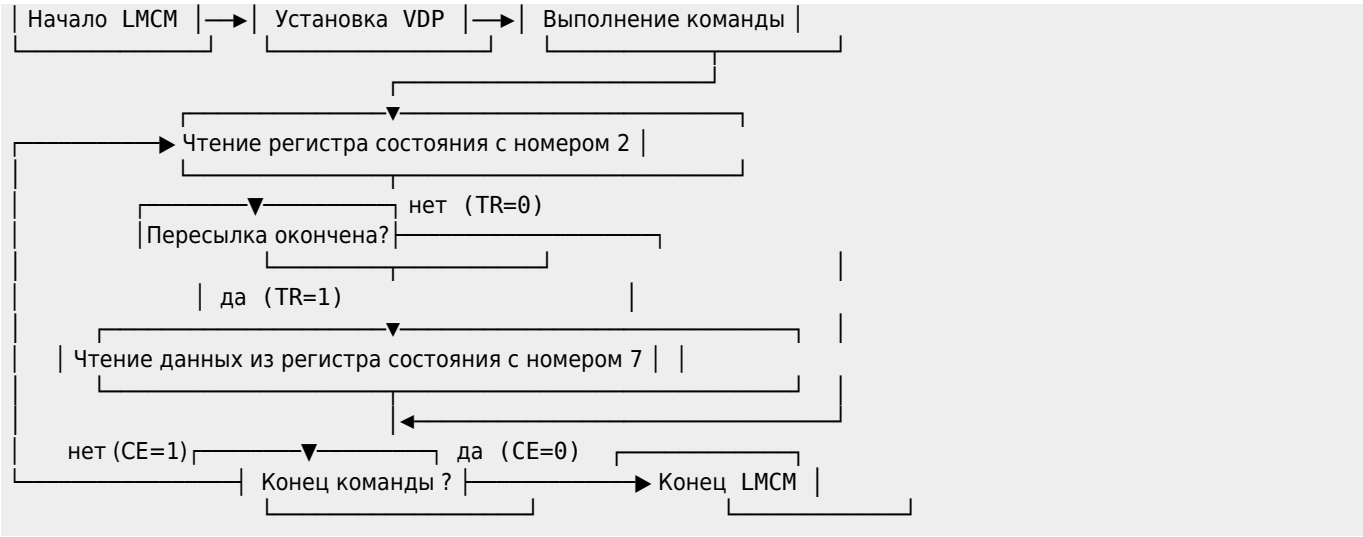
2. Выполнение команды LMCM происходит после помещения числа 1010b в 4 *старших* битах регистра команд с номером 46 и помещения кода логической операции в 4 *младших* битах регистра с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 1 0 1 0 0 0 0 0 Код команды LMCM Код логической операции (регистр команды)

3. Данные следует считывать из регистра состояния с номером 7, постоянно проверяя биты TR и CE в регистре состояния с номером 2.

Режимы SCREEN 5 и SCREEN 7	
Регистр статуса 7	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0
Режимы SCREEN 6	
Регистр статуса 7	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0
Режимы SCREEN 8	
Регистр статуса 7	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0

Приведем схему алгоритма выполнения команды LMCM.



Замечание 1.
TR-бит должен быть «сброшен» перед выполнением команды. Чтение регистра состояния с номером 7 происходит после настройки видеопроцессора.

Замечание 2.
Даже если данные установлены в регистре состояния с номером 7 и бит TR установлен в 1, то видеопроцессор выполнит команду и бит CE будет установлен в 0.

Fix Me! *Пример*. Иллюстрация работы команды LMCM
208-06.bas
208-06.bas

```
10 DATA 3E,00,F7,87,31,01,32,00,E0,C9:'Подпрограмма в кодах, "читающая"
20 FOR I=0 TO 9: READ A$              :'содержимое регистра статуса  ви-
30 POKE &HD000+I,VAL("&h"+A$):NEXT I  :'деопроцессора,номер которого на-
40 DEFUSR=&HD000                      :'ходится в ячейке  &hD001
50 I=&HA000                           :'Начальный адрес области памяти,
51 SCREEN 8                           :'где будем запоминать "картинку"
60 LINE (11,11)-(38,38),30,BF         :'  "К а р т и н к а"
61 VDP(32+1)=10:VDP(33+1)=0           :'X-координата запоминаемого блока
70 VDP(34+1)=10:VDP(35+1)=0           :'Y-координата запоминаемого блока
80 VDP(40+1)=30:VDP(41+1)=0           :'Длины сторон по осям  OX  и OY
90 VDP(42+1)=30:VDP(43+1)=0
100 VDP(45+1)=0                       :'Выбираем направление запоминания
110 VDP(46+1)=&B10100000              :'Код команды  LMCM
120 POKE &HD001,2:A=USR(0)            :'Читаем регистр статуса 2 и ◄┐
121 A=PEEK(&HE000)                    :'                            │
130 TR=(A AND &B10000000)             :'выделяем бит  TR.           │
140 IF TR=0 THEN 170                  :'Если он не равен 0, то ─────┐│
150 POKE &HD001,7:B=USR(0)            :'иначе читаем регистр стат.7 и│
160 POKE I,PEEK(&HE000):I=I+1         :'переносим данные             ||
170 CE=(A AND &B00000001)             :'Выделяем бит  CE  рег. стат. 2◄┘
180 IF CE=1 THEN 120                  :'Если он равен 1, то ──────────┘
190 END                               :'К о н е ц
```

2.8.3.7. Команда LMMM

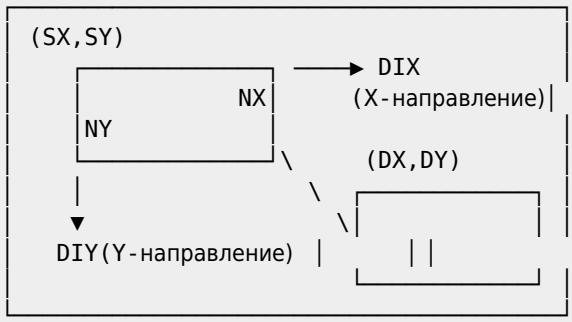
Даже если ваше объяснение настолько ясно,
что исключает всякое ложное толкование,
все равно найдется человек, который поймет
вас неправильно.

—Следствие из Третьего закона Чизхолма

Команда LMMM пересылает данные (заданную прямоугольную область) из *видеопамяти* или *расширенной*памяти в *видеопамять* или *расширенную* память.

Так как данные для пересылки организованы *поточечно*, то над данными адресата могут быть выполнены логические операции.

