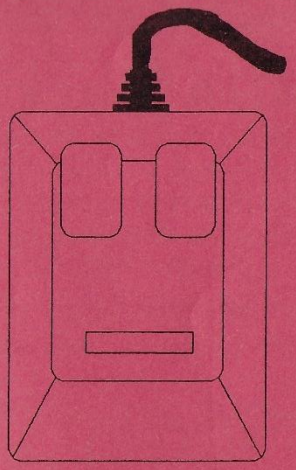
GIGA
Basic & Desk

User Guide



Gerd-Uwe Neukamp & Andre Claassen

NOTE:

Before starting work with the supplied software package you should copy your files. To do so, read the following notes.

GIGA Basic:

Insert your original copy of GB in drive 2 and enter the following line:
'exec_w mdv2_clone_exe'
When asked to enter the drive to copy to, enter 'mdv1_' (or even 'flp1_'!).

GIGA Desk:

Insert the original cartridge in drive 1 and a blank cartridge in drive 2. Then type: **'lrun mdv1_clone'**
When asked to enter the drive to copy to, enter 'mdv2_' (or even 'flp1_'!).

I hope you will have fun using this package!

If you have any problems with the enclosed cartridges, send your original copies and a cheque of German DM10,-(for p&p) to:

Gerd-Uwe Neukamp
Heresbachstr. 1
4000 Duesseldorf 1
Western Germany

Don't forget to specify your address!

We will send you the replacement immediately.

E.asily A.pplicable S.ystem E.nvironment

DISCLAIMER

The program was carefully developed and reproduced, neither the author nor the distributors, however, can guarantee that the program and this guide are free from errors. Therefore GIGA- SOFT under no circumstances will be liable for any direct, indirect, incidental or consequential loss of use, stored data, profit or contracts which may arise from any error, defect or failure of this software.

GIGA- SOFT has a policy of constant development and improvement of their products. We reserve the right to change manuals and software at any time and without notice.

Start of program

After you have pushed the reset button at the right upper edge of your QL, the well- known power- on message will appear. Now insert the enclosed microdrive cartridge and push 'F1' or 'F2', which depends on the screenmode you prefer. The program will automatically be booted. To start directly enter the following command:

```
LRUN mdv1_BOOT
```

Then the program will be loaded and started.

Introduction

E.A.S.E. is a totally new desktop surrounding for the SINCLAIR QL (R). It is an overlay of the operating system QDOS (R) of the QL and allows the user to carry out all functions of QDOS, such as the copying of files or the starting of programs in a new, easier manner. The times of

```
'COPY mdv1_text_doc to mdv2_text_doc'
```

have therefore been overcome. E.A.S.E. works with symbols and a mouse; so faulty inputs are practically impossible. One of the most powerful advantages of E.A.S.E. is the possibility of starting programs directly from this environment. These are then loaded and, after finishing the work with them, you will find yourself again in E.A.S.E.. This feature will work with an unexpanded QL, this however will mean that E.A.S.E. has to reload itself. For users with a memory- expanded QL this procedure is not necessary,

E.A.S.E. automatically checks the remaining amount of free memory and remains in place, if the program to be loaded fits in this space. In this case, E.A.S.E. will work much faster. The only drawback is the fact, that the supported PSION (R) programs in the versions up to including 2.00 do not set free the whole amount of memory they have used. This will result in the fact, that, if you frequently change the programs you work with, free memory will decrease drastically. However, PSION has promised to remove this error from version 2.03 on. A ready- to- use mouse is supplied with this package (refer to the guide). In the future, more programs with the same user- interface will follow. One of these will be a fount editor, which will be completely mousedriven. Ask your local software dealer.

