EMOS

L1 MOS

BASIC Language
Graphics Handling
User Guide

olivetti

PREFACE

This manual describes those BASIC statements and functions which are graphic oriented. It is a complement to the "BASIC Language User Guide", the reading of which is recommended.

The manual is intended as a user guide for programmers and analysts working on BASIC program developement in the scientific technical field.

SUMMARY

The manual is divided in two main parts:

- First Part examines the most salient graphic features available in the system.
- Second part gives a detailed description of each graphic statement and function, which are presented in alphabetical order.

REFERENCES

Read first...

Introduction to MOS
Code 4002130 G (vol. 2)

BASIC Language User Guide
Code 4002340 D

For further information, read...

MOS - EDITOR Reference Manual
Code 4002440 P

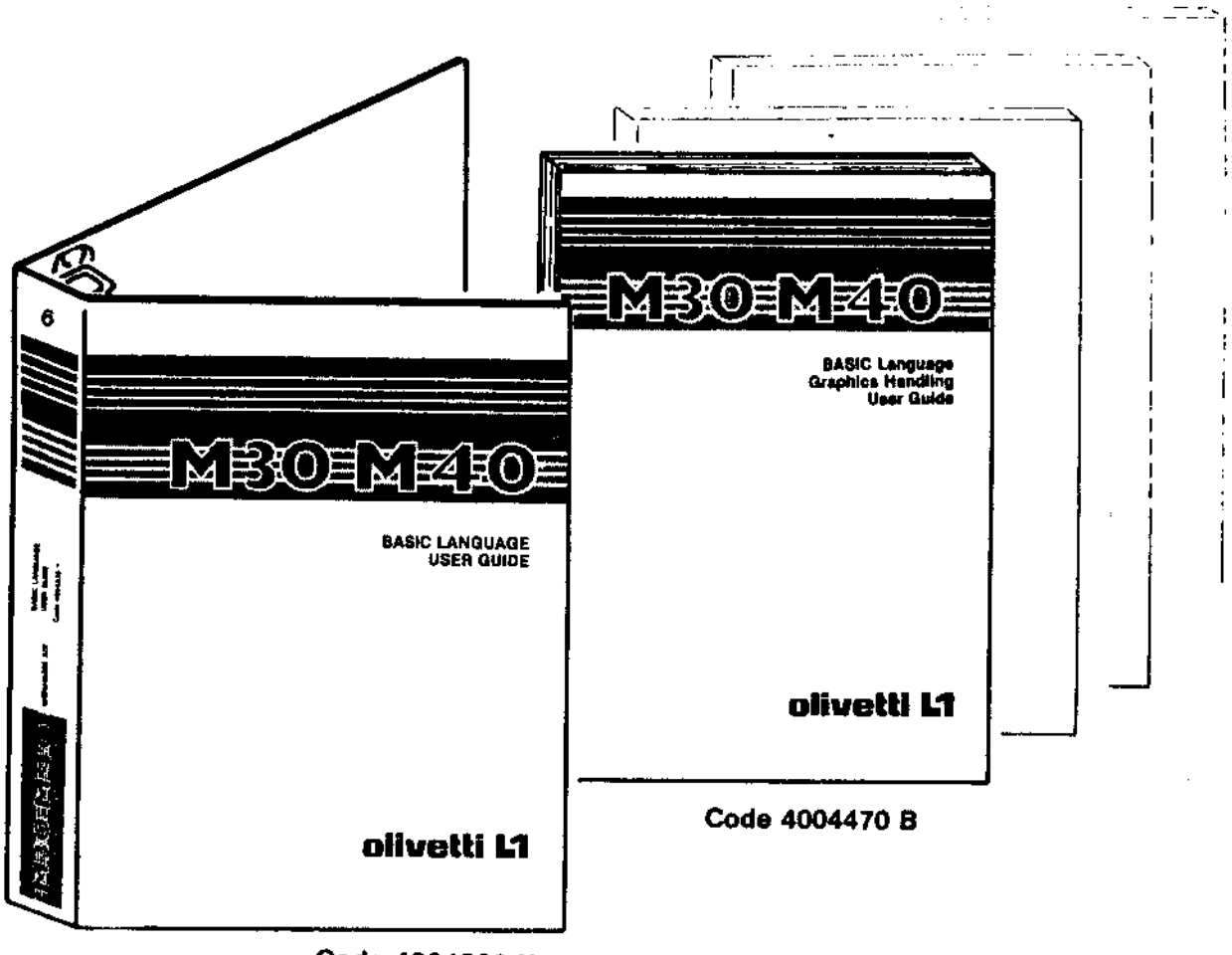
Glossary/Glossario
Code 4002140 H (vol. 1)

First Edition: January, 1983
Release 1.0

Second Edition: December, 1983
Release 2.0

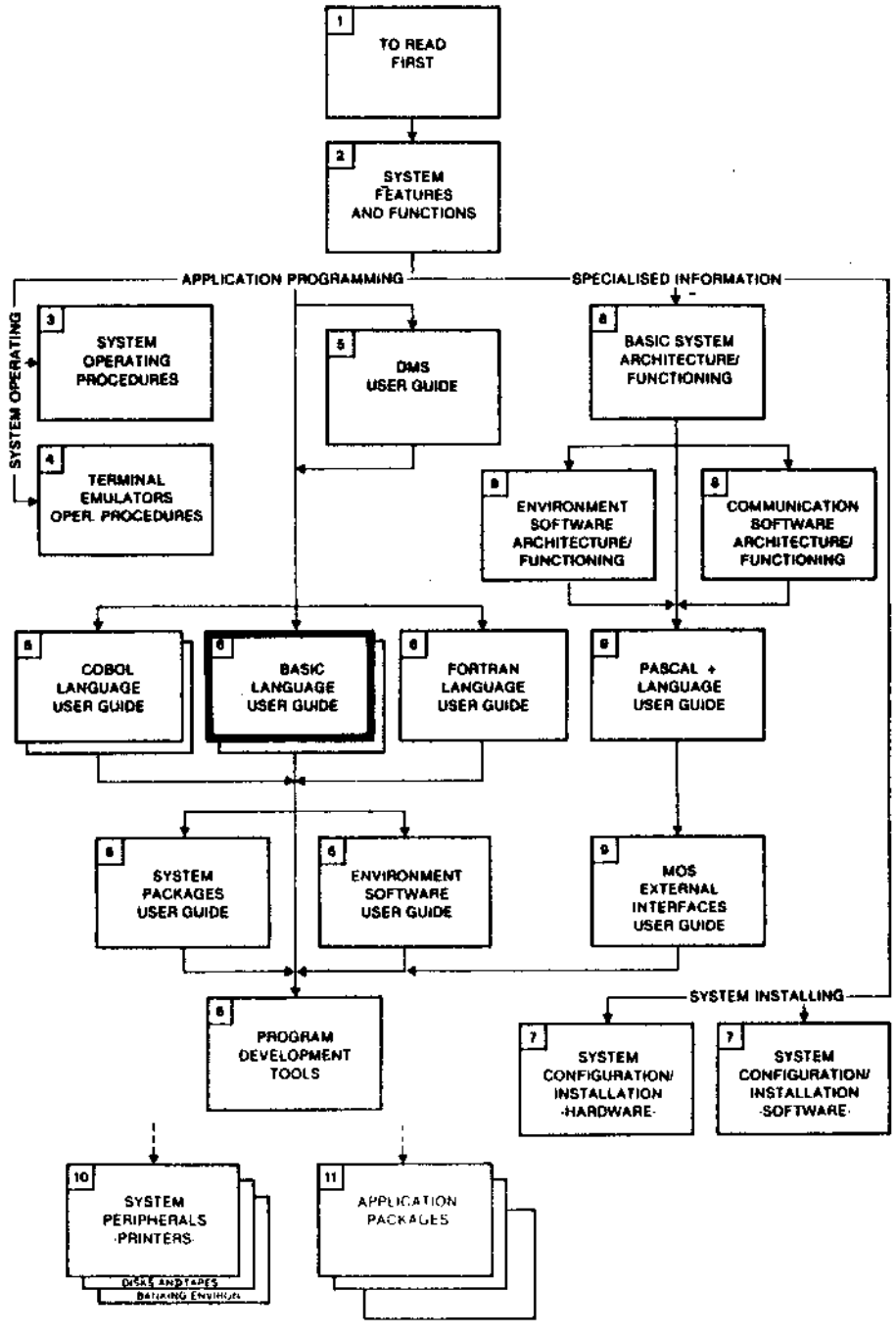
PUBLICATION ISSUED BY:

Ing. C. Olivetti & C., S.p.A.
Direzione Documentazione
77, Via Jervis - 10015 IVREA (Italy)



Code 4004470 B

Code 4004830 Y


1
TO READ FIRST
2
SYSTEM FEATURES AND FUNCTIONS
APPLICATION PROGRAMMING
SPECIALISED INFORMATION
SYSTEM OPERATING
3
SYSTEM OPERATING PROCEDURES
4
TERMINAL EMULATORS OPER. PROCEDURES
5
DMS USER GUIDE
8
BASIC SYSTEM ARCHITECTURE/ FUNCTIONING
8
ENVIRONMENT SOFTWARE ARCHITECTURE/ FUNCTIONING
8
COMMUNICATION SOFTWARE ARCHITECTURE/ FUNCTIONING
6
COBOL LANGUAGE USER GUIDE
6
BASIC LANGUAGE USER GUIDE
6
FORTRAN LANGUAGE USER GUIDE
6
PASCAL + LANGUAGE USER GUIDE
6
SYSTEM PACKAGES USER GUIDE
6
ENVIRONMENT SOFTWARE USER GUIDE
9
MOS EXTERNAL INTERFACES USER GUIDE
6
PROGRAM DEVELOPMENT TOOLS
SYSTEM INSTALLING
7
SYSTEM CONFIGURATION/ INSTALLATION -HARDWARE-
7
SYSTEM CONFIGURATION/ INSTALLATION -SOFTWARE-
10
SYSTEM PERIPHERALS -PRINTERS-
11
APPLICATION PACKAGES

INTRODUCTION

The L1 MOS System responds in a simple way to even complex problems. This simplicity can be further extended to allow the user to solve various problems graphically.

Technical data can be represented in the form of graphs, diagrams or drawings on paper, which present written text or groups of figures in a more convincing way.

The graphic features provided by the L1 MOS allow the user to store data on a screen or on an external printer connected to the system.

The L1 MOS graphic features used in scientific fields such as engineering, numeric control, mathematical analysis and statistics, offer a first-class solution in automating the ever-increasing need for graphics in a modern technological society.

1. BASIC GRAPHICS

The BASIC Graphics Package contains over thirty statements offering a set of features for the development of two dimensional graphic applications. These statements are described in Chapter 3 in alphabetical order.

The graphic features of the L1 MOS system allow:

- data to be displayed on the graphic screen; in this way, even complex information can be greatly simplified
- a coordinate system that is suitable for solving a particular problem to be defined on the graphic screen
- diagrams and drawings to be generated
- the screen to be split into a maximum of 16 rectangular areas (windows), each one handled independently.
- the screen to be regarded as a thin sheet of paper on which a "virtual pen" can be made to write. The user can move this pen to any part of the screen and in doing so, can either draw or not.

THE SCREEN AND MODES OF DISPLAY

The L1 MOS System has two different types of display available: alphanumeric and graphic. The second type of display can be used in two different modes: alphanumeric and graphic.

In both modes, the screen has a black background and a green foreground.

Alphanumeric Mode

The screen has a capacity to display a maximum of 25 lines of information with up to 80 characters per line.

The following can be displayed:

- system messages
- input from keyboard
- texts and data
- listings

The Screen in Alphanumeric Mode

The screen is divided into two parts, as the following figure illustrates, if the BASIC interpreter is run by Shell:

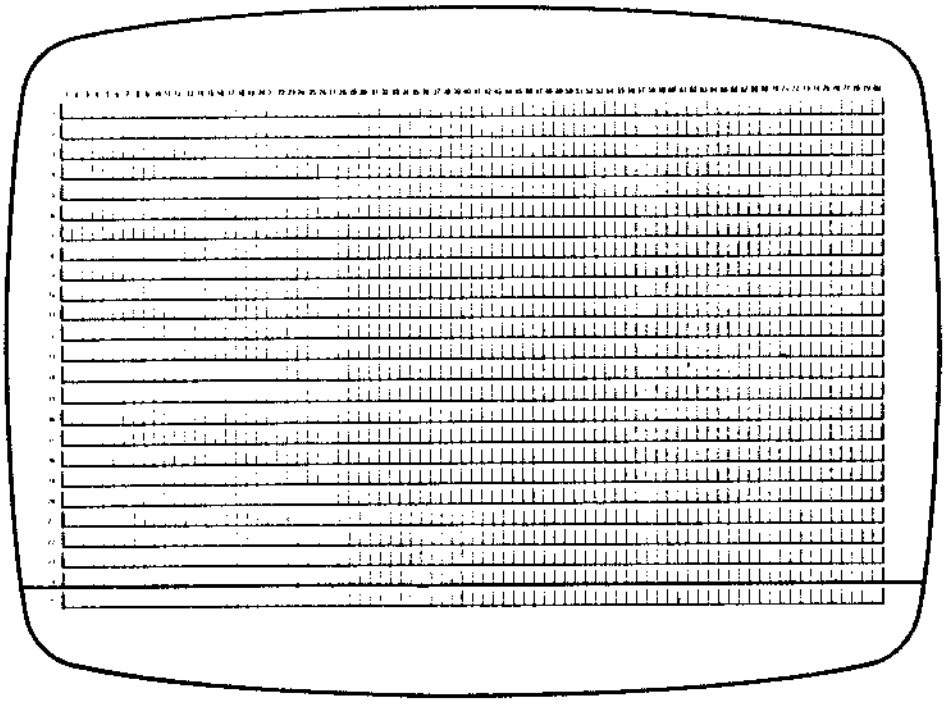


Fig. 1. 1 - The Screen in Alphanumeric Mode (with the interpreter run by Shell)

a lower part (called system line, which is reserved for the system) with a capacity of 80 characters (80 columns x 1 row); an upper part (which is reserved for the user) with a capacity of 1920 characters (80 columns x 24 rows). If the interpreter is run after the last system initialisation phase (Grandpa) all 25 lines can be used by the user.

Graphic Mode

Each picture displayed in graphic mode is not outlined by a continuous line but by a dotted line. The area of the screen where the picture is displayed can be considered a rectangular array of 640 x 384 pixels ("picture element" : a luminous dot on the screen) or 560 x 384 pixels, according to the system hardware configuration, regardless of whether the interpreter is run by Grandpa or Shell.

The Screens in Graphic Mode

If the hardware configuration is 560 x 384 pixels, in graphic mode the screen is divided into four parts, as the following figure illustrates:

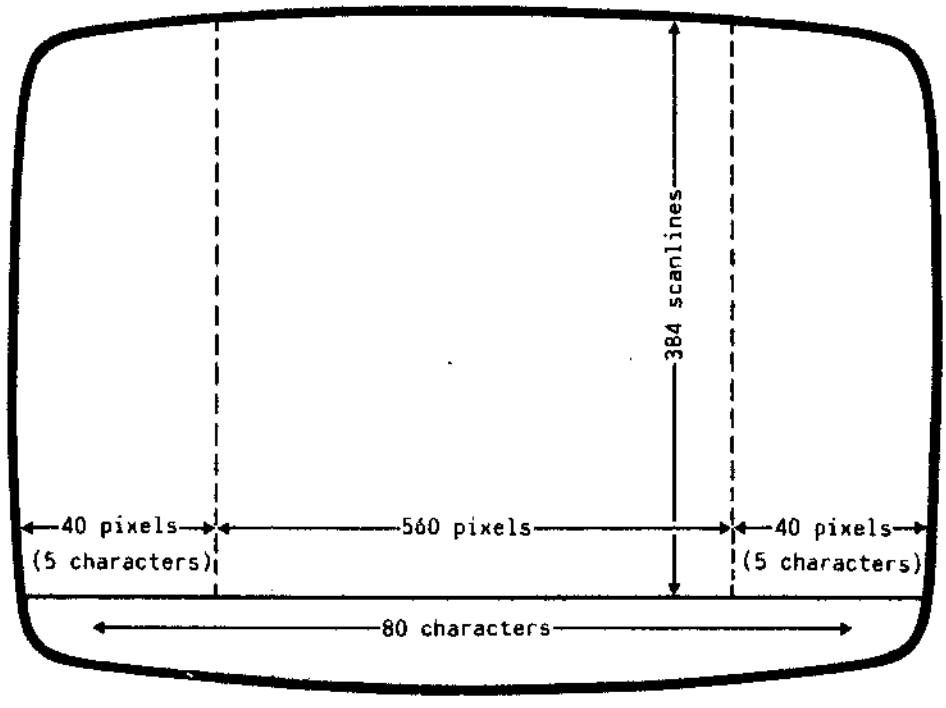


Fig. 1. 2 - The Screen in Graphic Mode (with the interpreter run by Shell)

a lower part (called system line, which is reserved for the system) with a capacity of 80 characters (80 columns x 1 row); the centre part (consisting of 560 pixels x 384 scanlines) which can be used in graphic mode; and the borders with 5 characters (corresponding to 40 pixels) which can only be used in alphanumeric mode.