Видеопамять или *расширенная память*



Теперь опишем начальную установку регистров и порядок выполнения команды LMMM.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
   - α)
     - MXS: выбор области памяти для *источника*
       (0: видеопамять (VRAM); 1: расширенная память);
     - MXD: выбор области памяти для *адресата*
       (0: видеопамять (VRAM); 1: расширенная память);
     - DIX: направление для NX от X-координаты точки источника
       (0: направо; 1: налево);
     - DIY: направление для NY от Y-координаты точки источника
       (0: вниз; 1: вверх).

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 | MXC | MXD | MXS | DIY | DIX | EQ | MAJ Направление пересылаемых данных (X) Направление пересылаемых данных (Y) Выбор области памяти для источника Выбор области памяти для адресата (регистр аргумента)

   - β)
     - SX: X-координата точки *источника* (от 0 до 511);
     - SY: Y-координата точки *источника* (от 0 до 1023);
     - DX: X-координата точки *адресата* (от 0 до 511);
     - DY: Y-координата точки *адресата* (от 0 до 1023).

Регистр 32	Номера битов 7 6 5 4 3 2 1 0 X7 | X6 | X5 | X4 | X3 | X2 | X1 | X0 (базовая координата X источника (младшая часть))
Регистр 33	0 | 0 | 0 | 0 | 0 | 0 | 0 | SX8 (базовая координата X источника (старшая часть))
Регистр 34	SY7 | SY6 | SY5 | SY4 | SY3 | SY2 | SY1 | SY0 (базовая координата Y источника (младшая часть))

Регистр 35	0 0 0 0 0 0 SY9 SY8 (базовая координата Y источника (старшая часть))
Регистр 36	DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))

- γ)
  - NX: «*ширина*» пересылаемого блока по X-направлению в точках (от 0 до 511);
  - NY: «*ширина*» пересылаемого блока по Y-направлению в точках (от 0 до 1023).

Регистр 40	Номера битов 7 6 5 4 3 2 1 0 NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек по X координате (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек по X координате (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек по Y координате (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек по Y координате (старшая часть))

2. Выполнение команды начинается после помещения числа 1001b в четырех *старших* битах и кода логической операции - в четырех *младших* битах регистра с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 1 0 0 1 LO3 LO2 LO1 LO0 Код команды LMMM Код логической операции (регистр команды)

3. При выполнении команды LMMM CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Иллюстрация работы команды LMMM

208-07.bas

208-07.bas

```
10 SCREEN 8                ' Возможны режимы  SCREEN 5 ÷ SCREEN 8
20 LINE(10,10)-(50,50),255 ' Копируемый объект
30 VDP(32+1)=10            ' X-координата и с т о ч н и к а
40 VDP(34+1)=10            ' Y-координата и с т о ч н и к а
50 VDP(36+1)=55            ' X'-координата а д р е с а т а
60 VDP(38+1)=55            ' Y'-координата а д р е с а т а
```

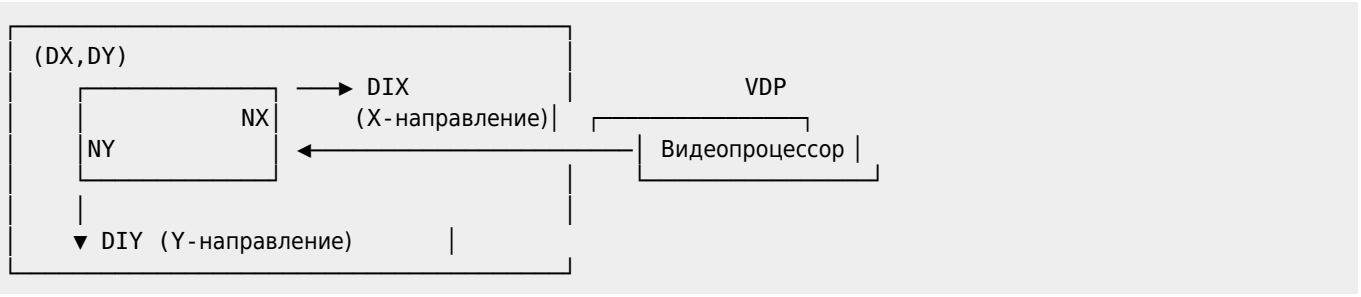
```
70 VDP(40+1)=40             ' Длина копируемого блока по оси  X
80 VDP(42+1)=40             ' Длина копируемого блока по оси  Y
90 VDP(45+1)=&B00000000     ' 0  желательно ставить...
91 :            ' В этом  регистре может находиться  любое
92 :            ' число, но при этом может  быть  нарушена
93 :            ' ориентация адресата  относительно  точки
94 :                        ' (X',Y')  или копирования не произойдет
100 A$=INPUT$(1)            ' П о д о ж д е м ...
110 VDP(46+1)=&B10010000    ' Выполнение команды  LMMM
120 A$=INPUT$(1)            ' П о д о ж д е м ...
```

2.8.3.8. Команда LMMV

> Король прусский неоднократно приказывал составлять заведомо негодные топографические планы разных местностей. На них указывалось, например, что такое-то болото непроходимо, и неприятель, полагаясь на карту, верил тому, чего на самом деле не было.
>
> *—С.Шамфор. Характеры и анекдоты*

Команда LMMV закрашивает заданную прямоугольную область видеопамяти или расширенной памяти цветом с указанным кодом. Так как данные для переноса организованы *поточечно*, то над данными адресата могут быть выполнены логические операции.

Видеопамять или *расширенная память*



Теперь опишем начальную установку регистров и порядок выполнения команды LMMV.

1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
   - α)
     - MXD: выбор области памяти для *адресата*
       (0: видеопамять (VRAM); 1: расширенная память);
     - DIX: направление для NX от X-координаты точки источника:
       (0: направо; 1: налево);
     - DIY: направление для NY от Y-координаты точки источника:
       (0: вниз; 1: вверх).

	Номера битов 7 6 5 4 3 2 1 0
Регистр 45	0 | MXC | MXD | MXS | DIY | DIX | EQ | MAJ Направление пересылаемых данных (X) Направление пересылаемых данных (Y) Выбор области памяти для адресата (регистр аргумента)

   - β)

- DX: базовая X-координата *адресата* (от 0 до 511);
- DY: базовая Y-координата *адресата* (от 0 до 1023).

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))

- γ)
  - NX: «*ширина*» пересылаемого блока по X-направлению в точках (от 0 до 511);
  - NY: «*ширина*» пересылаемого блока по Y-направлению в точках (от 0 до 1023).

Регистр 40	Номера битов 7 6 5 4 3 2 1 0 NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек по X координате (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек по X координате (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек по Y координате (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек по Y координате (старшая часть))

- б) CLR: данные о *цветовом коде*.