Handling

The handling of E.A.S.E. is very simple. The essential tool is the arrow on the screen, which can be moved with the mouse device. You can point at certain symbols on the screen; pushing the mousebutton will then activate the corresponding function. If, for example, you wish to load a program from 'mdv1_', you point at the symbol of drive one in the symbolline on the righthand side of the screen and push the button. Immediately, part of the screen will be separated and the directory of drive one will appear in this window. Each file will be represented by a name and a symbol, called 'icon'. This symbol depends on the type of the file. There are ten different symbols available, supporting all the files generated by the PSION (R) programs Quill (R), Easel (R), Archive (R) and Abacus (R) and more. Now you select the file you want to load in this window; the file will be marked. Pointing at the uppermost textline, select the word 'File' and press the button. A menu will fall down the textline, containing the options 'Copy', 'Delete', 'Format', 'Start' and 'Info'. Now select 'Start', the appropriate line will be inverted, and push the button again. E.A.S.E. will check the program to see whether the selected file is an executable program ('executable program' in this sense does not mean a Basicprogram, which can be started by E.A.S.E., too, but a file which can be started by Superbasic (R) by using the 'EXEC'- command. Such programs are the four supplied with your QL (only from version 2.00 on!).). If the file is executable, E.A.S.E. informs the user that this procedure will close all open windows and asks for confirmation. If you affirm this, the program will be loaded and started. This description reads longer than the actual process lasts. Now it is up to you to experiment with the system.

Commands

Iconcommands

If E.A.S.E. has been loaded and if you have affirmed the startup message, you will find yourself with an empty screen with a bordered line on the top, the so called 'pulldown-menu', and some funny- looking symbols at the right border. These symbols or 'icons' are described in the following section. We will go from the uppermost symbol downwards. The functions expressed by the symbols always refer to the last chosen window.

1st symbol:

The first symbol is a small box with arrows pointing into all directions. This symbol is used to move a window over the screen. Supposing, you have a window on the screen and select this icon, a small box will appear which can be moved by using the mouse. After fixing the box to a new position, using the button, the window will change its position to the new one immediately, the contents remaining the same. If you have more than one open window, you can select the one you want to move by simply pointing at it and pushing the button. If another window overlays this one, the selected window will change its position to the foreground.

2nd symbol:

This symbol represents a small box, which becomes larger in the lower right corner. This option allows the user to change the format of the selected window. If this option is activated, a small box will appear, which can be reduced or enlarged by using the mouse. The new size can be fixed by using the button, the background automatically being refreshed. This function can be used to minimize the size of a window, which is not used for the moment, and to enlarge it, if you want to work with it again.

3rd symbol:

This one shows a box containing arrows pointing into all directions. This function is the one which allows us to use the word 'windows'. E.A.S.E. supports 'real' windows, which means windows that can be scrolled into every direction. Also the contents of a window remain the same, whether you overlay it with another one or move it over the screen. A window is similar to a small frame through which you can see part of a greater picture. This frame can be moved over the picture to give you the possibility of seeing its whole contents. This is the only possibility to get round the restictions of the screenformat of the QL and to work with pages eight times the

size of the QL's screen. To quit this option have a second push on the button.

4th symbol:

This symbol shows a magnifying glass. The function can be used to enlarge a window to maximum size and to reduce it afterwards to its former state.

5th symbol:

This function is unique to E.A.S.E.. It allows the user to open another window upon the same page as the window selected. This gives you the possibility to work on seperate areas of the page simultaneously. The number of open windows is restricted to a maximum of seven. However this restriction will never affect your work, because the screenlayout would become very complex with too many windows beeing used. You will soon see, while working with E.A.S.E., that less might be more.

6th symbol:

This function is reverse to the one above. It removes a window from the screen. To prevent you from clearing a window erroneously, E.A.S.E. will ask for a safety affirmation before actually closing the window. Keep in mind that closing the last window upon a page will actually remove the page, too.

7th symbol:

The floppy- disk symbol with the number one opens a window and displays the directory of drive one.