If the hardware configuration is of 640 x 384 pixels the screen can be dedicated entirely to graphics, with the exception of the system line.

WINDOW HANDLING

Once in the BASIC environment, the system displays only one window (the entire screen). The user can split the screen into windows. Up to 16 windows can be handled on the screen at the same time and the dimension of each window can be established by the user through the WINDOW (Def) function. The user can operate on a window in exactly the same way as on the full screen: graphics, text or both operations can be carried out on the same window. Operations executed on different windows are entirely independent.

Opening Windows

The user can generate a new window on the screen by simply splitting, horizontally or vertically, the existing window (WINDOW (Def) function).

Whatever was previously contained in the rectangular area of the screen, which will now become the new window, is deleted. The background and foreground colours and the cursor shape are the same as those of the parent window.

Use of Windows

The user can display and/or compare texts and/or diagrams by using these windows: each window can be used to insert text and/or for graphics.

If the user tries to draw a figure which is larger than the window, only the part which fits is displayed. The rest is clipped.

The user can select the window on which to operate (WINDOW (Sel) statement). Whatever is displayed on a window can be cancelled at any time (CLS statement).

Closing Windows

It is possible to close any of the opened windows at any time. The system can also be brought back to the "initial state", whereby there is only one window: the entire screen. This can be carried out by means of the CLOSE WINDOW statement. The window space that is closed is assigned to the nearest windows.

Coordinate System

If graphic operations are carried out, the origin of the device coordinate system is placed in the lower left-hand corner of the window; the positive semi-axis of the x abscisses goes from the origin to the right. The positive semi-axis of the y ordinates goes from the origin upwards, and the coordinates of each dot are expressed in pixels (from 0 to 559 or from 0 to 639) along the x-axis and in scanlines (from 0 to 383) along the y-axis.

The graphic coordinates are relative to a specific window (for each window 560 or 640 along the x-axis and 384 along the y-axis).

It is possible for the user to operate on each window by defining his own coordinate system ("world coordinates") that is suitable to the problem to be handled (SCALE statement).The graphic axes and the displayed graphic contents can be rotated and/or translated via the ROTATE and TRANSLATE statements.

Colour Codes

The monochromatic graphic screen of the L1 MOS system has only two available colours: black and green.

The following table lists the two colours with their corresponding colour codes.

COLOUR CODE	COLOUR
0	black
1	green

Displaying the Cursors

Each window has a text cursor and a graphic cursor. The latter can be selected (CURSOR POINT statement) from the following cursors: hardware, software, cross, rubber band and rubber rectangle.

One of these can be displayed in any required position.
Its position does not vary when graphic statements are being executed. The GPOS function allows the position of the graphic cursor to be known.

The display of one of the cursors is only possible in the window in which the user is currently operating.

Once the user selects another window, the cursors that were displayed in the previous window disappear. They are, however, stored in memory and every time the user returns to that window, the cursors re-appear with the same characteristics. Note that when the text cursor is enabled, the graphic cursor is automatically disabled but remains on display. If, however, the user is operating in graphic mode, the text cursor is disabled and disappears from the screen.

HOW TO DRAW GEOMETRIC FIGURES

The L1 MOS graphics include statements allowing the user to draw geometric figures.

The most elementary graphic function, i.e. displaying a dot, is performed with the statement PSET.

It is possible to display lines (LINE statement), rectangles (LINE statement with B option) and cross-sections (POLYLINE statement) with the choice of seven types of line (STYLE statement). There is also the possibility of drawing a line away from the current position.
The L1 MOS Graphics Package has functions available for drawing arcs, circular segments, circular sections, circles and ellipses (CIRCLE and ARC statements). The graphic images can overlap one another.

The PAINT statement allows the user to fill the area closed by a figure with the foreground colour or with a character.

The user has two special statements at his disposal (GET and PUT). The GET statement allows the contents of a whole or a portion of a window to be stored in a one-dimensional integer array. The PUT statement allows the contents of a window, or part of it, that was previously stored in memory, to be transferred to the screen.

2. OPENING AND CLOSING A BASIC GRAPHIC SESSION

Modes

The user can choose to enter the BASIC Graphic environment through Shell or directly after the last system initialisation phase (Grandpa) as shown in the following figure:

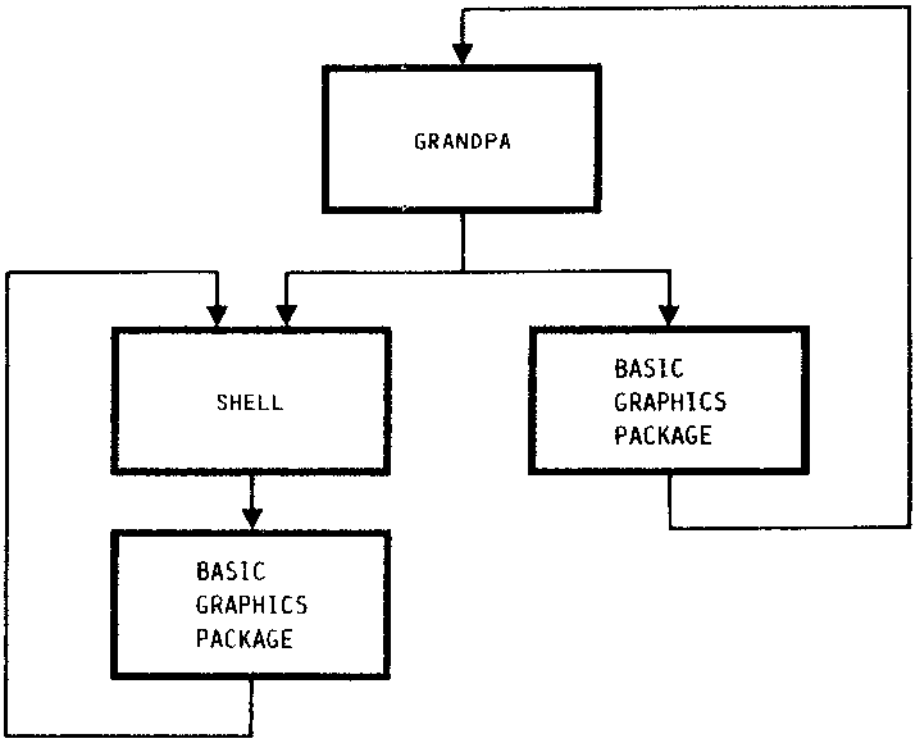


Fig. 2. 1 - Running the BASIC Graphic Interpreter

By entering through Shell, the user can vary the default conditions with which the BASIC Graphic environment is initialised (for example, the maximum number of files which can be opened at the same time, record length, etc.) and set more appropriate values.

To run the BASIC Graphic Interpreter from the Shell environment the user must simply enter the command GBASIC. The Shell environment remains stored in memory.

The command SYSTEM allows the user to exit from the BASIC Graphic environment and return to the Shell environment.

By running the BASIC Graphic Interpreter from Grandpa, the loading of the Shell environment into memory is avoided, but it also means that the user cannot alter the default conditions.

In this case, to enter the BASIC Graphic environment, only the number corresponding to the item concerned (which is present in the menu displayed at system initialisation time) has to be given.

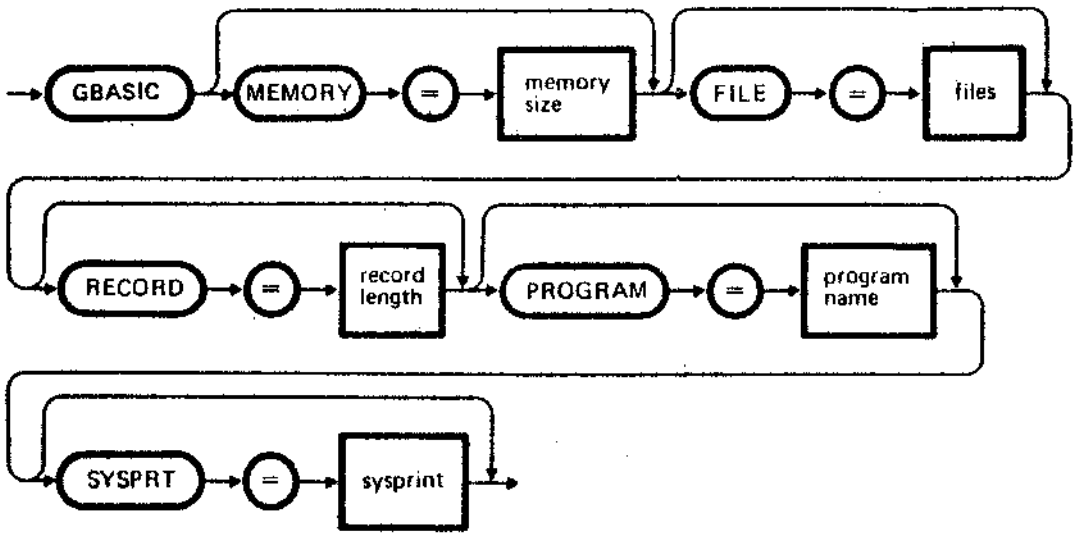
The SYSTEM command allows the user to exit from the BASIC Graphic environment and return to Grandpa.

The GBASIC and SYSTEM Commands

The two commands GBASIC and SYSTEM, are described in detail below

Allows the user to pass from Shell environment to BASIC environment, and to give the global parameters for BASIC programming activating, at the same time, the graphic operations.

IMMEDIATE Command



where:

memory size — is the initial value (expressed in number of bytes) assigned to each of the three user areas into which the memory area occupied by BASIC is divided.
The three user areas are specified by the user with the MEMORY parameter in the following order:

- area of the numeric variables (area A)
- area of the string variables (area B)

- code area (area C).

Therefore, the MEMORY parameter assumes the following structure:

MEMORY = area A/area B/area C

The maximum value for each area is 65535 bytes, the minimum value is 2048 bytes (the minimum value for the area of the string variables must not, however, be less than the area requested by the I/O buffers).

The default value for each area is 8192 bytes.

For example:

MEMORY = 40000/32000/60000

provides 40000 bytes for the numeric variables, 32000 bytes for the string variables and 60000 bytes for the code area.

MEMORY = 56000

provides 56000 bytes for the numeric variables and the default value (8192 bytes) for the string and code areas.

MEMORY = //63000

provides the default value (8192 bytes) for the numeric and string areas and 63000 bytes for the code area.

The BASIC interpreter automatically increments each area until the maximum value is reached (65535 bytes or the maximum area available in memory) if the space assigned to it is insufficient.

files — is the maximum number of files which can be opened at the same time in BASIC. The values allowed are between 1 and 15.
The default value is 15.

record length — is the maximum length of a record. The values allowed are between 1 and 4056.
The default value is 256 bytes.

program name — is the name of the program which is automatically run when the system enters the BASIC environment. When the program has been executed the system automatically returns to the Shell environment.

sysprint — identifies the device or file on which the output of the LLIST, LPRINT and SPRINT statements is to be directed. If sysprint is omitted the output is sent to the work-station printer (if connected), otherwise it is sent to the standard output device (video).
The sysprint parameter can assume the following values:

a) 1

b) 2

c) 3

d) 4

e) filename

In the first four cases the output of the LLIST, LPRINT and SPRINT statements is sent to the system printer 1, 2, 3, 4 respectively.
If the specified printer is not available the BASIC interpreter is not run. If the printer is available the output is sent to it, and this printer can be shared by other users.
In the fifth case the output can be sent to a file and the filename parameter can be:

- a global name

  For example:

  PIPPO/VOL/CA10,/USER/ROOT/PRINT

- a local name previously connected to a global name (with the Shell CONN command)

  For example:

  ```
  CONN PIPPO /USR/ROOT/VOL/PRINT
  GBASIC SYSPRT = PIPPO
  ```

  The LLIST, LPRINT and SPRINT statements operate on the file /USR/ROOT/VOL/PRINT

  ```
  CONN PIPPO /DEV/SYSPRT1
  GBASIC SYSPRT = PIPPO
  ```

  The LLIST, LPRINT and SPRINT statements operate on the system printer 1.

If the name SPOOL1, SPOOL2, SPOOL3, SPOOL4 is assigned to the file when it is opened the printers can be handled by the spooler. The following spooler

classes are used, respectively: 1, 2, 3, 4.

If the spooler has not been installed, or the specified class has not been activated, the file-opening operation fails. After having executed the write operations, the CLOSE statement appends the file.

Before entering the BASIC environment a connection between the spooler and the specified class must be made.

For example:

CONN PIPPO SPOOL1

the BASIC environment is run

OPEN "O", #1, "PIPPO"

is equivalent to:

OPEN "O", #1, "SPOOL1"

The only difference between the two examples is that, in the first case, the file is appended after the BASIC session has been closed, whilst, in the second case, the job is simply appended after a CLOSE statement has been executed.

Note

The call parameters in the BASIC environment are "keyword" types and can, therefore, be entered in any order.

Example

```
GBASIC MEMORY = 16384 RECORD = 512 FILE = 4
PROGRAM = PIPPO SYSPRT = PRINT
```

The system enters the BASIC environment. 16384 bytes are assigned as the initial value for the numeric variables and the default value (8192 bytes) is assigned to the string variables and code area. The maximum length of a record is 512 bytes. The maximum number of files which can be opened at the same time in BASIC is 4.
PIPPO is the name of the program which is automatically executed on entering the BASIC environment. The output is sent to the PRINT file.

Closes the BASIC session and returns the system to Shell environment or to the first menu of the environment selector (Grandpa).

PROGRAM/IMMEDIATE Command

Characteristics

The SYSTEM command can be executed in either immediate mode or program mode.
If executed in immediate mode and if a user program is present in memory, before returning to Shell or the first menu, the following request is displayed:

N-SAVE PROGRAM? N/Y

If the reply is Y/y, BASIC returns to Command Mode; if the reply is N/n the system returns to Shell or the first menu (Default: N).
Before returning to the Shell environment or the first menu, all the files are closed.

3. THE BASIC GRAPHIC STATEMENTS AND FUNCTIONS

Analytic Index

The statements and functions of the L1 MOS Graphics Package can be grouped in the following categories:

transformation and control

- varying the physical default limits of the main window: BOUNDS
- closing the specified window: CLOSE WINDOW
- deleting the contents of the window: CLS
- colouring an area: PAINT
- rotating the graphic axes of a specified angle: ROTATE
- defining the space of the world coordinates: SCALE
- translating the coordinates system's origin: TRANSLATE
- creating a new window: WINDOW (Def)
- selecting and activating the specified window: WINDOW (Sel)

graphic output

- drawing an arc, the outline of a circular segment, the outline of a circular sector: ARC
- drawing a circle or ellipse: CIRCLE
- moving the current graphic position within the specified window and drawing graphic images: DRAW
- moving the graphic cursor: GIN
- displaying a text character string: LABEL

- drawing a line or rectangle: LINE
- drawing a cross-section: POLYLINE
- displaying specified pixels: POLYPIXEL
- displaying a pixel in the background colour: PRESET
- displaying a pixel in the specified absolute position: PSET
- displaying a previously stored image: PUT
- executing the hard copy: SPRINT

attributes

- selecting the foreground and background colours for the current window: COLOR
- selecting the graphic cursor to be displayed and defining the shape of the cursor: CURSOR POINT
- selecting the type of line: STYLE

inquiry

- storing the contents of an entire window or a part of it: GET
- returning the position of the graphic cursor of the current window: GPOS
- returning the colour code of the pixel which is nearest to a specified point: POINT
- returning the device abscissa corresponding to the specified world abscissa: SCALEX
- returning the device ordinate corresponding to the specified world ordinate: SCALEY

buffer handling

- emptying the buffer: FLUSH
- deactivating the use of the buffer: BRESET

- activating the use of the buffer: BSET

Default Conditions

A default operating mode exists for the L1 MOS BASIC Graphic Package which automatically assumes all the values regarding the format. The user can vary these default conditions for values which are more appropriate to specific problems.