Режимы SCREEN 5 и SCREEN 7	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0 (регистр цвета)
Режимы SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0 (регистр цвета)
Режимы SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0 (регистр цвета)

2. Команда выполняется после помещения числа 1000b в четыре *старших* бита и кода логической операции в четыре

младших бита регистра команд с номером 46.

3. При выполнении команды LMMV CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Иллюстрация работы команды LMMV. Рисунок, состоящий из вложенных квадратов.
208-08.bas
208-08.bas Иллюстрация работы команды LMMV. Рисунок, состоящий из вложенных квадратов.

```
60  COLOR 15,0,0
70  SCREEN 8                                :'Возможны  SCREEN 5÷SCREEN 8
80  FOR I=0 TO 63
100  DL=128-I*2+1                           :'Вычисление длины стороны
110  VDP(36+1)=I: VDP(37+1)=0               :'Координаты прямоугольника
115  VDP(38+1)=I: VDP(39+1)=0               :'
120  VDP(40+1)=DL:VDP(41+1)=0               :'Длины сторон по осям  X  и Y
125  VDP(42+1)=DL:VDP(43+1)=0
130  VDP(44+1)=I                            :'Цвет прямоугольника
140  VDP(45+1)=&B00000000                   :'Ориентировка прямоугольника
141 :'      ▲▲▲          :'относительно точки  (X,Y):
142 :'      │ │└──────────────────:' н а п р а в о (1:налево)
143 :'      │ └──────────────────:' в н и з     (1:вверх)
144 :'      └──────────────────:'Выбираем  VRAM    (1:ERAM)
150  VDP(46+1)=&B10000000                   :'
151 :'           └▲┘ └▲┘                   :'
152 :'      │  └──────────:'Код логической операции
153 :'      └──────────:'Код команды  LMMV
154 A$=INPUT$(1)                            :'Очень важная задержка!
160 NEXT I
170 A$=INPUT$(1)                            :'К о н е ц
```

Если координата или длина стороны блока занимает один регистр (байт), то второй регистр, отвечающий за ту же координату (длину стороны) блока, нужно обнулить, иначе координата или длина стороны блока может оказаться настолько большой, что Ваш блок может не поместиться на экране !

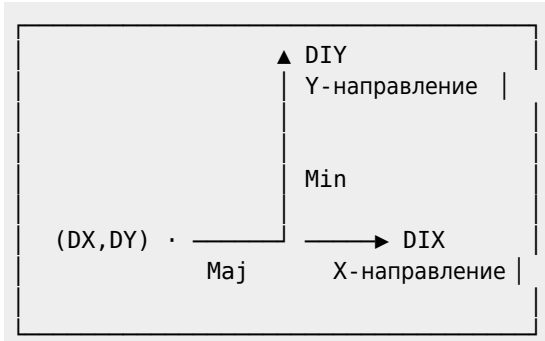
Если длины сторон прямоугольника равны 0, то он «растекается» в заданном направлении до конца экрана! Более того, если Ваш прямоугольник выходит за границы экрана и по X-, и по Y-координате, то видеопроцессор заполняет данными для прямоугольника участки Таблиц SGT, SCT и SAT и на экране появляются спрайты различных цветов и конфигураций.

2.8.3.9. Команда LINE

> Прямая — наидлиннейшее расстояние
> между двумя точками.
>
> *—Закон Мэрфи*

С помощью команды LINE можно начертить на экране отрезок прямой линии, используя данные, находящиеся в видеопамяти или расширенной памяти. Начерченный отрезок образуется как гипотенуза прямоугольного треугольника с заданными катетами (длинным и коротким). Величины катетов задаются расстояниями от одной и той же точки.

Видеопамять или *расширенная память*



Теперь опишем начальную установку регистров и порядок выполнения команды LINE.

1. 1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
   - α)
     - MXS: выбор области памяти для *адресата*
       (0: видеопамять (VRAM); 1: расширенная память);
     - DIX: направление от исходной точки до конечной точки \\(0: направо; 1: налево);
     - DIY: направление от исходной точки до конечной точки \\(0: вниз; 1: вверх);
     - MAJ: направление для *длинного* катета
       (0: по оси X; 1: по оси Y).

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 MXC MXD MXS DIY DIX EQ MAJ Выбор направления для длинного катета Направление пересылаемых данных (X) Направление пересылаемых данных (Y) Выбор области памяти для адресата (регистр аргумента)

   - β)
     - Maj: количество точек в *длинном* катете (от 0 до 1023);
     - Min: количество точек в *коротком* катете (от 0 до 511).

Регистр 40	Номера битов 7 6 5 4 3 2 1 0 MJ7 MJ6 MJ5 MJ4 MJ3 MJ2 MJ1 Maj (длина длинного катета вдоль оси X (младшая часть))
Регистр 41	0 0 0 0 0 0 MJ9 MJ8 (длина длинного катета вдоль оси X (старшая часть))
Регистр 42	MI7 MI6 MI5 MI4 MI3 MI2 MI1 MI0 (длина короткого катета вдоль оси Y (младшая часть))
Регистр 43	0 0 0 0 0 0 0 MI8 (длина короткого катета вдоль оси Y (старшая часть))

   - δ)
     - DX: базовая X-координата (от 0 до 511);
     - DY: базовая Y-координата (от 0 до 1023).

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X адресата (младшая часть))

Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X адресата (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y адресата (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y адресата (старшая часть))

- ω) CLR: данные о *цветовом коде*.

Режимы SCREEN 5 и SCREEN 7	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0 (регистр цвета)
Режимы SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0 (регистр цвета)
Режимы SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0 (регистр цвета)

2. 2. Выполнение команды LINE начинается после помещения числа 0111b в четыре *старших* бита и кода логической операции в четыре *младших* бита регистра команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 0 1 1 1 LO3 LO2 LO1 LO0 Код команды LINE Код логической операции (регистр команды)

3. 3. После выполнения команды LINE CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0.