8th symbol:

The directory of drive two

9th symbol:

The floppy- disk symbol with the question mark gives you the possibility to have access to other media than the standard ones. When using this command, E.A.S.E. asks you to enter the explicite name of the medium you want to have access to (e.g. 'flp1_', 'hdk1_' or 'mdv3_').

Pulldownmenu

The so- called 'pulldownmenu' is the box at the top of the screen, labeled with 'Desk', 'File', 'Options' and 'Exit'. Each of these words is the headline of a number of

terms. If you point at one of the four and press the button, a small window will fall down, containing some options. Moving the arrow over these options inverts the lines. Select the option you want to activate by pressing the button again. This reads more complicatedly than it is in reality.

Desk

The first pulldownmenu contains some options to have access to some tools like a calculator and a game.

About E.A.S.E.: Displays a copyright- message.

Calculator: This command activates the inbuilt pocket calculator. The calculator supports scientific functions, the format of in- and output being the same as the one of Superbasic (R). This means that negative values have to be entered with a preceding minus sign, the exponential part has to be marked with an 'E'. The input medium is the mouse. Point at the key you want to press and push the button. The calculator layout shows the minus sign twice. The lower left is used to enter the sign, the other to start the mathematical function. To enter the exponent use the 'E'.

Some examples of numbers and how to enter them:

0.5×10^{5} : 0.5E5
-3×10^{-3} : -3E-3

You will find the '='- key missing. With this calculator, we used the technique of the so called 'Reverse Polnic Notation' (RPN), which is in a similar form used by the calculators from Hewlett- Packard (R) and by the programming language Forth. In this notation, you first must enter the numbers you want to work with and then the operation to perform with these ones. The numbers you enter are arranged in the calculator's memory by using a structure called 'stack'. This structure is called that way, because it is similar to a stack of plates. The one, you put on the stack last, will be the first to be put off. Using the stack in another way will result in many broken pieces. This structure is called in technical terms

'first in/ last out' or 'last in/ first out'. You can put a number on the stack by using the 'ENTER' command or remove a number from the stack by using 'C'. Operations working on two operands remove one item from the stack in the sense that, before you select the operation, there have to be two values on the stack. Afterwards there will only be one item on it, the result of the calculation. Operations of this type are addition, subtraction, multiplication, division and raising the power. Operations working with one operand, like the trigonometric functions (they are working in degrees!) only modify the value on top of the stack. Output functions, like 'PI' or 'MR' add a new value to the stack.

Examples:

Calculation:	Input	Stack	(<- bottom	top ->)
3+5	3	0	0	0
	ENTER	0	0	3
	5	0	0	3
	+	0	3	5
		0	0	8
sin (30)	30	0	0	0
	ENTER	0	0	30
	SIN	0	0	0.5
sin ((3+7)*(1+2))	3	0	0	0
	ENTER	0	0	3
	7	0	0	3
	+	0	3	7
		0	0	10
	1	0	0	10
	ENTER	0	10	1
	2	0	10	1
	+	10	1	2
		0	10	3
	*	0	0	30
	SIN	0	0	0.5

Working through the above examples and testing calculations of your own you will soon get familiar with RPN. Now the description of the command- set:

+	Addition

-	Subtraction
*	Multiplication
/	Division
^	Raising the power
SQR	Squareroot
SIN	Sine
COS	Cosine
TAN	Tangent
ASN	Arcsine
ACS	Arcosine
ATN	Arctangent
LN	Natural logarithm
LOG	Logarithm to base ten
EXP	e^x
PI	Output of pi (3.1415...)
C	Take uppermost number from stack
E	Input of the 'E' sign
ENTER	Put number on stack
MS	Displayed number to memory
MR	Displayed number from memory
MC	Clear memory
M+	Add number to memory
M-	Subtract number from memory
X-Y	Swap the two uppermost values of the stack

The stack used with the calculator supports up to four levels, which is enough in all cases. To store intermediate values, you can use the memory. The consequent usage of the window technique allows you to open up more than one calculator at the same As Because every calculator has its own stack and memory, this gives you the advantage of working with more than one simultaneously.