The default conditions for the BASIC Graphic Package are the following.

- the world coordinates interval coincides with the device coordinates interval (except that these last are whole numbers) and go from 0.0 to 559.0 or from 0.0 to 639.0 on the x axis, and from 0.0 to 383.0 on the y axis (see Chapter 1 "Graphic Mode").

- window number 1 is the entire screen, with device coordinates from 0 to 559 or from 0 to 639 on the x axis, and from 0 to 383 on the y axis (see Chapter 1, "The Screens in Graphic Mode").

- the colour depends on the system configuration

  a) the monochromatic system assigns the two values 0 (black) and 1 (green) respectively to the background and foreground colour

  b) the eight-colour system assigns the following values:

     0 = black, 1 = green, 2 = blue, 3 = cyan,
     4 = red, 5 = yellow, 6 = magenta, 7 = white;

     the background colour is black and the foreground is green (when available)

- the logical operator is PSET, which allows the output to be displayed in the foreground colour

- the line type is continuous

- the graphic cursor is not displayed

- the text cursor is displayed

- the shape of the "cross" graphic cursor is a greek cross with five pixels to each arm

- the shape of the "software" graphic cursor is an arrow leaning to the left, within an 8 x 10 pixel box

- the initial graphic position is (0.0, 0.0)

- the text cursor position is 1,1 (first column, first row)

Description of Statements and Functions

This chapter contains a detailed description of all the BASIC Graphic statements and functions. They are listed in alphabetical order for easy reference. Each description is ordered as follows:

- The function of the statement or the function with indications as to whether it can be used in program and/or immediate mode. If there is just a function, these indications are omitted as a function can be used in an immediate line and a program line

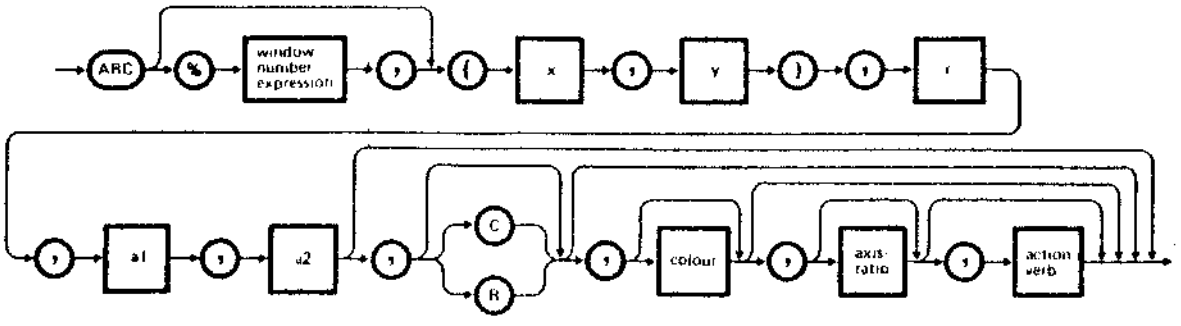
- Syntax

- Operating modes of the statement or function

See Appendices A and B of the manual "BASIC Language - User Guide" for the error messages.

Draws an arc or the outline of a segment or sector of a circle or an ellipse.

PROGRAM/IMMEDIATE Statement



where:

window number expression is a real numeric expression, whose value, rounded to the nearest integer, selects the window on which the statement must operate. See the window number variable parameter of the WINDOW (Def) function for the values which this expression can assume. The default value is the current window.

x,y are real numeric expressions, whose values provide the coordinates of the centre of the circle or ellipse of which an arc is to be drawn.

r is a real numeric expression, whose value must be positive. This value is the radius of the circle or the horizontal semi-axis of the ellipse.

a1 — is a real numeric expression, whose value must be within the interval -2π and $+2\pi$. This value defines the initial angle of the arc, in radians, measured anticlockwise from the x axis.

a2 — is a real numeric expression, whose value must be within the interval -2π and $+2\pi$. This value defines the final angle of the arc, in radians, measured anticlockwise from the x axis.

C — is a keyword which, when used, allows the chord of the arc to be drawn. The outline of the segment of the circle or the ellipse is drawn in this way.

R — is a keyword which, when used, allows the two radii to be drawn which join the arc with the centre of the circle or ellipse. The outline of the sector of the circle or ellipse is drawn in this way.

colour — is a colour code, which selects the colour in which to draw the arcs and the outlines of the segments and sectors. It is a whole number within the interval (-1,1). For the meaning of these values, see the same parameter of the COLOR statement. The default value is the current foreground colour of the selected window. The action verb parameter has no effect when colour's value is -1.

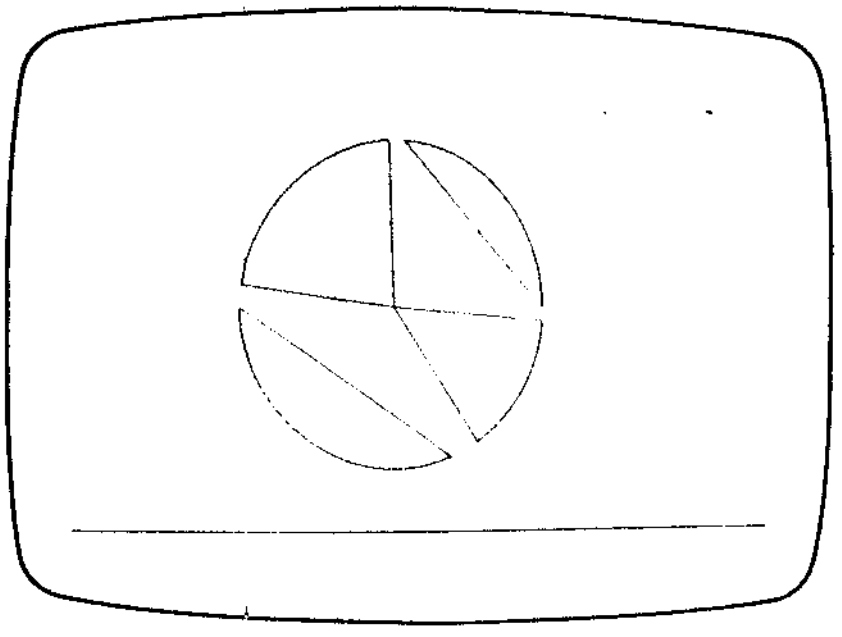
axis-ratio — is a real numeric expression, whose value must be positive. It defines the ratio between the vertical and horizontal axes of the ellipse for which the arc is to be drawn. The default value is 1 and an arc of the circumference is drawn.

action verb — is a parameter which can have one of the following values: AND, XOR, OR, NOT, PSET, PRESET. Each of these values defines the operation to be carried out on each individual pixel of the curve. With PSET the arcs and the outlines of the segments and sectors are drawn in the colour specified by the colour parameter. With AND, OR and XOR the colour used is the result of the logical operation between the binary value of the colour code specified in colour and the binary value of the colour code of each pixel on the figure to be drawn on the screen. With NOT the curves are drawn in the complementary colour to that specified by the colour parameter. With PRESET the curves are drawn in the current background colour. The default value is PSET. If the colour parameter has the value -1 the action verb parameter has no effect.

Characteristics	The current graphic position is updated to the (x,y) point, the centre of the circle or ellipse of which the arc is to be drawn.
Note	The r and axis-ratio parameters must be selected so that the semi-axes are longer or the same length as a pixel.

Example

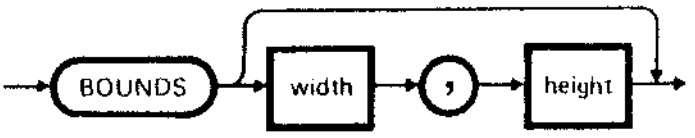
```
10 CLEAR
20 ARC (250,192),140,0,1.5,C,1
30 ARC (250,192),140,1.6,3,R,1
40 ARC (250,192),140,3.14,5.1,C
50 ARC (250,192),140,5.3,6.2,R
60 END
```



BOUNDS

Changes the default values of the sizes and units of measurement of the main window of the graphic screen. The same unit of measurement is adopted in both directions.

PROGRAM/IMMEDIATE Statement



---

where:

width — is a real numeric expression, whose rounded value must be a positive integer. This value represents the number of units of measurement which the user wants to have on the x axis ("horizontal dimension")

height — is a real numeric expression, whose rounded value must be a positive integer.
This value represents the number of units of measurement which the user wants to have on the y axis ("vertical dimension")

Characteristics

If the width and height parameters are omitted, the entire graphic zone is enabled for the graphic output, with units of measurement equal to 1; that is, the initial conditions are restored.

The sizes of the graphic area and of the units of measurement, which can be used after the execution of a BOUNDS statement, are calculated as described below.

The indications are:

hx,hy	the physical dimensions (in pixels) of the graphic screen.
Nhx,Nhy	the physical dimensions (in pixels) of the graphic screen, usable after the execution of BOUNDS
x,y	the values of the height and width parameters
Ux,Uy	the values of the units of measurement

If: hx/x is less than hy/y then:

Nhx = hx
Ux = hx/x
Nhy = (Ux*y) rounded up to 16
Uy = Nhy/y (equal to Ux if Ux*y is a multiple of 16)

If : hx/x is greater than hy/y then:

Nhy = hy
Uy = hy/y
Nhx = (Uy*x) rounded up to 8
Ux = Nhx/x (equal to Uy if Uy*x is a multiple of 8)

If : hx/x is equal to hy/y the entire screen will be used and Ux = Uy = hx/x = hy/y

The rounding up to 8 and 16 are consequences of the fact that the number of pixels forming an alphanumeric character (plus the line spacing) make up an 8 x 16 pixel array.

The area resulting from the execution of BOUNDS is considered the main graphic window and each successive window opened will refer to it.

This statement only effects the part of the screen which can be handled in graphic mode.

BRESET

Deactivates the use of the buffer.

PROGRAM/IMMEDIATE Statement

Characteristics

After the execution of this statement, each graphic statement is executed and the result displayed immediately.
Once the BRESET statement is entered, it remains active until the BSET statement has been executed.

BSET

Activates the use of the buffer.

PROGRAM/IMMEDIATE Statement

Characteristics

After the execution of this statement (and by default) the graphic commands are buffered. They are executed only:

a) when the buffer is full;

b) when a FLUSH statement is executed;

c) when a GET/SPRINT statement is executed;

d) when a POLYLINE/POLYPIXEL statement is executed;

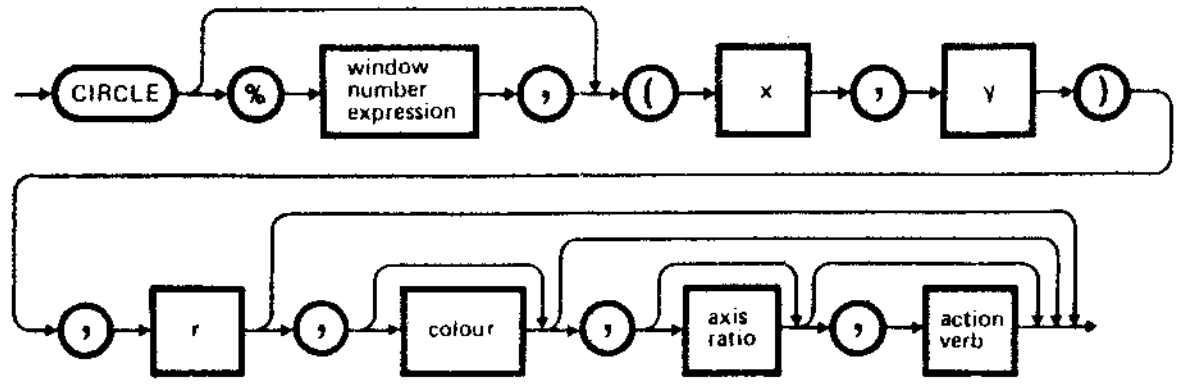
e) at the end of program execution.

Once the BSET statement is entered, it remains active until the BRESET statement has been executed.

CIRCLE

Draws a circle or an ellipse.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies and selects the window on which the statement must operate. For the values which it can assume see the window number variable parameter of the WINDOW (Def) function. If omitted, the current window is selected.
x,y	are real numeric expressions whose values supply the coordinates which define the centre of the circle or the ellipse.
r	is a positive real numeric expression whose value supplies the radius of the circle or the horizontal semi-axis of the ellipse.

colour — is a colour code specifying the colour with which the circle or the ellipse will be drawn. For the values which it can assume see the corresponding parameter of the COLOR statement. If colour has a value of -1 the action verb parameter has no effect. The default value is the current foreground colour of the selected window.

axis ratio — is a positive real number which defines the ratio between the vertical and horizontal axes of an ellipse. The default value is 1 and draws a circle.

action verb — is a parameter which can assume one of the following values: AND, XOR, OR, NOT, PSET, PRESET.
Each of these define the operation to be executed on each pixel of the curve (circle or ellipse).
With PSET the circle or ellipse is drawn with the colour specified by the colour parameter. AND, OR and XOR indicate that the colour of the circle or ellipse is the result of a logical operation between the binary value of the specified colour code and the binary value of the colour code of each pixel along the curve on the screen.

With PRESET the circle or ellipse are drawn with the current background colour. With NOT, the circle or ellipse are drawn in the complementary colour to that specified by the colour parameter. The default value is PSET.

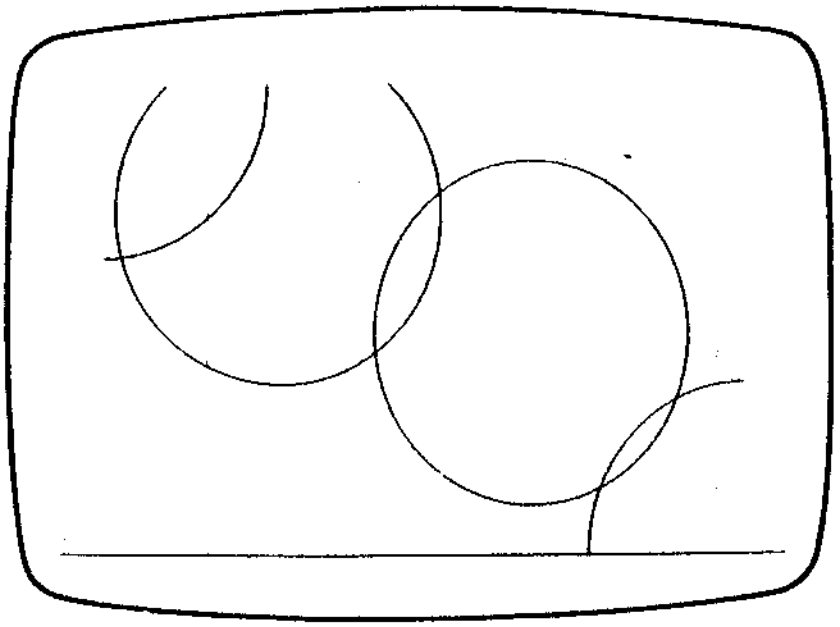
If colour has the value -1 the action verb parameter has no effect.

Characteristics — The current graphic position is updated to the (x,y) point, centre of the circle or ellipse.

Note — The r and axis ratio parameters must be selected so that the semi-axes are longer or the same length as a pixel.