Fix Me! *Пример*. Иллюстрация выполнения команды LINE

208-09.bas

208-09.bas

```
20 SCREEN 5
30 FOR T=32 TO 46:VDP(T+1)=0:NEXT :'Очистка регистров команд
60 VDP(36+1)=128:VDP(37+1)=0       :'Координата базовой точки  X
70 VDP(38+1)=106:VDP(39+1)=0       :'Координата базовой точки  Y
75 'Длинный катет обязан быть  д л и н н е е  короткого!
80 VDP(40+1)=50 :VDP(41+1)=0       :'Длинный катет
90 VDP(42+1)=10 :VDP(43+1)=0       :'Короткий катет
95 VDP(44+1)=8                     :'Цвет линии
100 VDP(45+1)=&B00000000           :'Байт аргументов:
110 '       |||└──────:'  длинный катет - по оси  X
120 '       ||└──────────:'  рисовать  н а п р а в о
130 '       |└──────────:'  рисовать  в н и з
```

```
140 '      └──────────────:'  рисовать  в о   VRAM
150 VDP(46+1)=&B01110000           :'Выполнение команды  LINE
160 '           └▲─┘ └▲─┘          :'
170 '      │  └──────────:'Код логической операции
180 '      └─────────────:'Код команды  LINE
190 A$=INPUT$(1)                   :'К о н е ц
```

2.8.3.10. Команда SRCH

> То, что вы ищете, вы найдете в самом последнем месте. Надо было посмотреть там с самого начала.
>
> *—Закон Мэрфи*

Команда SCRH проводит поиск цвета границы в видеопамяти или расширенной памяти «вправо» или «влево» от базовой точки.

Видеопамять или *расширенная память*

```
┌──────────────────────────────────────────┐
│ (SX,SY)                                  │
│           X-направление поиска           │
│         . ──────────────────────▶. DIX   │
│ Базовая точка    Точка цвета границы     │
└──────────────────────────────────────────┘
```

А теперь мы расскажем Вам об установке регистров и порядке выполнения команды SRCH.

1. 1. Вначале устанавливаются необходимые параметры в регистры команд VDP, а именно:
    - α)
        - MXD: выбор области памяти для поиска (0: видеопамять (VRAM); 1: расширенная память);
        - DIX: направление для поиска точки источника: (0: направо; 1: налево);
        - EQ: при 0 выполнение команды заканчивается,когда *найден* граничный цвет; при 1 выполнение заканчивается, когда цвет отличается от граничного.

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 0 MXC MXD MXS DIY DIX EQ MAJ └ Выбор направления для длинного катета └ Условие окончания команды └ Направление пересылаемых данных (X) └ Выбор области памяти для адресата (регистр аргумента)

    - β)
        - SX: базовые X-координаты для поиска (от 0 до 511);
        - SY: базовые Y-координаты для поиска (от 0 до 1023).

Регистр 32	Номера битов 7 6 5 4 3 2 1 0 SX7 SX6 SX5 SX4 SX3 SX2 SX1 SX0 (базовая координата X для поиска (младшая часть))

Регистр 33	0 0 0 0 0 0 0 SX8 (базовая координата X для поиска (старшая часть))
Регистр 34	SY3 SY2 SY1 SY0 SY3 SY2 SY1 SY0 (базовая координата Y для поиска (младшая часть))
Регистр 35	0 0 0 0 0 0 SY9 SY8 (базовая координата Y для поиска (старшая часть))

- γ) CLR: данные о *коде цвета границы* для поиска.

Режимы SCREEN 5 и SCREEN 7	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0 (регистр цвета)
Режим SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0 (регистр цвета)
Режим SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0 (регистр цвета)

2. 2. Выполнение команды осуществляется помещением кода 01100000b в регистр команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 0 1 1 0 0 0 0 0 Код команды SRCH (регистр команды)

3. 3. При выполнении команды CE-бит регистра состояния с номером 2 будет установлен в 1, а после выполнения - в 0. Кроме того, местоположение най- денного цвета границы (X-координата) окажется в регистрах статуса 8 и 9.

Регистр *статуса* 2	Номера битов 7 6 5 4 3 2 1 0 BD CE Устанавливается в 0 по окончании команды SRCH Устанавливается в 1, если н а й д е н цвет границы
Регистр *статуса* 8	Номера битов 7 6 5 4 3 2 1 0 BX7 BX6 BX5 BX4 BX3 BX2 BX1 BX0 (местоположение найденного цвета границы (младшая часть X-координаты))

Регистр *статуса* 9	Номера битов 7 6 5 4 3 2 1 0 1 1 1 1 1 1 BX9 BX8 (местоположение найденного цвета границы (старшая часть X-координаты))

Приведем схему алгоритма выполнения команды SRCH.

Fix Me! *Пример*. Изображение на экране *белой* линии. Далее, видеопроцессор ищет X–координату этой линии

208-10.bas

208-10.bas

```
5 'Программу составил Беленький Г. (IX класс)
10 SCREEN 7                                 :' Возможны SCREEN 5÷SCREEN 8
20 LINE (254,0)-(254,121),15                :' Граница для поиска
30 VDP(32+1)=0: VDP(33+1)=0                 :' X-координата начальн. точки
40 VDP(34+1)=0: VDP(34+1)=0                 :' Y-координата начальн. точки
50 VDP(44+1)=15                             :' Ищем б е л у ю точку
60 VDP(45+1)=&B00000000                     :'
70 '      ||└──────────────────:' Выполнение закончится,когда
71 '      ||          :' будет найдена белая точка
80 '      |└──────────────────:' Ищем направо от нач. точки
90 '      └──────────────────:' Ищем во VRAM
100 VDP(46+1)=&B01100000                    :' Выполнение команды SRCH
110 VDP(15+1)=2:A=INP(&H99):VDP(15+1)=0     :' Читаем регистр ст.2; ◄┐
120 CE=(A AND &B00000001)                   :' Выделяем бит CE       │
130 IF CE=1 THEN 110                        :' Если он равен 1, то ──┘
140 BD=(A AND &B00010000)                   :'   выделяем бит BD
150 IF BD=0 THEN 190                        :' Если он равен 0, то ──┐
160 VDP(15+1)=8:A=INP(&H99):VDP(15+1)=0     :'   читаем регистр ст.8 и │
170 OPEN "GRP:" AS #1                       :'   печатаем X-координату,│
180 PRESET(0,0):PRINT #1,A                  :'   на которой найдена   │
181 '                 :' белая точка      │
190 A$=INPUT$(1)                            :' К о н е ц ◄──────────┘
```

2.8.3.11. Команда PSET

Слово только оболочка,
Пленка, звук пустой, но в нем
Бьется розовая точка
Странным светится огнем.

—Арс.Тарковский

Команда PSET устанавливает точку определенного цвета в видеопамяти или расширенной памяти. При этом над уже находящейся в указанных координатах точкой производится логическая операция.

Видеопамять или *расширенная память*

```
┌──────────────────────────┐
│       (DX,DY)            │
│          .               │
└──────────────────────────┘
```

А теперь мы расскажем Вам об установке регистров и порядке выполнения команды PSET.

1. Сначала устанавливаются необходимые параметры в регистры команд VDP, а именно:

   - α) MXD: выбор области памяти
     (0: видеопамять (VRAM); 1: расширенная память).

Регистр 45	Номера битов 7 6 5 4 3 2 1 0 | | | MXD | | | | | └──── Выбор области памяти (регистр аргумента)

   - β)
     - DX: X-координата точки (от 0 до 511);
     - DY: Y-координата точки (от 0 до 1023).

Регистр 36	Номера битов 7 6 5 4 3 2 1 0 DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (базовая координата X точки (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (базовая координата X точки (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (базовая координата Y точки (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (базовая координата Y точки (старшая часть))

   - γ) CLR: данные о *коде цвета точки*.

Режимы SCREEN 5 и SCREEN 7	

Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0 (регистр цвета)
Режимы SCREEN 6	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0 (регистр цвета)
Режимы SCREEN 8	
Регистр 44	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0 (регистр цвета)

2. Выполнение команды осуществляется заданием числа 0101b в четырех *старших* битах и кода логической операции в четырех *младших* битах регистра команд с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 0 1 0 1 LO3 LO2 LO1 LO0 Код команды PSET Код логической операции (регистр команды)

3. При выполнении команды PSET CE-бит регистра состояния с номером 2 устанавливается в 1, а после выполнения - в 0.

Fix Me! *Пример 1(а).*. Программа установки *белой* точки при помощи команды видеопроцессора PSET.
208-11.bas
208-11.bas

```
5' Программу составил Беленький Г. (IX класс)
10 COLOR 15,1,1:SCREEN 7      ' Возможны режимы SCREEN 5 ÷ SCREEN 8
20 VDP(36+1)=254              ' X-координата
30 VDP(38+1)=110              ' Y-координата
40 VDP(44+1)=15               ' Цвет
50 VDP(45+1)=0                ' Если Вы хотите увидеть результат своего
70 :            ' труда, то 0 обязателен!
90 VDP(46+1)=&B01010000       ' Код команды PSET
110 A$=INPUT$(1)              ' К о н е ц
```

Fix Me! *Пример 1(б).*.
208-11.asm Эта же программа, написанная на макроассемблере имеет следующий вид:

```
VDP MACRO @A,@B                 ;Макрос, который помещает число  @B  в
          LD   B,@B             ;в регистр видеопроцессора с номером @A
          LD   C,#99            ;
          OUT  (C),B            ;
          LD   A,@A             ;
          OR   #80              ;
          OUT  (C),A            ;
    ENDM                        ;
          LD   A,7              ;Установка режима
          DEFB #F7,#87,#D1,0    ;SCREEN 7
          VDP  36,255           ;X-координата точки
          VDP  38,128           ;Y-координата точки
          VDP  44,8             ;Цвет
          VDP  45,0             ;
```

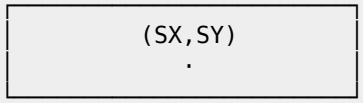
```
        VDP  46,%01010000       ;Код команды видеопроцессора  PSET
        DEFB #F7,0,#9F,0        ;Задержка
        DEFB #F7,#87,#D5,0      ;Установка режима  SCREEN 0
        RET                     ;Возврат в  MSX-DOS
```

2.8.3.12. Команда POINT

> Черное - цвет? Цвет. Белое - цвет? Цвет.
> Кто сказал, что у меня нет цветного телевизора?
>
> *—Философия владельца черно-белого телевизора*

Команда POINT считывает цвет указанной точки из видеопамяти или расширенной памяти.

Видеопамять или *расширенная память*



А теперь мы расскажем Вам об установке регистров и порядке выполнения команды POINT.

1. Вначале устанавливаются необходимые параметры в регистр команд VDP, а именно:
   - α) MXD: выбор области памяти (0: видеопамять (VRAM); 1: расширенная память).

	Номера битов							
	7	6	5	4	3	2	1	0
Регистр 45			MXS					
			▲					
			└──── Выбор области памяти					
(регистр аргумента)								

   - β)
     - SX: X-координата точки (от 0 до 511);
     - SY: Y-координата точки (от 0 до 1023).

	Номера битов							
	7	6	5	4	3	2	1	0
Регистр 32	SX7	SX6	SX5	SX4	SX3	SX2	SX1	SX0
(базовая координата X точки (младшая часть))								
Регистр 33	0	0	0	0	0	0	0	SX8
(базовая координата X точки (старшая часть))								
Регистр 34	SY3	SY2	SY1	SY0	SY3	SY2	SY1	SY0
(базовая координата Y точки (младшая часть))								
Регистр 35	0	0	0	0	0	0	SY9	SY8
(базовая координата Y точки (старшая часть))								

2. Выполнение команды осуществляется заданием кода 01000000b в регистре с номером 46.

Регистр 46	Номера битов 7 6 5 4 3 2 1 0 0 1 0 0 0 0 0 0 Код команды PSET (регистр команды)

3. При выполнении видеопроцессором команды POINT CE-бит регистра состояния с номером 2 устанавливается в 1, а после выполнения - в 0.

   Кроме того, найденный цветовой код помещается в регистр статуса 7.

Регистр *статуса* 2	Номера битов 7 6 5 4 3 2 1 0 CE Устанавливается в 0 по окончанию команды SRCH
Режимы SCREEN 5 и SCREEN 7	
Регистр *статуса* 7	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 C3 C2 C1 C0 (цветовой код)
Режимы SCREEN 6	
Регистр *статуса* 7	Номера битов 7 6 5 4 3 2 1 0 0 0 0 0 0 0 C1 C0 (цветовой код)
Режимы SCREEN 8	
Регистр *статуса* 7	Номера битов 7 6 5 4 3 2 1 0 C7 C6 C5 C4 C3 C2 C1 C0 (цветовой код)

Fix Me! *Пример*. Определение цвета заданной точки

208-12.bas

208-12.bas

```
5' Пример составил Беленький Г. (IX класс).
10  COLOR 15,1,1:SCREEN 7  ' Возможны режимы SCREEN 5 ÷ SCREEN 8
20  VDP(36+1)=254          ' X-координата
30  VDP(38+1)=110          ' Y-координата
40  VDP(44+1)=15           ' Цвет
50  VDP(45+1)=0            ' Если Вы хотите увидеть результат
60 '          'своих трудов, то ставьте 0  |
70  VDP(46+1)=&B01010000   ' Код команды PSET              |
80  VDP(32+1)=254          ' X'-координата        |
90  VDP(34+1)=110          ' Y'-координата        |
100 VDP(45+1)=&B00000000   ' <-------------------------------┘
110 VDP(46+1)=&B01000000   ' Код команды POINT
120 DATA 3E,07,F7,87,31,01,32,00,E0,C9 ' Считываем регистр статуса 7
130 '       ▲▲                                                  |
140 '       └┘--------------------------------------------------┘
150 FOR I=0 TO 9: READ A$               '
160 POKE &HD000+I,VAL("&h"+A$): NEXT I  '
170 DEFUSR=&HD000:A=USR(0)              '
180 OPEN "GRP:" AS #1: PRESET(0,0),0    ' Печатаем
190 PRINT #1, PEEK(&HE000)              ' код цвета
```

```
200 A$=INPUT$(1)
```

Характеристики купленного прибора соответствуют
спецификации довольно долго,все время,
пока продавец демонстрирует его Вам.