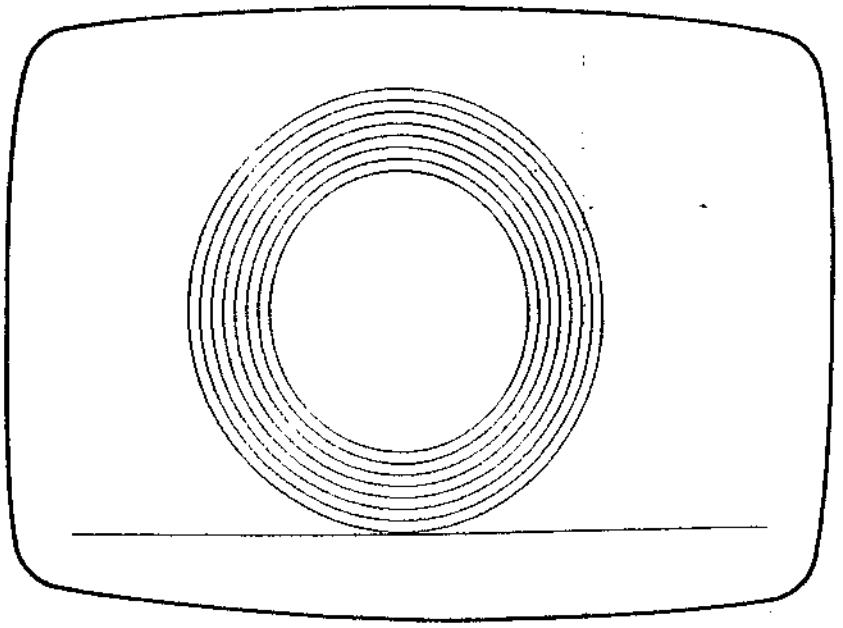
Example 1

```
10 CIRCLE (0,384),140
20 CIRCLE (150,280),140
30 CIRCLE (370,180),140
40 CIRCLE (560,0),140
```



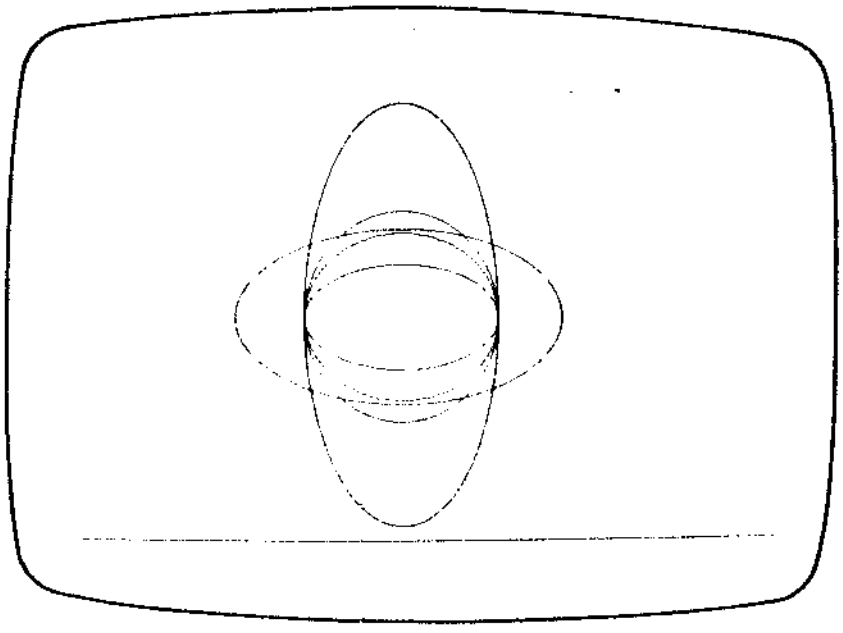
Example 2

```
10 FOR I = 0 TO 7
20 CIRCLE (256,191),190 - I*10
30 NEXT
```



Example 3

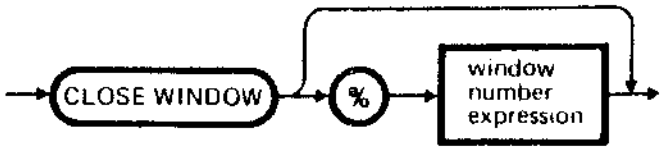
```
10 CLEAR
20 CIRCLE (250,192),90,1,2
30 CIRCLE (250,192),90,1,.5
40 CIRCLE (250,192),90,1,.8
50 CIRCLE (250,192),90
60 CIRCLE (250,192),150,,.5
70 END
```



CLOSE WINDOW

Closes a specified window or all opened windows and clears its/their contents.

PROGRAM/IMMEDIATE Statement



where:

window number expression — is a real numeric expression whose value, rounded to the nearest integer, identifies the window to be closed. If omitted, the system returns to the initial state (only one window: the entire screen). For the values it can assume see the window number variable parameter of the WINDOW (Def) function.

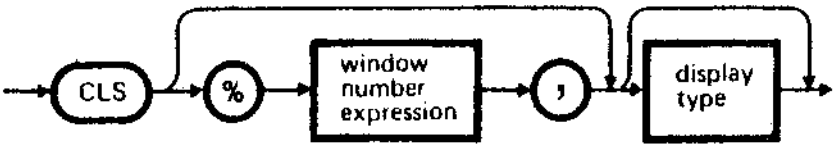
Characteristics — The system assigns the area of the closed window to the rectangle which was originally split to open it.

The CLOSE WINDOW statement has no effect on the main window. Window number 1 (or the main window) can never be closed.

The current graphic position and the graphic cursor position are moved to the lower left-hand corner of all windows to which the space of the closed window has been assigned.

The graphic and/or alphanumeric contents of the current window (or of a specified window) is cleared. The text cursor is positioned in the upper left-hand corner of the window itself.
The graphic cursor and the current graphic position are moved to the origin of the coordinates.

PROGRAM/IMMEDIATE Statement



where:

window number expression — is a real numeric expression whose value rounded to the nearest integer identifies the window on which the statement must operate. For the values which it can assume see the window number variable parameter of the WINDOW (Def)·function. If omitted, the operation is executed on the current window.

display type — is a parameter that can assume two values: A or G.

A : clears only the alphanumeric contents and positions the text cursor

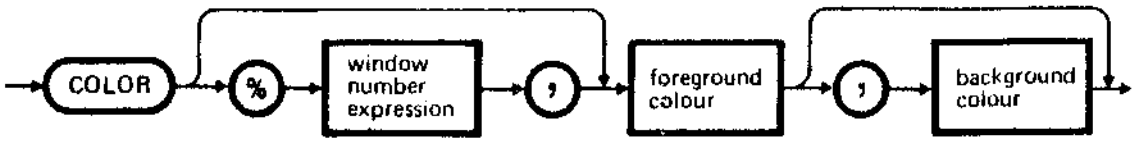
G : clears only the graphic contents and does not move the text cursor

If omitted, both types of contents (graphic and alphanumeric) are cleared.

COLOR

Selects the background and foreground colours for the current (or specified) window.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, identifies the window on which the statement must operate. For the values it can assume see the window number variable parameter of the WINDOW (Def) function. If omitted, the system operates on the current window.
foreground colour	is a colour code which specifies and selects the foreground colour. Foreground colour can assume a whole value between (-1,1). The value 1 corresponds to green, the value 0 corresponds to black, and -1 is complementary. The default value is 1.
background colour	is a colour code which specifies and selects the background colour. Background colour can assume a whole value between (-1,1). The value 1 corresponds to green, the value 0 corresponds to black, and -1 is complementary. The default value is the current background colour.

Note — The effects of this statement will be visible only when the graphic statements are executed.

Example 1 — COLOR 0,1

The current window has a green background and a black foreground.

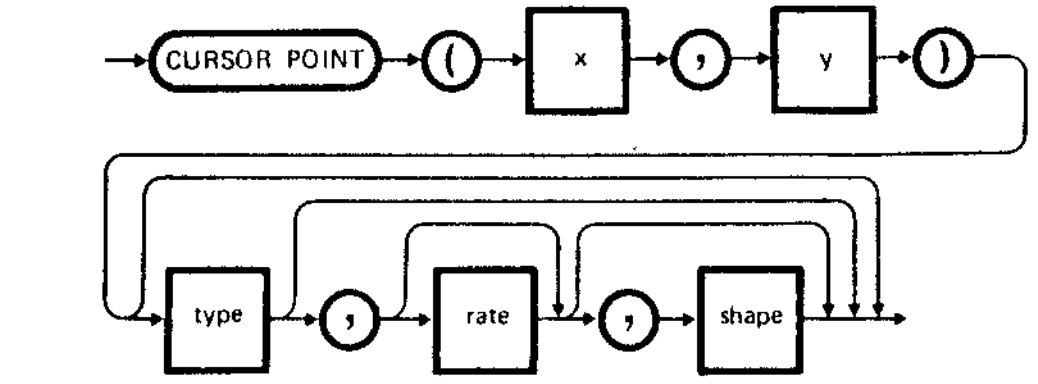
Example 2 — COLOR %A,0,1

As above, but the COLOR statement operates on the window identified by the variable A.

CURSOR POINT

Selects one of the graphic cursors and specifies its position and its characteristics.

PROGRAM/IMMEDIATE Statement



where:

x,y — are real numeric expressions whose values specify where the cursor must be positioned.

type — is a parameter which specifies whether or not the cursor is to be displayed and if so which one. The permitted values are:

0 : graphic cursor OFF

1 : software cursor ON

2 : cross cursor ON

3 : hardware cursor ON

4 : rubber band cursor ON

5 : rubber rectangle cursor ON

rate — this parameter has no effect; it is accepted only for compatibility.

shape — allows the shape of the graphic software and graphic cross cursors to be varied. It is the first element of a one-dimensional five-element integer array: it defines the bitmap (an 8*10 pixel array) of the software cursor. Each element of the array represents the status of 16 bits. The default shape for the software cursor is an arrow pointing upwards and to the left.
The point of the arrow coincides with the specified point (x,y).
The size of the cross size cursor, however, is established using a single-element integer array: the value of this element defines the length of the arm of the cross (expressed in numbers of pixels). The default value is 5 pixels.

Characteristics — The software cursor is the result of the definition of the status of the bits in a 8 x 10 pixel array.

The position of the graphic software cursor (x,y) coincides with the upper left-hand corner of the bitmap.

The cross cursor is a greek cross, centred on the point (x,y).

The hardware cursor is a cross which is centred on the point (x,y) which covers the whole graphic area.

The rubber band cursor is a line joining the current point with the specified point (x,y).

The rubber rectangle cursor is a box whose diagonal joins the current point with the specified point (x,y).

The graphic cursor characteristics are cleared by entering a new CURSOR POINT statement or a CLEAR statement.

The graphic cursor position is updated to the (x,y) point, whilst the current graphic position remains unchanged.

Example 1

```
CURSOR POINT (80,30)
```

The graphic cursor is positioned at the point with coordinates (80,30) and not displayed.

Example 2

```
CURSOR POINT (50,50) 1
```

The graphic cursor is positioned at the point with coordinates (50,50); and the software cursor is displayed.

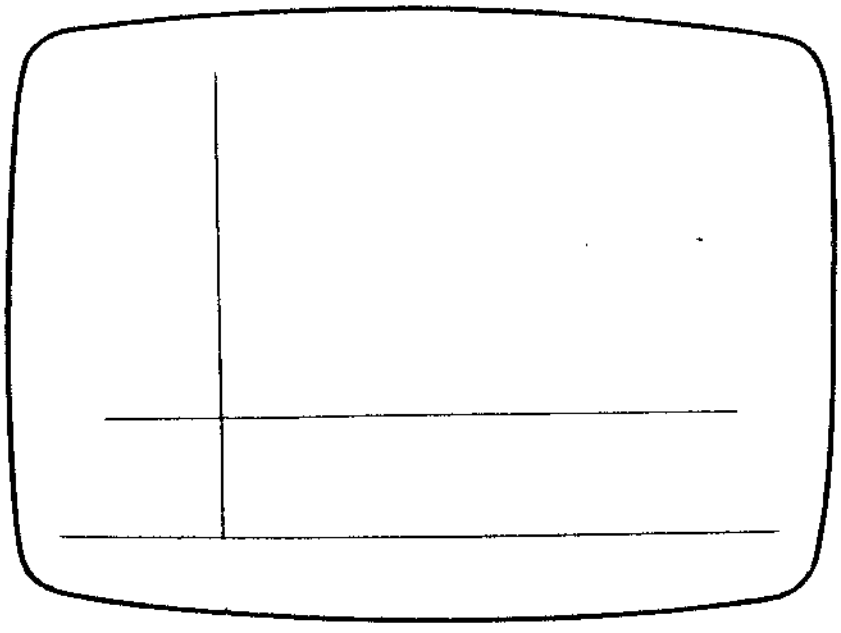
Example 3

```
CURSOR POINT (50,50) 2
```

The graphic cursor is positioned at the point with coordinates (50,50); and the cross cursor is displayed.

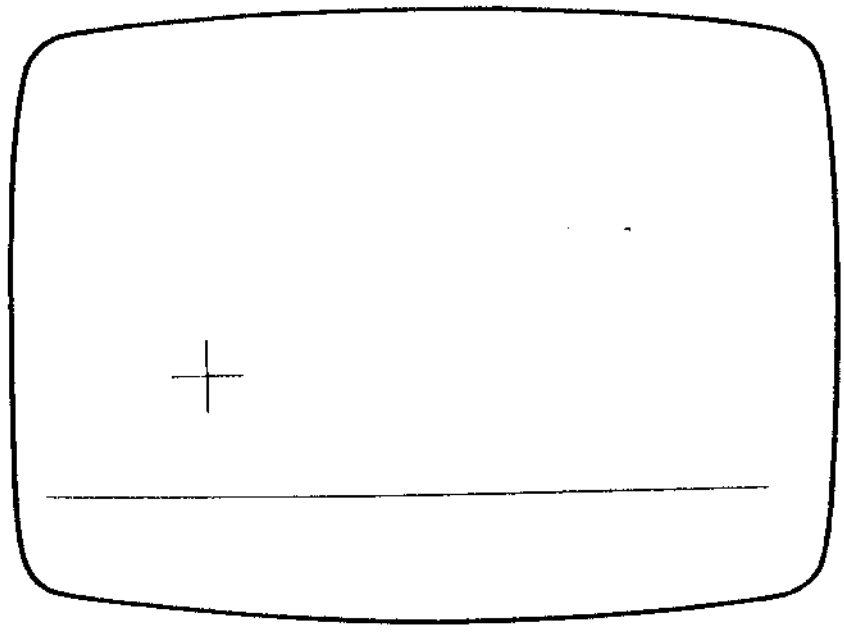
Example 4

```
10 CURSOR POINT (100,100)3
15 A=GIN(9)
```



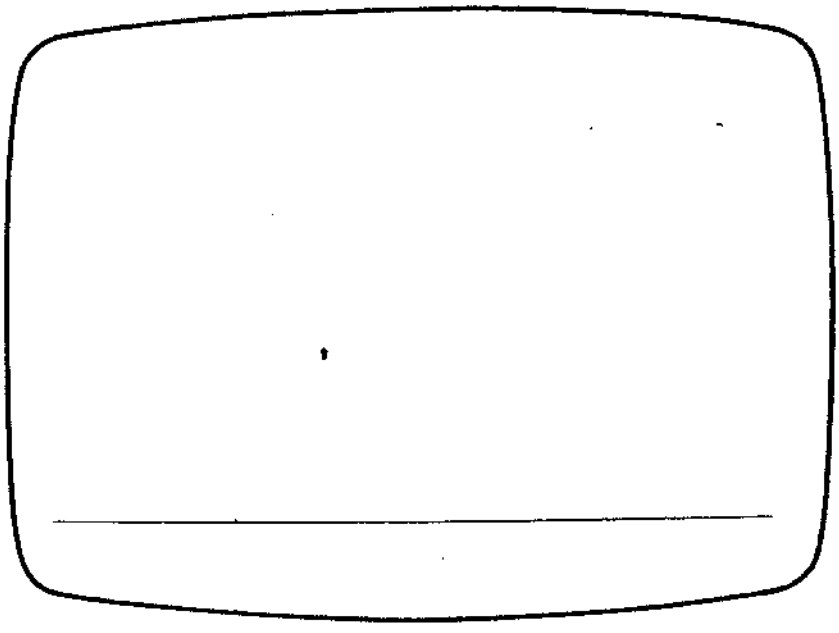
Example 5

```
5 A%(0) = 30
10 CURSOR POINT (100,100)2,,A%(0)
20 A = GIN(15)
```



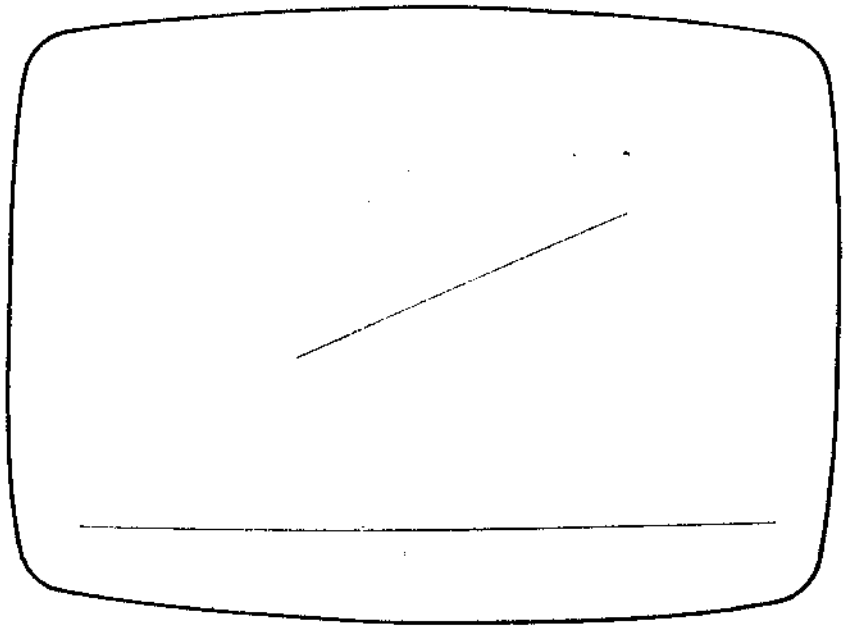
Example 6

```
5 A%(0) = 4152
6 A%(1) = 31998
7 A%(2) = 14392
8 A%(3) = 14392
9 A%(4) = 14392
15 CURSOR POINT (10,10)1,,A%(0)
20 A = GIN(15)
```