—Закон Мэрфи

Fix Me! *Пример*. Демонстрация действия *всех* команд видеопроцесора
208-13.bas
208-13.bas

```
5 ' Программу составил Беленький Г. (IX класс)
10 DATA 01,00,00,11,00,00,3E,00,32,B3,FC  :' Эта подпрограмма в  машин-
20 DATA 32,B5,FC,32,F2,F3,F7,87,85,00,C9  :' ных кодах  моделирует опе-
30 FOR I=0 TO 21:READ A$                  :' ратор LINE (0,0)-(0,0),0 ,
40   POKE &HD000+I,VAL("&h"+A$)           :' который обнуляет неиспольз-
50 NEXT I                                 :' уемые регистры  видеопро-
60 DEFUSR=&HD000                          :' цессора.
70  COLOR 15,0,0: SCREEN 8
80  FOR I=0 TO 63
90  DL=128-I*2+1
100 Z=USR(0)                      :'┐
110 VDP(36+1)=I:VDP(38+1)=I       :' ├─ LMMV (закрашивает прямоугольную
120 VDP(40+1)=DL:VDP(42+1)=DL     :' │  область с логическим преобразова-
130 VDP(44+1)=I:VDP(45+1)=0       :' │  нием цвета)
140 VDP(46+1)=&B10000100          :'┘            │
150 NEXT I '          ▲           :'             │
160 :'                └──────────────────────────┘
170 VDP(36+1)=0:VDP(38+1)=0               :'┐
180 VDP(44+1)=255                         :' ├─ PSET (установка точки)
190 VDP(45+1)=0:VDP(46+1)=&B01010000      :'┘
200 A$=INPUT$(1)
210 VDP(32+1)=0:VDP(34+1)=0               :'┐
220 VDP(36+1)=129:VDP(38+1)=0             :' │
230 VDP(40+1)=129:VDP(42+1)=129           :' ├─ HMMM (быстрая пересылка)
240 VDP(45+1)=0                           :' │
250 VDP(46+1)=&B11010000                  :'┘
260 A$=INPUT$(1)
270 VDP(32+1)=130:VDP(34+1)=0             :'┐
280 VDP(36+1)=0:VDP(38+1)=0               :' │
290 VDP(40+1)=129:VDP(42+1)=129           :' ├─ LMMM(пересылка с логичес-
300 VDP(45+1)=0                           :' │       ким преобразованием)
310 VDP(46+1)=&B10010100                  :'┘               │
320 A$=INPUT$(1):'    └──────────────────────────────────────┘
330 FOR I=0 TO 255
340 VDP(36+1)=I:VDP(38+1)=0               :'┐
350 VDP(40+1)=128:VDP(42+1)=0             :' │
360 VDP(44+1)=I                           :' ├─ LINE (линия)
370 VDP(45+1)=&B00000001                  :' │
380 VDP(46+1)=&B01110100                  :'┘
390 NEXT I
400 A$=INPUT$(1)
410 VDP(34+1)=0:VDP(36+1)=0               :'┐
420 VDP(38+1)=129:VDP(42+1)=83            :' ├─ YMMM(быстрая пересылка толь-
430 VDP(45+1)=0                           :' │       ко в  Y-направлении)
440 VDP(46+1)=&B11100000                  :'┘
450 A$=INPUT$(1)
460 Z=USR(0)
470 VDP(36+1)=0:VDP(38+1)=0               :'┐
480 VDP(40+1)=255:VDP(42+1)=64            :' ├─ HMMV (закраска прямоуголь-
```

```
490 VDP(44+1)=0:VDP(45+1)=0           :' │         ной области без  ло-
500 VDP(46+1)=&B11000000              :'└─        гических операций)
510 OPEN "GRP:" AS #1: PRESET(0,0)
520 PRINT #1," А теперь найдем, на какой  X-координате находится линия с цветом  128."
530 DATA 3E,00,F7,87,31,01,32,00,F0,C9     :'─┐
540 FOR I=0 TO 9:READ A$                   :' │
550 POKE &HE000+I,VAL("&h"+A$):NEXT I      :' │
560 DEFUSR1=&HE000                         :' │
570 VDP(32+1)=0:VDP(34+1)=128              :' │
580 VDP(44+1)=128:VDP(45+1)=0              :' │
590 VDP(46+1)=&B01100000                   :' ├─ SRCH  (поиск точки
600 POKE &HE001,2:A=USR1(0):A=PEEK(&HF000) :' │  заданного цвета)
610 CE=(A AND &B00000001)                  :' │
620 IF CE=1 THEN 600                       :' │
630 BD=(A AND &B00010000)                  :' │
640 IF BD=0 THEN 670                       :' │
650 POKE &HE001,8:A=USR1(0):A=PEEK(&HF000) :'└─
660 PRESET (100,30):PRINT #1, A;"                      255-128=127 (!)"
670 A$=INPUT$(1)
680 CLS
690 PRESET (0,0): PRINT #1," Поставим точку ";
700 VDP(36+1)=128:VDP(38+1)=106            :'─┐
710 VDP(44+1)=128:VDP(45+1)=0              :' ├─ PSET  (ставит точку)
720 VDP(46+1)=&B01010000                   :'─┘
730 PRINT #1,"и определим ее   цвет."
740 VDP(32+1)=128:VDP(34+1)=106            :'─┐
750 VDP(45+1)=0:VDP(46+1)=&B01000000       :' ├─ POINT(определяет цвет
760 POKE &HE001,7:Z=USR1(0):A=PEEK(&HF000) :'─┘         заданной точки)
770 PRINT #1," Цвет точки  :",A
780 A$=INPUT$(1)
790 CLS
800 PSET (0,0): PRINT #1,"Нарисуем  ";CHR$(34);"картинку";CHR$(34);"."
810 Z=USR(0)
820 VDP(36+1)=110:VDP(38+1)=90
830 VDP(40+1)=30:VDP(42+1)=30
840 VDP(44+1)=255:VDP(45+1)=0
850 VDP(46+1)=&B10000000
860 Z=USR(0)
870 VDP(36+1)=115:VDP(38+1)=95
880 VDP(40+1)=20:VDP(42+1)=20
890 VDP(44+1)=7:VDP(45+1)=0
900 VDP(46+1)=&B10000000
910 PRINT #1,"И запомним ее в  RAM  начиная с   адреса  &hA800."
920 PRINT #1,"Подождите ! Процесс длится около  д в у х  минут."
930 I=&HA800
940 VDP(32+1)=110:VDP(34+1)=90
950 VDP(40+1)=30:VDP(42+1)=30
960 VDP(45+1)=0:VDP(46+1)=&B10100000
970 POKE &HE001,2:A=USR1(0):A=PEEK(&HF000)
980 TR=(A AND &B10000000)
990 IF TR=0 THEN 1020
1000 POKE &HE001,7:B=USR1(0):B=PEEK(&HF000)
1010 POKE I,B:I=I+1
1020 CE=(A AND &B00000001)
1030 IF CE=1 THEN 970
1040 PRINT #1,"Запомнили ..."
1050 A$=INPUT$(1)
1060 CLS
1070 PRESET (0,0):PRINT #1,"Очистили экран."
1080 PRINT#1,"Теперь восстановим рисунок из памяти в нормальном виде."
1090 I=&HA801
1100 VDP(36+1)=110:VDP(38+1)=90
1110 VDP(40+1)=30:VDP(42+1)=30
1120 VDP(44+1)=256+NOT(PEEK(&HA800))
```

```
1130 VDP(45+1)=0:VDP(46+1)=&B11110000
1140 POKE &HE001,2:A=USR1(0):A=PEEK(&HF000)
1150 CE=(A AND &B00000001)
1160 IF CE=0 THEN 1210
1170 TR=(A AND &B10000000)
1180 IF TR=0 THEN 1140
1190 I=I+1:VDP(44+1)=PEEK(I)
1200 GOTO 1140
1210 A$=INPUT$(1)
1220 PRINT #1,"А теперь в инвертированном ..."
1230 I=&HA801
1240 VDP(36+1)=110:VDP(38+1)=90
1250 VDP(40+1)=30:VDP(42+1)=30
1260 VDP(44+1)=256+NOT(PEEK(&HA800)) 'Маленькая хитрость!
1270 VDP(45+1)=0:VDP(46+1)=&B10110100
1280 POKE &HE001,2:A=USR1(0):A=PEEK(&HF000)
1290 CE=(A AND &B00000001)
1300 IF CE=0 THEN 1350
1310 TR=(A AND &B10000000)
1320 IF TR=0 THEN 1280
1330 I=I+1:VDP(44+1)=PEEK(I)
1340 GOTO 1280
1350 A$=INPUT$(1)
```