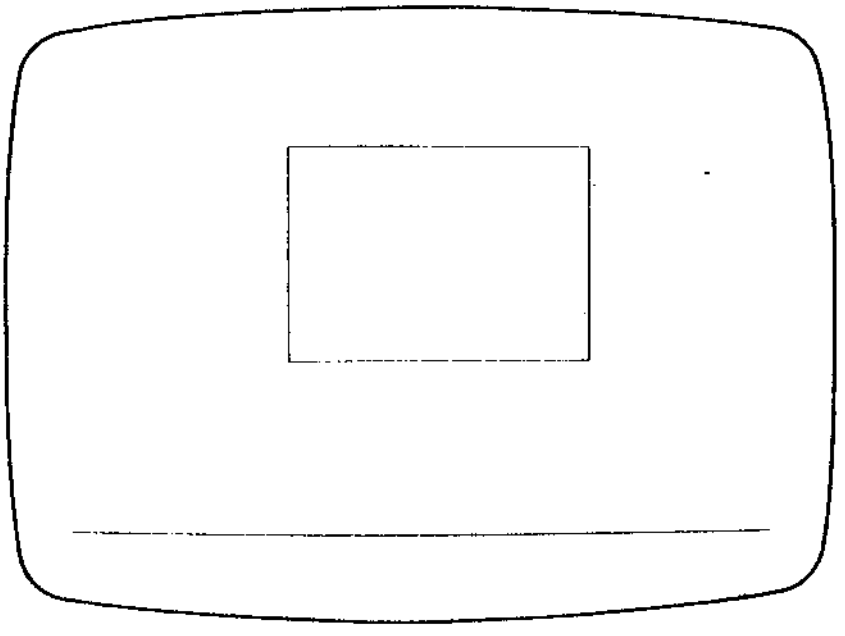
Example 7

```
10 CLEAR
20 PRESET (150,150)
30 CURSOR POINT (300,250)4
40 A = GIN(10)
50 END
```



Example 8

```
10 CLEAR
20 PRESET (150,150)
30 CURSOR POINT (300,250)5
40 A = GIN(10)
50 END
```



DRAW

Draws graphic images.

PROGRAM/IMMEDIATE Statement

where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
command string	can be either a string variable or a string constant. The string, in both cases, consists of one or more commands (listed in the table Commands) which control the movement of the current graphic position. With the exception of the C command, all commands may be prefixed with the B option, which inhibits drawing, and followed by one of the following action verb parameters: AND, XOR, OR, NOT, PSET, PRESET. Each of these define an operation that can be executed on any point of the line. They are specified by the first letter of their name, except PRESET which is specified by R. P (i.e. PSET) draws the figure in the specified colour. A (i.e. AND), O (i.e. OR), and X (i.e. XOR) indicate that the colour of the figure is the result of a

logical operation between the binary value of the specified colour code and the binary value of the colour code of each pixel along the figure on the screen.
N (i.e. NOT) indicates that the colour of the figure is the complement of the colour specified in the colour parameter.
R (i.e. PRESET) draws the figure in the current background colour.
The default value is P.

If the colour parameter of the C command has a value of -1, the action verb parameters have no effect.

COMMAND	MEANING
M dx, dy	Moves the graphic position from its current position (a,b say) to the position indicated by: (a+dx, b+dy)
J x,y	Moves the graphic position to the position indicated by: (x,y)
U dy	Moves the graphic position up by dy positions along the y axis
D dy	Moves the graphic position down by dy positions along the y axis
L dx	Moves the graphic position left by dx positions along the x axis
R dx	Moves the graphic position right by dx position along the x axis
C colour	Sets the colour to be used to draw the graphic images. A colour code must be specified after C. For the values which colour can have, see the same parameter of the COLOR statement. If colour has a -1 value the action verb parameters have no effect. If no C is specified, either the last colour used in a previous DRAW statement or the foreground colour of the current window is assumed.
X Name$	Draws graphic images using the sequence of commands specified by the Name$ string

Tab. 3. 1 - Commands

Characteristics

Command parameters (dx, dy, x, y and colour) can be expressed as variables. In this case, the variable names must be written between equal signs.
The sequence of commands in a DRAW statement may be entered either in lower-case or in upper-case letters. They may be separated by blanks or written contiguously.
The current graphic position is updated to the last selected point.

Example 1

```
90 PSET (10,20)
100 X=23
...
130 DRAW "M=X=,25"
...
```

Statement 90 colours the point (10,20) with the current foreground colour.

Statement 100 sets X = 23

Statement 130 draws a line from the current graphic position (10,20) to the position (33,45), that is, (10+23, 20+25)

Example 2

```
250 A$ = "BM 10,2
    D 20 MR 15,-3"
```

Statement 250 assigns the following command string to the A$ variable:

- the M command with the B option to move the graphic position, without drawing,from its current position (a, b say) to the position (a+10, b+2)
- the D command to move the graphic position down 20 positions, i.e. to the point (a+10, b-18)
- the M command to move the graphic position from its current position (a+10, b-18) to the point (a+25, b-21)

The R option (PRESET) indicates that the line must be drawn in the current background colour.

Example 3

```
260 DRAW A$
```

Statement 260 executes the sequence of commands specified by the A$ variable.

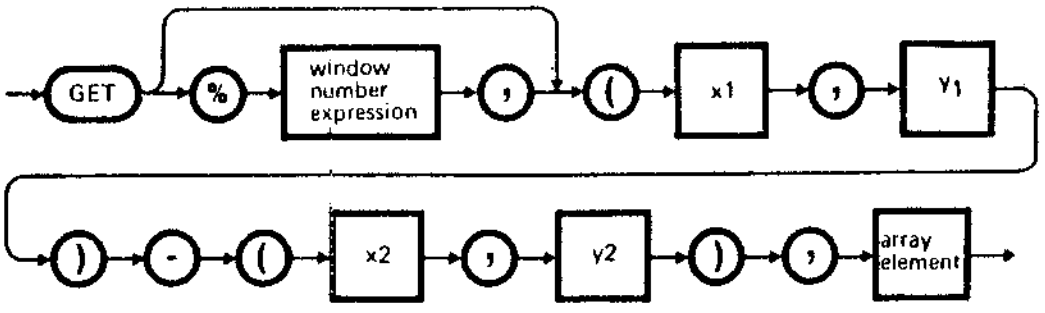
FLUSH

Empties the buffer. The bufferized graphic commands are executed.

IMMEDIATE/PROGRAM Statement

Stores the whole or a portion of a window in a one-dimensional integer array (16 bits for each element).

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x1,y1, x2,y2	are real numeric expressions whose values specify the coordinates which define the extreme points of the diagonal of the rectangle to be stored. The specified extremes must be within the current window.
array element	is the first element of a one-dimensional integer array. The array elements are used as follows: - the first element will contain the width of the rectangle

- the second will contain the height of the rectangle

- the value of the third element determines whether the image to be stored is monochrome or not.

The remainder of the array will contain information on the status of each bit of each scanline which forms the rectangle. Each array element contains the status of 16 bits.
This one-dimensional array must be previously dimensioned using the DIM statement.

The following formula shows how to calculate the number of elements of the array:

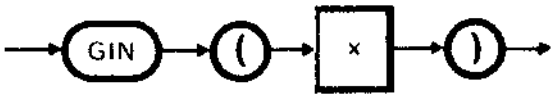
$$\left\lceil \frac{\text{width}}{16} \right\rceil \times \text{height} + 3$$

where $\lceil\ \rceil$ means take integer (always round up)

Characteristics

The current graphic position is not updated.

This function moves the graphic cursor according to the arrow keys (↑ , ↓ , → , ←) and returns the ASCII code of the first key to be entered after the cursor arrow keys or the hardcopy key.



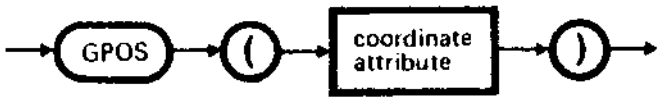
where:

x — is an integer which specifies the number of pixels involved in the graphic cursor movement.

Characteristics — Pressing the CONTROL key together with an arrow key causes the cursor to move by 1 pixel.
In the case of the hardcopy key the corresponding ASCII code is not returned but the hardcopy operation is executed.

GPOS

This function returns the graphic cursor position, in world coordinates, in the current window.



where:

coordinate attribute

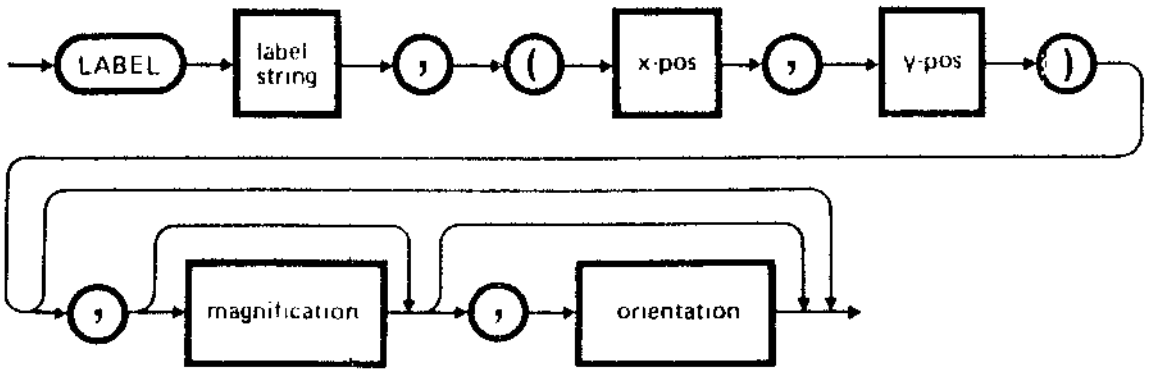
specifies whether the value relative to the x-axis or y-axis must be returned.
The permitted values are:

0 : returns the graphic cursor position on the x-axis

≠0 : returns the graphic cursor position on the y-axis

Displays a character string in a given position.

PROGRAM/IMMEDIATE Statement



where:

label string — is a printable character string to be displayed in a position specified by the parameters x-pos, y-pos.

x-pos, y-pos — are real expressions whose values are used by the LABEL statement to determine the position of the first character of the string. This position coincides with the lower left-hand corner of the first character.

magnification — is a whole number from 1 to 16. It represents the multiplying factor of the standard sizes of each character in the string. The default value is 1.

orientation — is a parameter which establishes the orientation of the string to be displayed.

The permitted values are:

0: the string is displayed parallel to the x-axis from the left to right

1: the string is displayed parallel to the y-axis from the bottom upwards

2: the string is displayed parallel to the y-axis from the top downwards

The default value is 0.

Characteristics

Printing characters in graphic mode is only possible with the LABEL statement.

If the parameters of a LABEL statement are such that the string fills the whole screen, or specified window, the statement is executed and only the position of the string inside the current window is displayed. The remainder of the string is clipped.

The characters are drawn using the last type of line set by the STYLE statement.

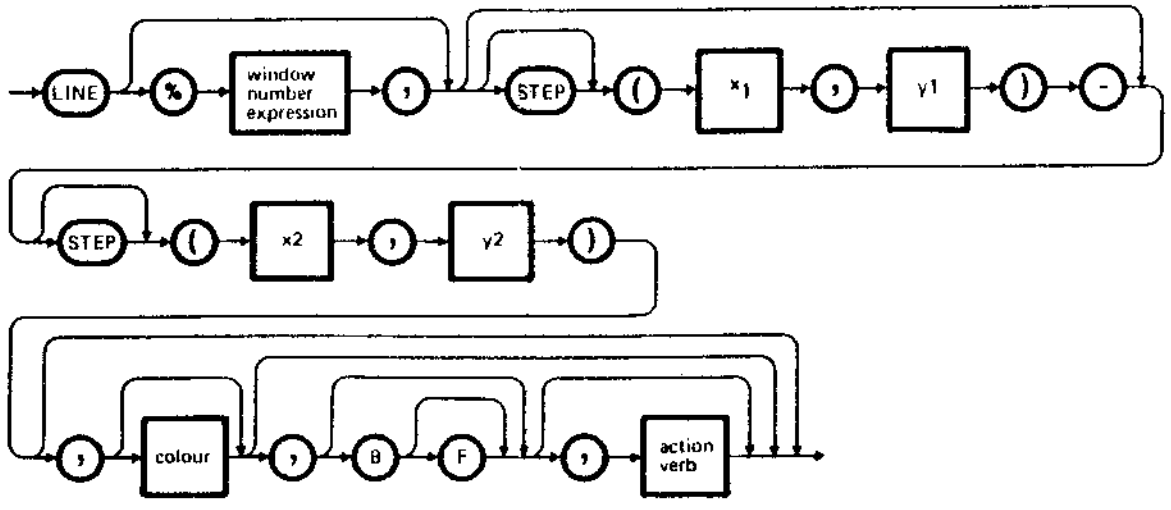
The current graphic position is updated to the (x-pos, y-pos) point.

Example

```
10 CLEAR
20 FOR I = 1 TO 9 STEP 3
30 LABEL "OLIVETTI",(190+I,150+I),6
40 NEXT
50 LABEL "OLIVETTI",(50,300),7
60 LABEL "LIVETTI",(30,320),7,2
70 END
```

Draws a line or a box.

PROGRAM/IMMEDIATE Statement



where:

window number expression — is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume see the window number variable parameter of WINDOW (Def) function. The default value is the current window.

STEP	is an optional keyword. This allows the use of relative coordinates. Through the use of STEP, the starting coordinates (x1, y1) become relative to the last point drawn or (in absence of such a point) to coordinates' origin. The final coordinates (x2, y2) are relative to the start of the line (or rectangle).
x1,y1	are real numeric expressions whose values specify the coordinates of the starting point of the line. If omitted, the line starts from the last point drawn.
x2,y2	are real numeric expression whose values specify the coordinates of the final point of the line.
colour	is a colour code specifying the colour with which the line or box will be drawn. See the same parameter of the COLOR statement for the values allowed and their meaning. The default value is the current foreground colour of the window. If the value is -1 the action verb parameter has no effect.
B(box)	is a parameter which allows the drawing of a box. The box (with its sides parallel to the x and y axes) has the coordinates (x1,y1) and (x2,y2) as its diagonal.
F(Filled)	is a parameter to be used only with the parameter B. BF draws a box which will be filled with the colour specified by the parameter colour.
action verb	is a parameter which may assume the following values: AND, XOR, OR, NOT, PSET, PRESET. PSET ensures that the line or the box is drawn in the specified colour. AND, OR and XOR indicate that the colour with which to draw the line or the box is the result of the corresponding logical operation, between the binary values of the specified colour code and the colour code of the pixels in the figure to be drawn on the screen. This operation is repeated for every pixel of the line or box. NOT indicates that the curves are drawn with the complement of the colour specified in the colour parameter. With PRESET the line or box is drawn with the background colour of the specified window. The default value is PSET. If the colour parameter has a value of -1 the action verb parameter has no effect.