В заключение поговорим вкратце о возможностях ускорения выполнения команд видеопроцессора.

Выполнение команд видеопроцессора можно ускорить *двумя* следующими способами.

1. Запрещение отображения спрайтов («гашение» спрайтов)

   Если бит 1 регистра с номером 8(SPD) установлен в 1, то время, расходуемое на обработку спрайтов может быть использовано для выполнения команд.

2. Запрещение отображения экрана («гашение» экрана).

   Если бит 6 регистра видеопроцессора с номером 1(BL) установлен в 0, то время, затрачиваемое на отображение содержимого экрана, может быть использовано для выполнения команд.

Даже самые светлые в мире умы
Не смогли разогнать окружающей тьмы.
Рассказали нам несколько сказочек на ночь-
И отправились мудрые спать, как и мы.

—О.Хайям. Рубайят

2.8.4. Работа с "мышью" и световым пером

«Мышь»

Ниже мы объясним функции «мыши» видеопроцессора. Так как «мышь» использует цветовую шину видеопроцессора, то при работе с мышью *нельзя* воспользоваться цветовой шиной видеопроцессора с любой другой целью!

Напомним Вам, что для использования «мыши» следует установить бит 7 регистра с номером 8 в 1, а бит 6 в 0.

Регистр 8	Номера битов 7 6 5 4 3 2 1 0 MS LP TP CB VR 0 SP BW 0: световое перо о т к л ю ч е н о 1: "мышь" в к л ю ч е н а

Когда бит 7 регистра с номером 8 установлен в 1, то направление цветовой шины автоматически устанавливается на ввод. Можно узнать, нажата ли кнопка «мыши», читая регистр состояния с номером 1.

Регистр статуса номер 1	Номера битов 7 6 5 4 3 2 1 0 FL | LPS | Идентификация | F11 | SX8 1: луч развертки обнаружен; 0: луч развертки не обнаружен; 1: переключатель 2 нажат; 0: переключатель 2 не нажат;

Относительные координаты движения «мыши» устанавливаются в двоичном дополнительном коде в регистрах состояния с номерами 3 и 5.

Регистр статуса номер 3	Номера битов 7 6 5 4 3 2 1 0 X7 | X6 | X5 | X4 | X3 | X2 | X1 | X0 (регистр столбца)
Регистр статуса номер 5	Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 (регистр строки)

Когда в регистре с номером 15 устанавливаются 3 или 5, то отсчет «мыши» не происходит. Когда читаются регистры состояния с номерами 3 и 5 или когда отсчет начинается, содержимое регистра с номером 15 должно быть изменено.

Световое перо

Для работы со световым пером следует установить бит 7 регистра с номером 8 в 0 и бит 6 в 1.

Регистр 8	Номера битов 7 6 5 4 3 2 1 0 MS | LP | TP | CB | VR | 0 | SP | BW 1: световое перо включено 0: "мышь" отключена

Для получения прерывания во время работы светового пера следует установить бит 5 регистра с номером 0 в 1. Прерывание сбрасывается, если прочитан регистр состояния с номером 1.

Регистр статуса номер 1	Номера битов 7 6 5 4 3 2 1 0 FL | LPS | Идентификация | F11 | SX8 1: переключатель нажат; 0: переключатель не нажат; 1: луч развертки обнаружен; 0: луч развертки не обнаружен;

Координаты, при которых световое перо обнаружило луч развертки, фиксируются в регистрах состояния с номерами 3,4,5,6. Данные, установленные в этих регистрах, верны до тех пор, пока не считывается регистр состояния с номером 5.