Characteristics

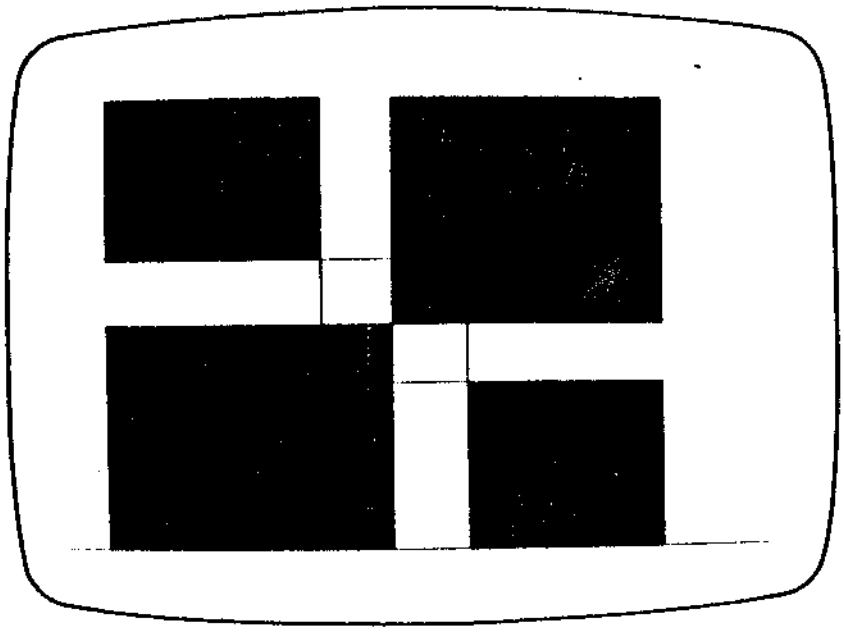
If the specified parameters for drawing a line or a box are such that part of the line or box falls outside the window boundaries, only the contents inside the window are displayed and what is outside is clipped.

The line or box are drawn using the last type of line set by the STYLE statement.

The graphic position is updated to the second (or only) end of the line drawn.

Example

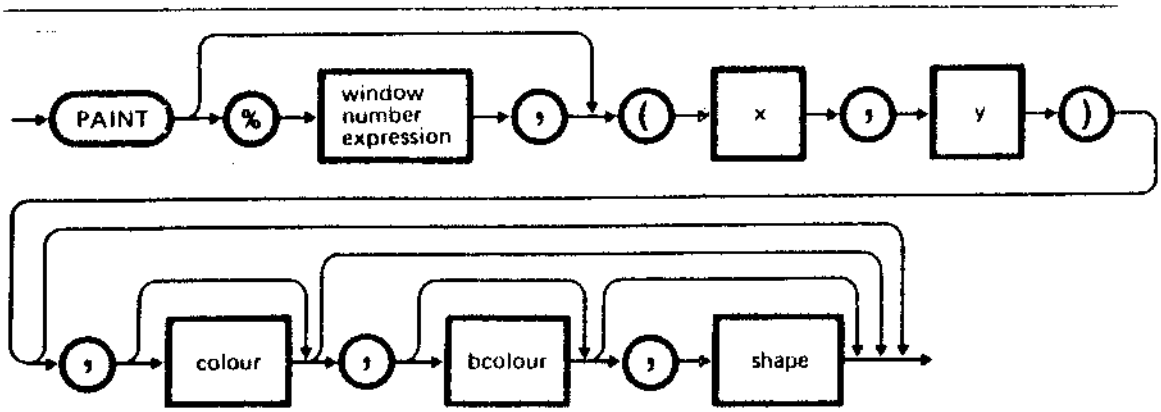
```
10 CLEAR
20 LINE (0,0) - (250,190),,BF
30 LINE STEP (250,190),,BF
40 LINE (0,0) - (500,380),,, XOR
50 LINE (185,140) - (320,245),0,B,NOT
60 LINE (320,140) - (500,0),,BF
70 LINE STEP(-500,380)- STEP(185, -135),,BF
```



PAINT

Paints or fills with a specified character the whole or a portion of a window, starting from the nearest pixel to the specified x,y co-ordinates.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x,y	are real expressions, whose values specify the coordinates of the pixel from which the operation starts.
colour	is a colour code specifying how to colour the whole or part of a window enclosed by a figure. For the values allowed for colour, see the same parameter of the COLOR statement. The colour code -1 is not allowed. The default value is the current foreground colour.

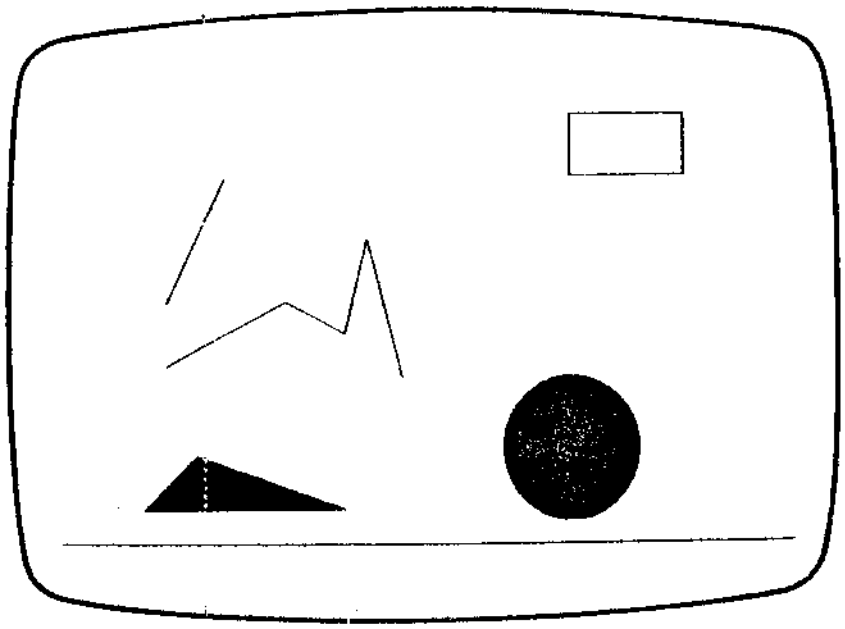
bcolour — is a colour code specifying the colour to be considered as border of the area to be painted. The colour code -1 is not allowed.
The default value is the current foreground colour.

shape — is the first element of a one-dimensional five-element integer array. The components of this array define the bitmap, which is an 8 x 10 pixel array, of the character which fills the figure.
When this parameter is activated the colour and bcolour parameters have no effect.

Characteristics — The current graphic position remains unchanged.

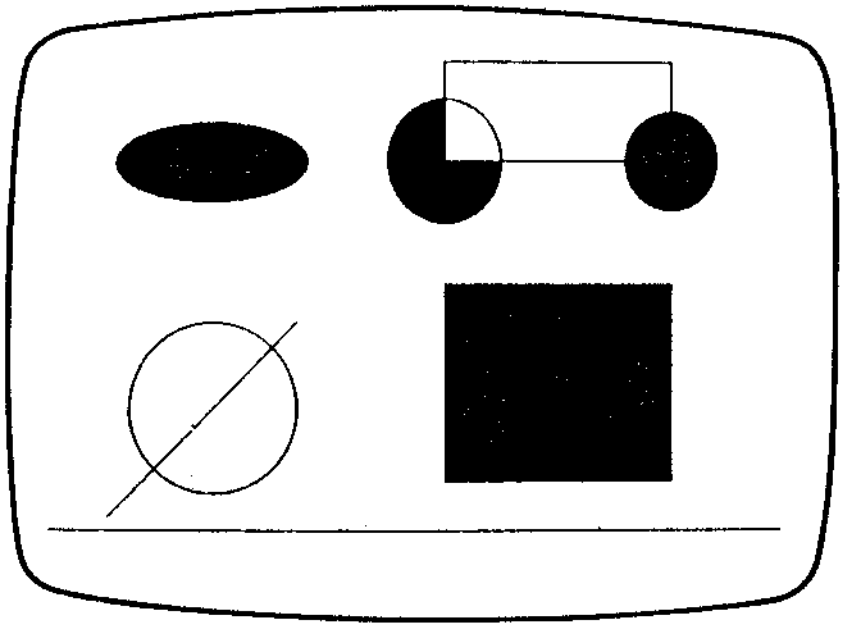
Example 1

```
5 CLS
10 LINE (50,200) - (100,300)
20 LINE (50,150) - (150,200)
30 LINE (150,200) - (200,175)
40 LINE (200,175) - (220,250)
50 LINE (220,250) - (250,140)
60 LINE (30,30) - (200,30)
70 LINE (30,30) - (75,75)
80 LINE (75,75)-(200,30)
90 PAINT (75,50)
100 CIRCLE (400,80),60
110 PAINT (400,80)
120 LINE (400,300) - (500,350),,B
```



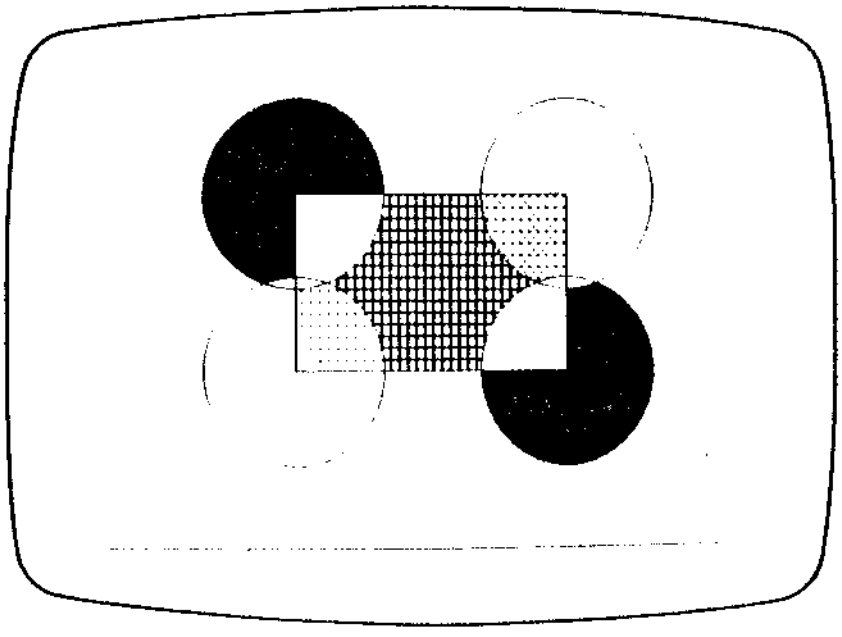
Example 2

```
5 CLS
10 CIRCLE (100,300),80,,5*.807
11 PAINT (100,300)
20 LINE (300,300) - (500,380),,B
30 CIRCLE (300,300),50
31 PAINT (290,290)
40 CIRCLE (500,300),40
41 PAINT (501,300)
42 PAINT (490,310)
50 CIRCLE (100,100),70
60 LINE (10,10) - (170,170)
70 LINE (300,40) - (500,200),,BF
```

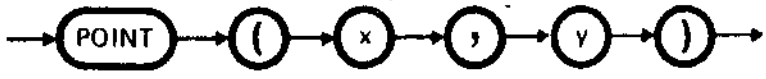


Example 3

```
10 CLEAR
20 LINE (150,150) - (400,300),,B
30 CIRCLE (150,150),80
40 CIRCLE (400,300),80
50 CIRCLE (150,300),80
60 CIRCLE (400,150),80
70 A%(0)=0:A%(1)=0:A%(2)=6168:A%(3)=0:A%(4)=0
80 PAINT (200,200),,,A%(0)
90 B%(0)=16416:B%(1)=4104:B%(2)=1799:
  B%(3)=2064:B%(4)=8256
100 PAINT (399,298),,,B%(0)
110 C%(0)=6168:C%(1)=6399:C%(2)=-232:
  C%(3)=6168:C%(4)=6168
120 PAINT (250,250),,,C%(0)
130 PAINT (150,310)
140 PAINT (410,150)
```



This function returns the colour code which has activated the pixel nearest to the specified (x,y) coordinates.



where:

x,y — are real numeric expressions whose values specify the coordinates of the pixel whose colour code must be known.

Characteristics — The current graphic position remains unchanged.

Example

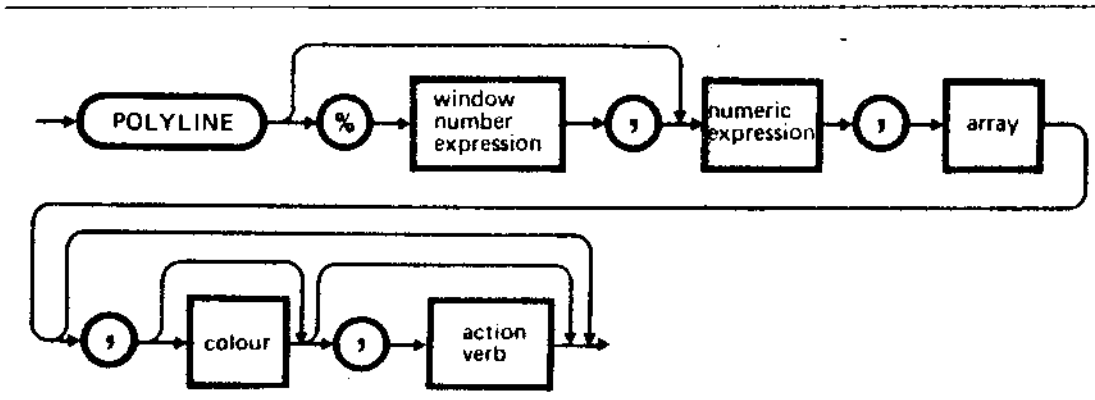
```
10 CIRCLE (50,50),20
20 PSET (50,50)
30 A% = POINT (50,50)
40 PRINT A%
```

Draws a circle on the screen with its centre at (50,50) and radius 20.
Sets the pixel nearest to (50,50) with the default foreground colour.
Assigns the value 1 to the A% variable. Displays the contents of A%.

POLYLINE

Draws a connected sequence of lines.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a numeric expression whose value is rounded to the nearest integer. It identifies the window on which the statement must operate. For the values that it can assume see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
numeric expression	is a numeric expression whose value rounded to the nearest integer provides the number of points from/to which the lines are drawn. The minimum value is 2 and the maximum value is 50.
array	is the first element of an integer or real array in simple precision; it contains the coordinates (x1,y1, x2,y2, ...) of the vertices of the segmented lines to be activated. The array must be previously defined.

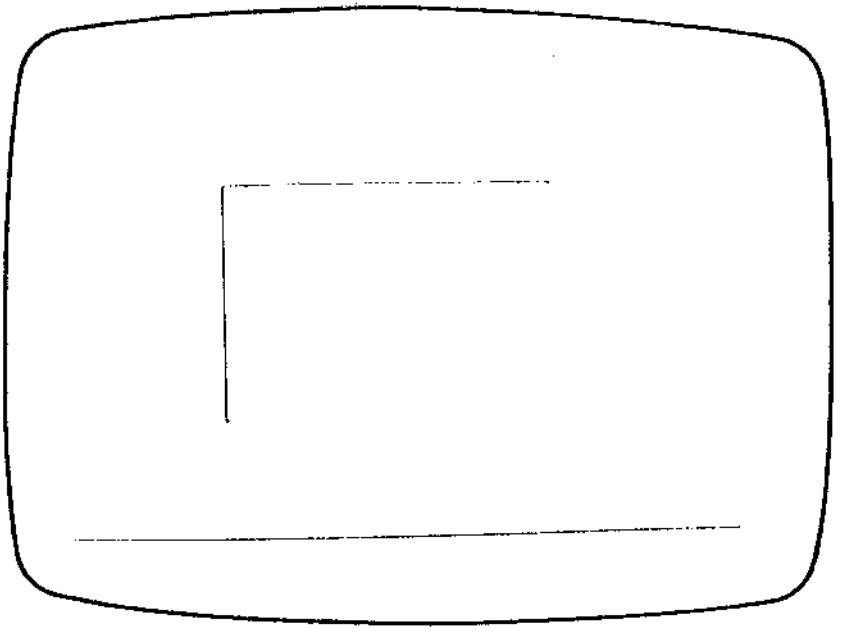
colour — see the LINE statement.

action verb — see the LINE statement.

Characteristics — The segmented line is drawn using the last type of line set by the STYLE statement.
The current graphic position is updated to the last vertix of the segmented line drawn.

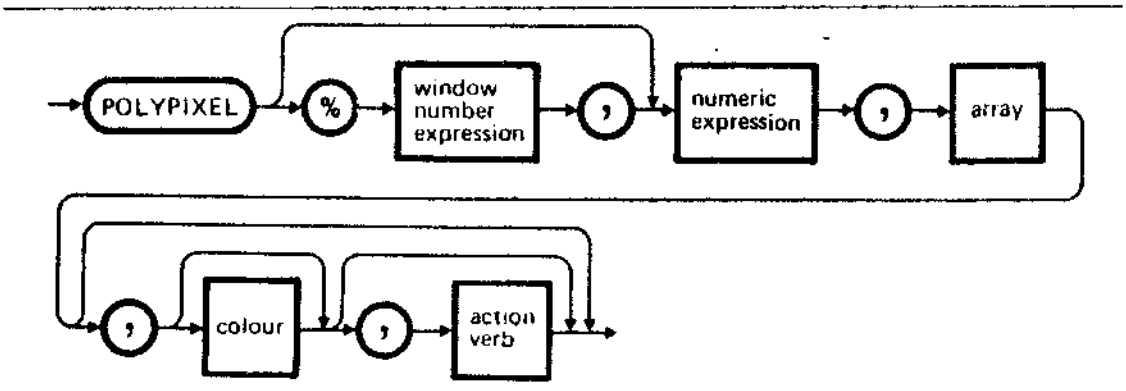
Example

```
10 CLEAR
20 A%(0)=100:A%(1)=100:A%(2)=400 :
A%(3)=300
30 A%(4)=100:A%(5)=300:A%(6)=100 :
A%(7)=100
40 POLYLINE 4, A%(0)
50 END
```



Sets pixels in the specified colour.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value is rounded to the nearest integer. It identifies the window on which the statement must operate. For the values that it can assume see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
numeric expression	is a numeric expression, whose value rounded to the nearest integer provides the number of pixels to set. The minimum value is 1 and the maximum value is 50.
array	is the first element of an integer or real array in simple precision: it contains the coordinates (x1,y1, x2,y2, ...) of the pixels to set. The array must be previously defined.

colour — is a colour code, which identifies the colour with which the pixels must be set.

See the same parameter of the COLOR statement for the values allowed.
The default value is the foreground colour. If colour has a -1 value the action verb parameters have no effect.

action verb — see the PSET statement.

Characteristics — The current graphic position is updated to the last point activated in the window.

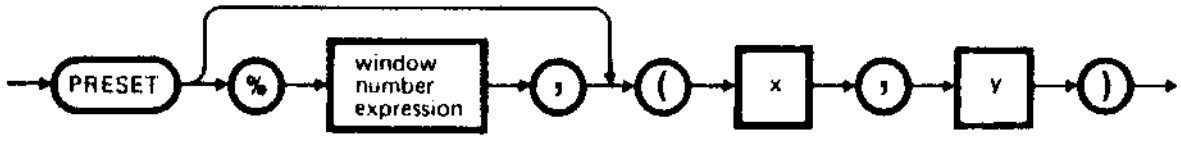
Example

```
10 CLEAR
20 A%(0)=100:A%(1)=100:A%(2)=200:A%(3)=200
30 A%(4)=300:A%(5)=300
40 POLYPIXEL 3,A%(0)
50 END
```

PRESET

Activates the pixel closest to the (x,y) coordinates with the background colour of the specified window.

PROGRAM/IMMEDIATE Statement



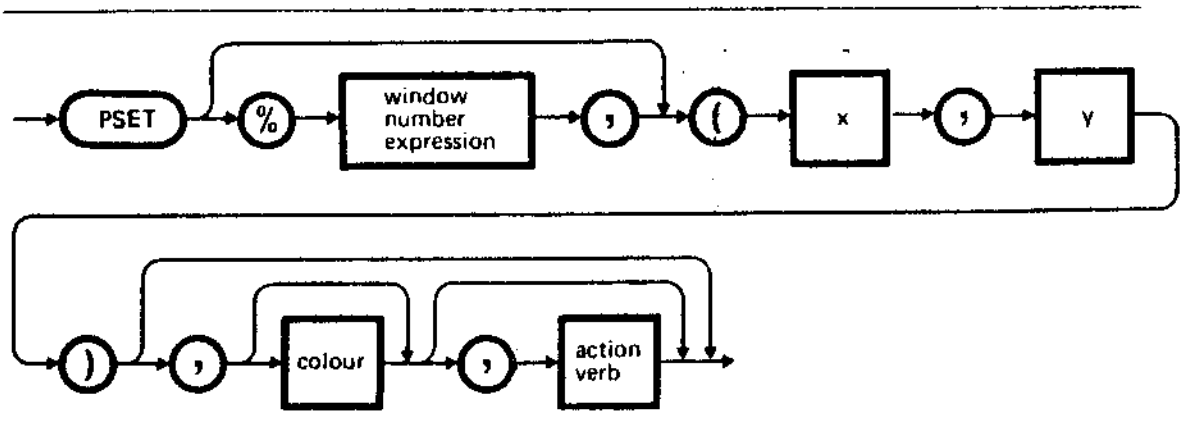
where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x,y	are real expressions whose values specify the coordinates of the point to be activated with the background colour.
Characteristics	The current graphic position is updated to the (x,y) point, if it is inside the window.
Note	If the specified coordinates are out of range, nothing will happen.

PSET

Sets the pixel closest to the specified (x,y) coordinates with a specified colour.

PROGRAM/IMMEDIATE Statement



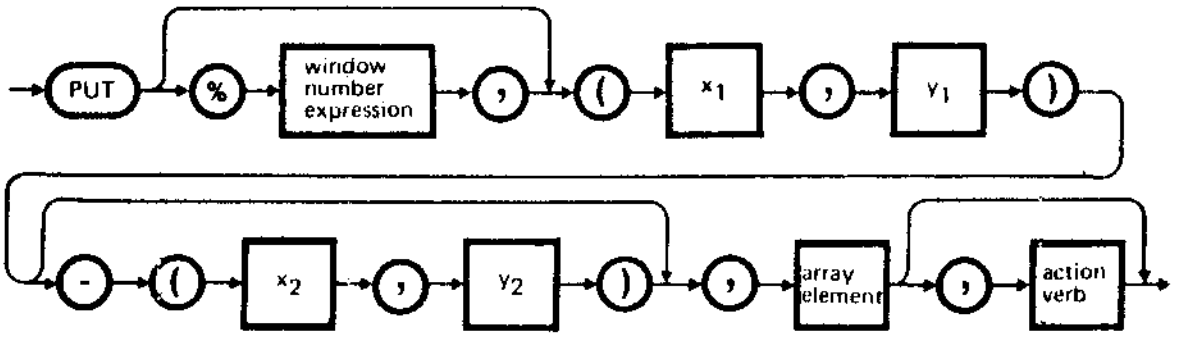
where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x,y	are real numeric expressions whose values specify the coordinates of the point to be activated.
colour	is a colour code which specifies the colour to set the pixel nearest to the x,y coordinates. See the same parameter of the COLOR statement for the values allowed. The default value is the current foreground colour. If colour has a -1 value the action verb parameters have no effect.

action verb	is described in the statements LINE and CIRCLE.
Characteristics	The colour parameter of the PSET statement does not change the current foreground or background colours of the specified window. The current graphic position is updated to the (x,y) point, if it is inside the window.

Displays an image previously stored in a one-dimensional integer array (16 bits per element).

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the PUT statement must operate. For the values that it can assume, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x1,y1, x2,y2	are real numeric expressions whose values specify the coordinates representing the end points of the diagonal of the box stored in the one-dimensional array. Its contents must be displayed on the screen. If the diagonal defined by these coordinates represents a box of a different dimension from that stored in the array, the box intersection is displayed. The default values for the x2 and y2 coordinates are those of the lower right-hand corner of the window being worked in.

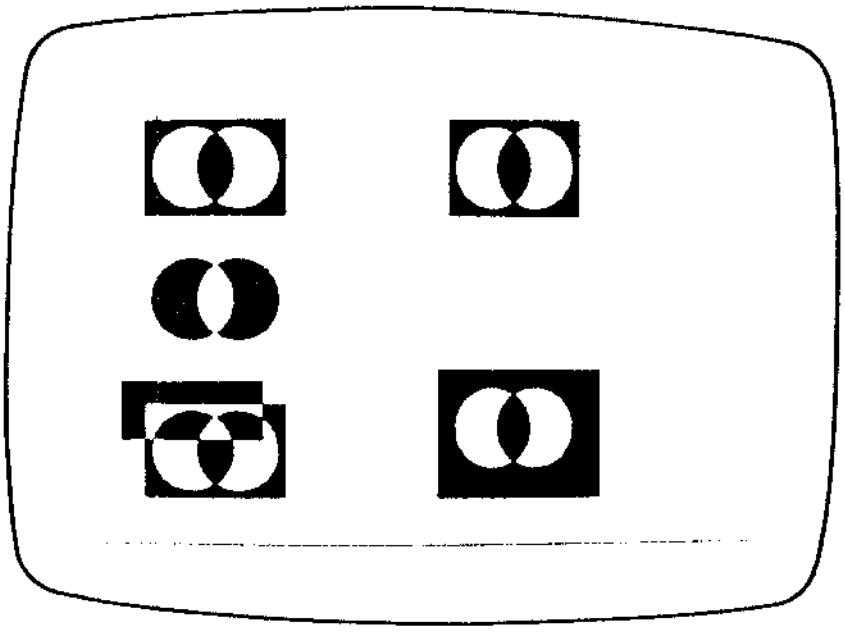
array element — is the first element of the one-dimensional integer array which contains the information stored with a GET statement.

action verb — is an optional parameter which can assume one of the following values: AND, XOR, OR, NOT, PSET, PRESET.
With the PSET option the displayed box is that stored in array.
The AND, OR and XOR operations specify that the box displayed is the result of the corresponding logical operation between the colour codes of the pixels in the array and the colour codes of the pixels already existing in the area to be occupied by the box on the screen.
The NOT option indicates that the complement of the colour codes of the pixels present in the array will be taken.
The PRESET option, however, indicates that the complement of the colour codes of the pixels on the screen will be taken.
The default value is PSET.

Characteristics — The current graphic position remains unchanged.

Example

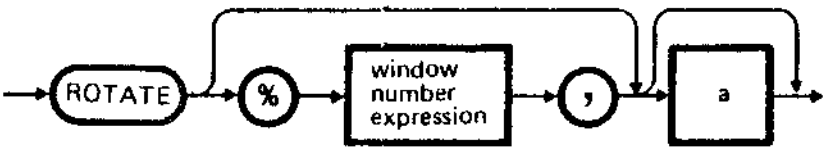
```
10 CLEAR
20 DIM A%(651)
30 LINE (30,280)-(150,360),,B
40 CIRCLE (110,320),35
50 CIRCLE (70,320),35
60 PAINT (31,321)
70 PAINT (100,320)
80 GET (30,280)-(150,360),A%(0)
100 PUT (300,360),A%(0)
120 PUT (30,250),A%(0),NOT
140 LINE (10,90)-(130,140),,BF
150 PUT (30,120),A%(0),XOR
170 LINE (290,40)-(440,150),,BF
180 PUT (300,140),A%(0),AND
190 END
```



ROTATE

Rotates the graphic axes.

PROGRAM/IMMEDIATE statement.



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, identifies the window on which the statement must operate. For the values allowed, see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
a	is a real numeric expression whose value must be between -2π and $+2\pi$. It defines the rotation angle, in radians, measured anticlockwise, starting from the x axis. If omitted, the axes are rotated in parallel to the window borders.

Characteristics

When this statement has been executed, the successive graphic statements are executed according to the rotated axes.

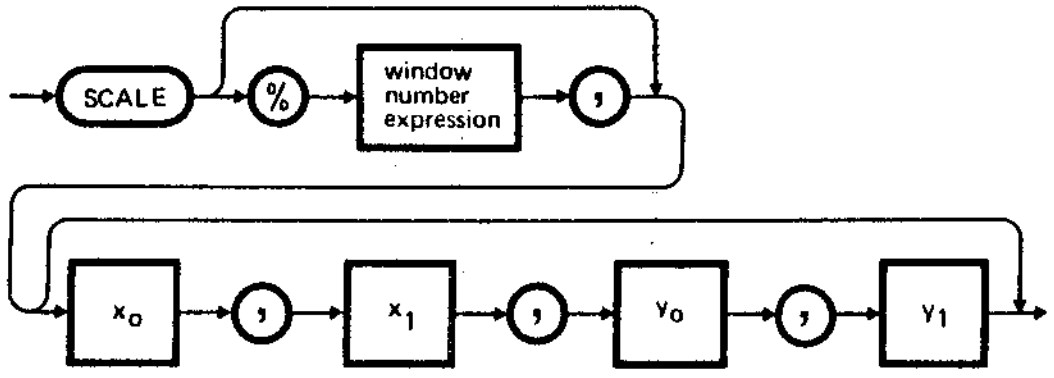
In the GET and PUT statements, the scaling is, however, applied only to the extremes of the diagonal of the box to be stored; the bottom of this box is always horizontal, and not parallel to the rotated x axis.

The SCALEX and SCALEY functions cannot be called after a ROTATE statement.

The SCALE statement cancels the effects of the ROTATE statement.

Defines the scaling between the world coordinates and the device coordinates.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, specifies the window on which the statement must operate. For the values that it can assume see the window number variable parameter of the WINDOW (Def) function. The default value is the current window.
x0,x1, y0,y1	are the window dimensions (in world coordinates)

x0: abscissa of the left-hand side of the window; default value is 0

x1: abscissa of the right-hand side of the window; default value is 559 or 639

y0: ordinate of the bottom of the window;

default value is 0

y1: ordinate of the top of the window;
default value is 383

Characteristics

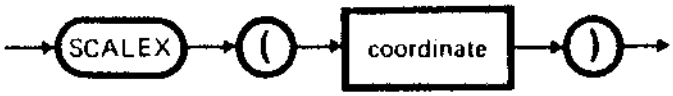
x1-x0, y1-y0 can be either positive or negative, but must never be equal to zero.
After having executed a SCALE statement, each time the user expresses the value of a coordinate, this is entered in world coordinates.
The coordinate system is the default one if:

- no SCALE statement has been executed
- the SCALE statement has been executed without parameters.

The execution of the SCALE statement cancels the effects of the ROTATE and TRANSLATE statements.

SCALEX

This function converts a world coordinate into the associated device coordinate on the x-axis.



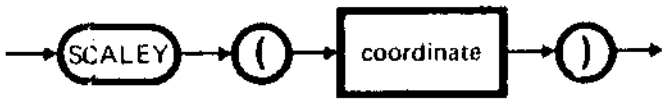
where:

coordinate — is a world coordinate on the x-axis.

Note — The SCALEX function cannot be called after a ROTATE statement.

SCALEY

This function converts a world coordinate into the associated device coordinate on the y-axis.

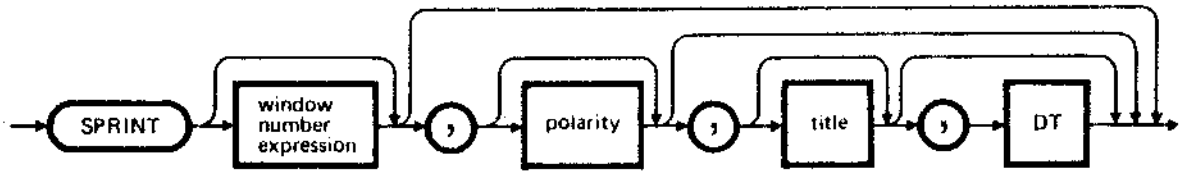


where:

coordinate	is a user coordinate on the y-axis
Note	The SCALEY function cannot be called afther a ROTATE statement.

Carries out the hardcopy (only on PR2400) of the whole screen or of a specified window.

PROGRAM/IMMEDIATE Statement



where:

window number expression	is a real numeric expression that is rounded to the nearest integer. It specifies on which window the hardcopy must be carried out. The hardcopy is centred. The value 0 allows the hardcopy of the whole screen. For the values allowed see the window number variable parameter of the WINDOW (Def) function. The default value is the hardcopy of the current window.
polarity	describes printing in monochromatic terms. It can assume two values: N or P. N (negative) associates the black print on paper to the pixel on, and P (positive) associates printing to pixel off. The default value is N.
title	is an alphanumeric string. It must not exceed 60 characters otherwise it is clipped and it is printed in the left-hand corner as graphic output title. The default value is no title.

DT

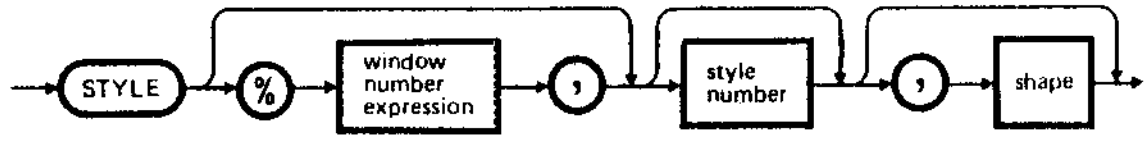
is a keyword which determines whether or not to print the date and the time at the top-right of output.
If omitted neither the date or time is printed.

Note

The hardcopy of a graphic image is only possible on the PR2400 printer.
It must be remembered that, with this printer, a hardcopy with a maximum of 560 horizontal pixels can be obtained. An eventual hardcopy of an area of screen greater than 560 pixels will be cut to the right.
The BOUNDS statement means a number less than the physical number of pixels can be used; when BOUNDS is executed with the width parameter equal to 560, a complete hardcopy of the graphic image is obtained.