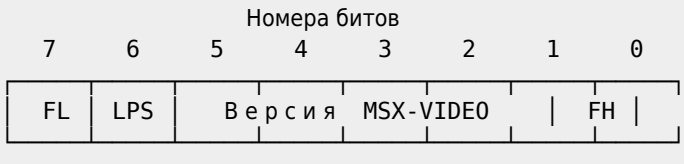
Регистр 3	Номера битов 7 6 5 4 3 2 1 0 X7 | X6 | X5 | X4 | X3 | X2 | X1 | X0 (регистр столбца (младшие биты))
Регистр 4	1 | 1 | 1 | 1 | 1 | 1 | 1 | X8 (регистр столбца (старшие биты))

Регистр 5	Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 (регистр строки (младшие биты))
Регистр 6	1 | 1 | 1 | 1 | 1 | 1 | Y9 | Y8 Этот бит указывает номер поля при чтении: 0: 1-е поле; 1: 2-е поле (регистр строки (старшие биты))

Приведем блок-схему использования светового пера:

2.8.5. Регистры статуса и регистры команд

Регистр статуса с номером 1.



Кратко расскажем Вам о назначении каждого бита.

- FL: флаг светового пера (флаг светового пера установлен). Если световым пером определены координаты точки на экране, то этот бит (FL),также как и бит IE2, должен быть установлен для возможности прерывания.

  Обычно при считывании регистра статуса с номером 1 значение бита FL сбрасывается. Если же второй переключатель «мыши» нажат, то при считывании регистра статуса с номером 1 значение бита FL не сбрасывается;

- LPS: переключатель светового пера (установка флага светового пера). Нажат переключатель светового пера. В этом случае при считывании регистра статуса с номером 1 значение бита LPS не сбрасывается. Переключатель «мыши» 1 (флаг «мыши» установлен). Первый переключатель «мыши» нажат. В этом случае при считывании регистра статуса с номером 1 значение бита LPS не сбрасывается.
- FH: флаг прерывания от горизонтального сканирования (который устанавливается в регистре с номером 19). Если бит IE1 установлен, то возможно прерывание. При считывании регистра статуса с номером 1 значение бита FH сбрасывается.

Регистр статуса с номером 2.

```
7  6  5  4  3  2  1  0 Номера битов
```

TR	VR	HR	BD	1	1	EO	CE

- TR: флаг готовности пересылки. Когда центральный процессор посылает команды в видеопамять и другие устройства, он проверяет этот флаг во время передачи данных. Передача может быть осуществлена, если этот флаг установлен в 1.
- VR: флаг синхронизации вертикального сканирования. Во время вертикального сканирования этот флаг установлен в 1.
- HR: флаг синхронизации горизонтального сканирования. Во время горизонтального сканирования этот флаг установлен в 1.
- BD: флаг обнаружения цвета границы. При выполнении команды поиска (одна из команд видеопроцессора) этот флаг определяет, был ли обнаружен цвет границы.
- EO: флаг поля отображения:
  - 0 - отображается первое поле.
  - 1 - отображается второе поле.

* CE: флаг выполнения команды. Он указывает, что в настоящее время выполняется команда видеопроцессора.

Регистры статуса с номерами 3, 4, 5, 6.

Эти регистры предназначены для указания координат столкновения спрайтов, координат местонахождения светового пера и координат относительного смещения «мыши».

Заметим, что при считывании содержимого регистра статуса с номером 5 содержимое регистров статуса с номерами 3,4,5 сбрасывается.

	Номера битов 7	6	5	4	3	2	1	0
Регистр статуса 3	X7	X6	X5	X4	X3	X2	X1	X0
Регистр статуса 4	1	1	1	1	1	1	1	X8
Регистр статуса 5	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
Регистр статуса 6	1	1	1	1	1	1	1	Y8

Значения, содержащиеся в регистрах состояния с номерами 3,4,5 будут определять координаты смещения в соответствии с формулами:

- X(содержимое регистра статуса 5, содержимое регистра статуса 3)
- Y(содержимое регистра статуса 5, содержимое регистра статуса 5).

Координаты столкновения спрайтов легко находятся по формулам:

```
XC = X/12   ,    YC = Y/8
```

Регистр статуса номер 7 (регистр цвета).

Этот регистр используется в том случае, когда выполняется команда видеопроцессора POINT или команды видеопроцессора типа «VRAM - to CPU». Данные из VRAM размещаются в этом регистре.

Регистры статуса с номерами 8 и 9.

В эти регистры помещается значение координаты X в случае,когда выполняется команда поиска SRCH (см. Приложение 2, раздел 8.3.10) и обнаружен цвет бордюра.

Регистр статуса 8	Номера битов 7 6 5 4 3 2 1 0 BX7 BX6 BX5 BX4 BX3 BX2 BX1 BX0 (координата X границы (младшая часть))
Регистр статуса 9	1 1 1 1 1 1 1 BX8 (координата X границы (старшая часть))

Регистры команд

Следующие регистры команд используются при выполнении команд видеопроцессора. Работа с ними описана в Приложении 2, раздел 8.3 .

Регистр 32	Номера битов 7 6 5 4 3 2 1 0 X7 X6 X5 X4 X3 X2 X1 X0 (регистр исходной координаты X (младшая часть))
Регистр 33	0 0 0 0 0 0 0 SX8 (регистр исходной координаты X (старшая часть))
Регистр 34	SY7 SY6 SY5 SY4 SY3 SY2 SY1 SY0 (регистр исходной координаты Y (младшая часть))
Регистр 35	0 0 0 0 0 0 SY9 SY8 (регистр исходной координаты Y (старшая часть))
Регистр 36	DX7 DX6 DX5 DX4 DX3 DX2 DX1 DX0 (регистр конечной координаты X (младшая часть))
Регистр 37	0 0 0 0 0 0 0 DX8 (регистр конечной координаты X (старшая часть))
Регистр 38	DY7 DY6 DY5 DY4 DY3 DY2 DY1 DY0 (регистр конечной координаты Y (младшая часть))
Регистр 39	0 0 0 0 0 0 DY9 DY8 (регистр конечной координаты Y (старшая часть))
Регистр 40	NX7 NX6 NX5 NX4 NX3 NX2 NX1 NX0 (количество точек по X координате (младшая часть))
Регистр 41	0 0 0 0 0 0 0 NX8 (количество точек по X координате (старшая часть))
Регистр 42	NY7 NY6 NY5 NY4 NY3 NY2 NY1 NY0 (количество точек по Y координате (младшая часть))
Регистр 43	0 0 0 0 0 0 NY9 NY8 (количество точек по Y координате (старшая часть))

Регистр 44	CH3 | CH2 | CH1 | CH0 | CL3 | CL2 | CL1 | CL0 (регистр цвета)
Регистр 45	0 | MXC | MXD | MXS | DIY | DIX | EQ | MAJ (регистр аргумента)
Регистр 46	CM3 | CM2 | CM1 | CM0 | LO3 | LO2 | LO1 | LO0 (регистр команды)

---

https://sysadminmosaic.ru/msx/basic_dialogue_programming_language/208

2023-02-19 21:22