Defines the type and thickness of the line.

PROGRAM/IMMEDIATE statement.



where:

window number expression	is a real numeric expression whose value, rounded to the nearest integer, selects the window to be worked on. See the window number variable parameter of the WINDOW (Def) function for the values allowed. The default value is the current window.
style number	is a real numeric expression whose value, rounded to the nearest integer, defines the type of line. The values allowed are between 1 and 7.
shape	is the first element of a one-dimensional integer array of 4 elements which defines the form of the line. Each element indicates, in order, the number of pixels to be activated and the number which must remain deactivated. The 0 value indicates the start of a repetition. Any further element is not considered. If style number =1, the first array element defines the line thickness and must be between 1 and 256. The default is 1. If shape is omitted, the last line-type set by that particular style number remains active. Otherwise,

the default values are used.
The default values, in hexadecimal, are given in the following table to clarify the sequence of pixels on and pixels off.

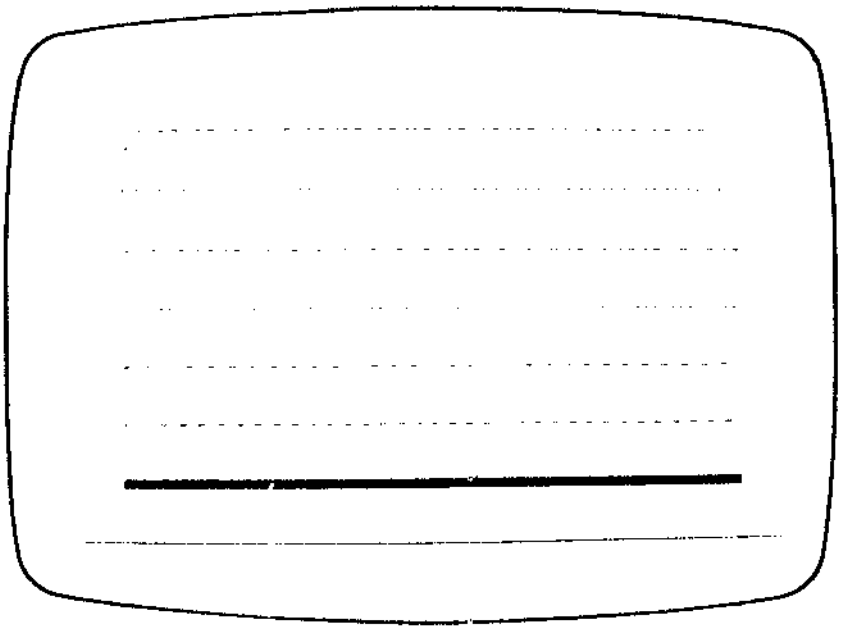
```
style 2 : &H0808;&H0000;...
style 3 : &H0804;&H0204;&H0000;...
style 4 : &H0404;&H0000;...
style 5 : &H0808;&H0404;&H0000;...
style 6 : &H0408;&H0404;&H0000;...
style 7 : &H0408;&H0000;...
```

Note

When this statement has been executed, all the straight lines will be drawn in the defined style. If the parameters are omitted the continuous thin line is restored.

Example

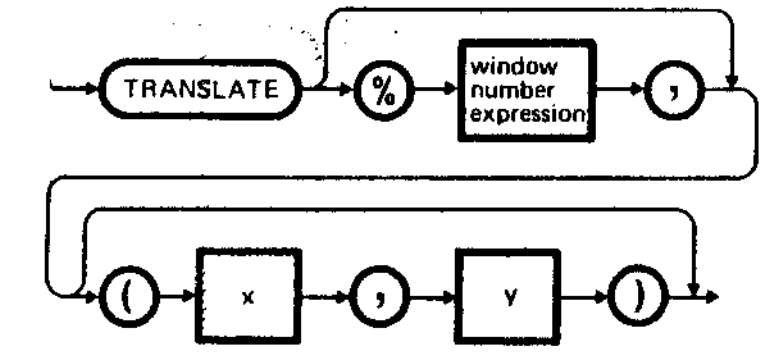
```
 10 CLEAR
 20 A%(0)=8
 30 STYLE 1,A%(0)
 40 LINE(0,50)-(559,50)
 50 STYLE 2
 60 LINE(0,100)-(559,100)
 70 STYLE 3
 80 LINE(0,150)-(559,150)
 90 STYLE 4
100 LINE(0,200)-(559,200)
110 STYLE 5
120 LINE(0,250)-(559,250)
130 STYLE 6
140 LINE(0,300)-(559,300)
150 A%(0)=&H505 :A%(1)=&H808 :A%(2)=H303:A%(3)=&H0
160 STYLE 4,A%(0)
170 LINE(0,350)-(559,350)
180 END
```



TRANSLATE

Translates the origin of the coordinates system

PROGRAM/IMMEDIATE Statement.



where:

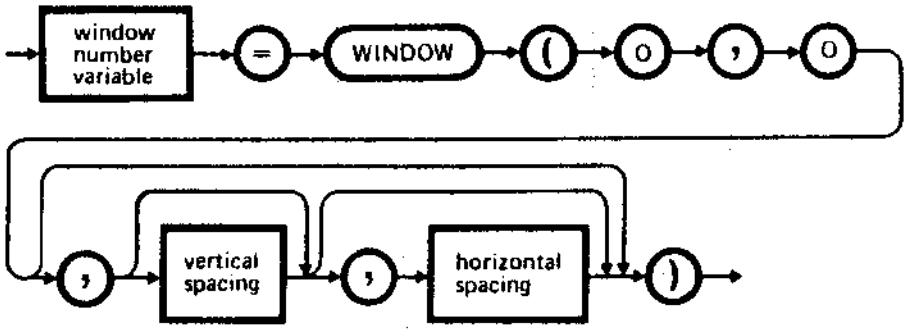
window number expression	ia a real numeric expression whose value, rounded to the nearest integer, identifies the window on which the statement must operate. See the window number variable parameter of the WINDOW (Def) function for the values allowed. The default value is the current window.
x	is a real numeric expression whose value is the abscisse of the new origin of the coordinate system.
y	is a real numeric expression whose value is the ordinate of the new origin of the coordinates system.

Characteristics

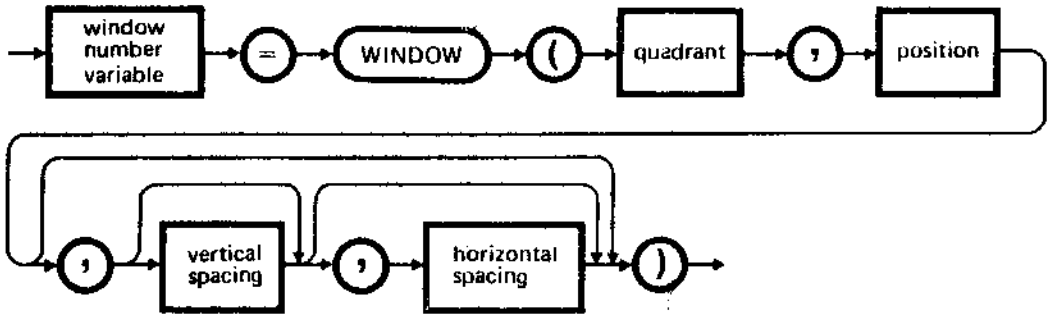
If the x and y parameters are omitted, the origin defined with the last SCALE statement is restored or, in the absence of this, the default origin.

The execution of SCALE cancels the effects of TRANSLATE.

It is accepted only for reasons of compatibility.



This function opens a new window by splitting the current window. The current window is the one on which the user is operating.



where:

window number expression — is an integer variable to which BASIC assigns a value identifying the window to be opened. This value is an integer in the range (1 - 16). The values are assigned by BASIC in ascending order. The smallest available number will always be assigned to the new window.
The window that initially corresponds to the whole screen is identified by the number 1 even if it is successively split into other windows.
When a new window is opened, the current window that created it is also called "parent window".

quadrant — specifies in which part of the "parent" window a new window will be opened.

There are four possibilities:

0: the new window will be the top of the parent window

1: the new window will be the bottom of the parent window

2: the new window will be the left part of the parent window

3: the new window will be the right part of the parent window.

position — is a parameter which defines the position where the current window is to be split to open a new window.

If the value of "quadrant" is 0 or 1 a horizontal split will be made. In this case, the position parameter is an integer number of scanlines, calculated from the top of the current window (16 scanlines for each line of text). This number is a multiple of 16; in the opposite case, the number is automatically rounded to the next lowest multiple of 16.

If the value of "quadrant" is 2 or 3 a vertical split will be made. In this case, the position parameter is the number of alphanumeric columns (8 pixels per column) calculated from the left border of the parent window.

If position is equal to -1, the parent window is split in half (vertically or horizontally, depending on the value of quadrant).

vertical spacing — is accepted for compatibility.

horizontal spacing — is accepted for compatibility.

Characteristics — The previous contents of the screen which will become the new window are cleared. The background and foreground colours and the cursor assumed are those of the parent window.
The graphic cursor and the current position are given as (0,0) both in the window just opened and the window which has been split.

Example 1

```
A = WINDOW (0,100)
```

Splitting is horizontal, the new window becomes the top part of the current window.

The height of the window to be opened is 96 scanlines, corresponding to 16 alphanumeric lines.

Example 2

```
B = WINDOW (2,50)
```

Splitting is vertical, the new window becomes the left-hand part of the current window.
The width of the window to be opened is 50 character positions.

Example 3

```
A = WINDOW (0,100)
PRINT A
```

It is always possible to know the integer number with which the BASIC identifies a window.

The WINDOW statement opens a window which the user identifies with the variable A. The BASIC assigns an integer value to this variable.

The contents of A is then displayed with the statement PRINT A.

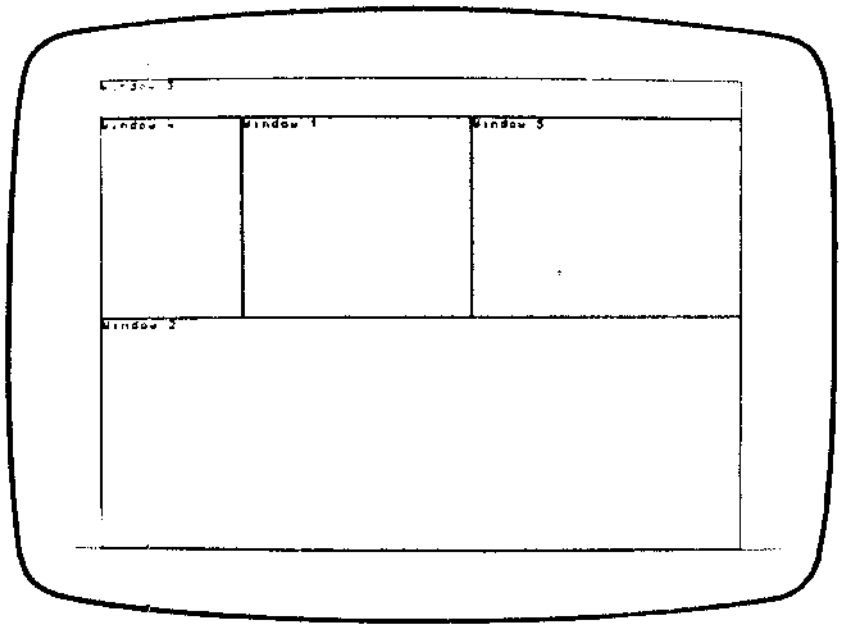
Example 4

```
A=3 : D=40
W=WINDOW (A,D)
```

Splitting is vertical. The new window is the right-hand side of the parent window. The length of the window to be opened is 40 character positions.

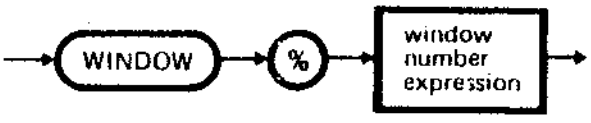
Example 5

```
W2 = WINDOW (1,205)
W3 = WINDOW (0,40)
W4 = WINDOW (2,20)
W5 = WINDOW (3,25)
```



Selects the window on which to operate.

PROGRAM/IMMEDIATE Statement



where:

window number expression — is a real numeric expression whose value is rounded to the nearest integer. It identifies a previously opened window and selects it. For the values that it can assume, see the corresponding parameter of the WINDOW (Def) function.

Example 1

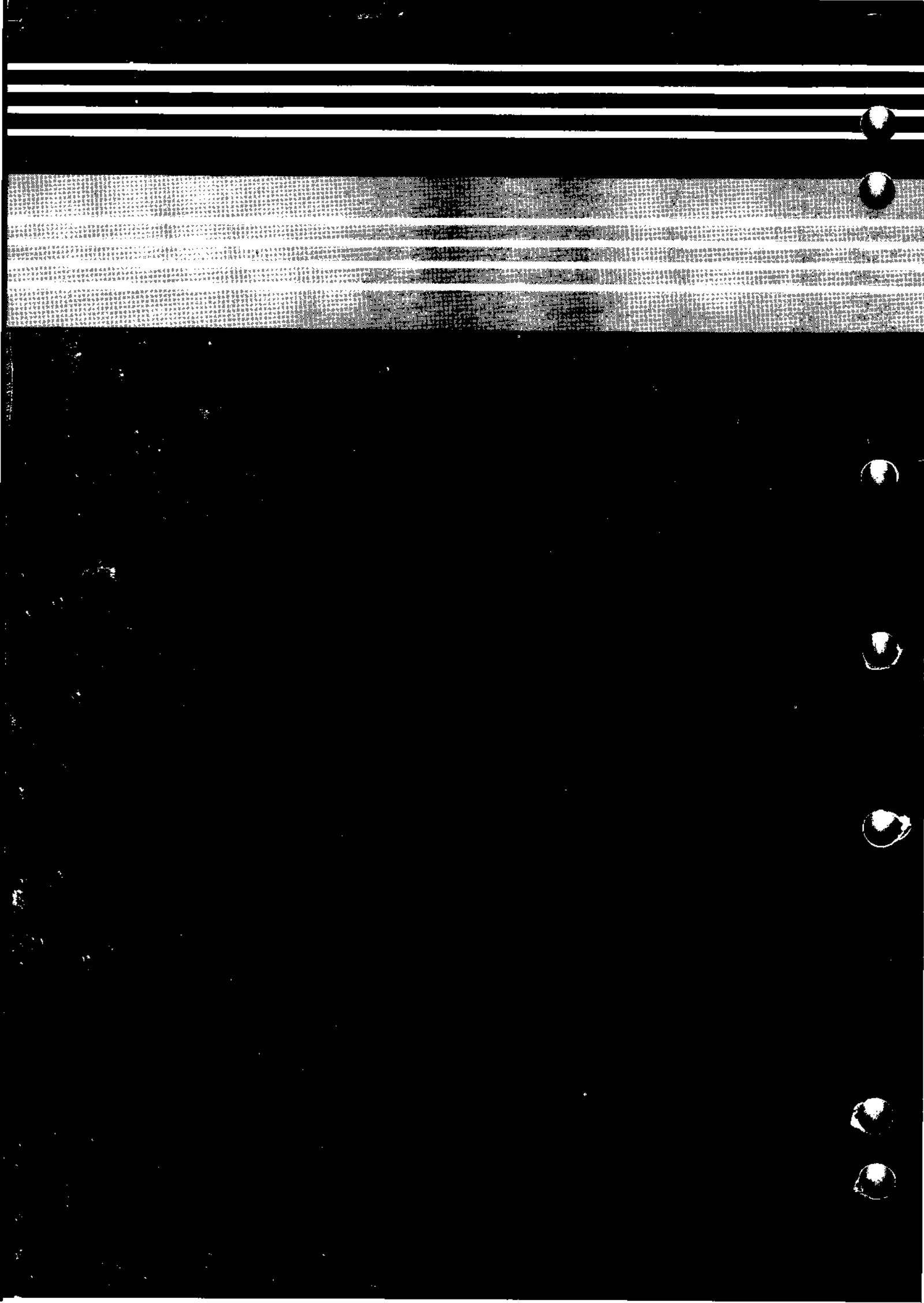
```
10 X = 5
20 A = ABS(SIN(1.5) + SQR(X1))
30 WINDOW %A
```

The system selects the window that is identified by the integer value contained in the variable A.

Example 2

```
WINDOW %1
```

The system selects the window identified by the number 1. As previously mentioned, it is the main window.



M30 M40

BASIC Language
User Guide

olivetti L1