Game:

E.A.S.E. contains a game that you will know from your childhood. It is a little puzzlegame, consisting of fifteen pieces, numbered from one to fifteen in a grid of four times four with a small gap giving the possibility to shift the pieces. When activated, the game will be in the order, you will have to restore afterwards. To shuffle the game, you can select the option 'SHUFFLE' at the lower left edge of the game. To move a piece, simply point at it and press the button; the computer will automatically determine the direction to shift to. To quantify your skill, a field appears which displays the

number of moves you needed. If the above instructions about the calculator have bored you, you can now relax playing puzzle (I need about 130 moves.). You also can activate more than one game simultaneously and play puzzle with other persons.

Panel: When you have finished playing, we will go on. This command activates a menu of its own, which can be handled like a pulldownmenu. It may be used to alter some system parameters, like the default drives or the printer device.

The commands in extenso:

Quit panel: The default option. Quits the panel without doing anything. It is used, if you activated the 'Panel' command erroneously.

Default medium: The default media for E.A.S.E. are the microdrives. If you work with a disc-station, this command can be used to switch to 'flp', the diskette symbols from now on giving the directories of 'flp1_' and 'flp2_'.

Printer device: Similar to the above, this command can be used to alter the printerdevice, which is standardly 'ser'.

Stepsize: Alters the scrollspeed of the windows.

Movesize: Alters the speed with which windows can be moved over the screen.

High resolution: Activates the four- colour- mode on the QL. Whenever possible, you should work with this mode, since it needs a lower amount of the rare memory than the other one.

Low resolution: Switches to low resolution.

The changing of the screenmode will close all open windows, which results in a loss of all processed data. An erroneous usage of these commands is therefore prevented by an affirmation request.

File

Copy: Copies all files which are marked in the uppermost directory window. E.A.S.E. will ask you to enter the medium to copy to, this can be entered by selecting the appropriate icon.

Delete: Deletes all files which are marked in the uppermost directory window. Before every deletion, E.A.S.E. asks for an affirmation, which has to be answered by using the mouse. To switch this request off, select the item 'All', to quit the deletion, select 'Quit'.

Format: Prepares a new medium for use on the QL, respectively clears all files from a previously used medium. E.A.S.E. asks the user to specify the medium to format, which has to be entered in the same way as in the 'Copy' command, and to enter the mediumlabel. As this command erases all data from the medium, you should use it very cautiously E.A.S.E. will ask you whether you really want to format the medium.

Start: One of the most powerful commands of E.A.S.E.. It gives the possibility to call other programs from E.A.S.E.. After finishing work with the other program, you will find yourself back in E.A.S.E.. If you have to change the microdrive cartridge during your work, E.A.S.E. asks you to enter a cartridge containing the file 'desk_bin'. It is not necessary to insert your original copy. The only files you can start from E.A.S.E. are those which can also be started from Superbasic using the 'EXEC' command and Superbasic programs. If E.A.S.E. recognizes a file as not being executable and does not find the extension '_BAS', it asks you what to do. There are four possibilities, which are described in the following part:

Quit: Leaves 'Start' without doing anything.

Run: Starts a Superbasic program. This command leaves the E.A.S.E. system. To reload E.A.S.E. from Basic use the

command 'LRUN mdv1_BOOT' at the end of your program. Another possibility to restart E.A.S.E. is to use the new defined command 'DESK', which will reload E.A.S.E. (memory permitting!). When E.A.S.E. has been started by using this command, it is no more possible to start Basicprograms from E.A.S.E. (but if you quit E.A.S.E. with the option 'Exit to Basic', your Basic program will continue work behind the 'DESK' command).

Show: If the file is an ASCII- file, like e.g. a Superbasic (R) program, this option gives you the possibility to have a look on it. The output can be frozen with 'CTRL' & 'F5' or with the button. After pressing the button, you are back in E.A.S.E..

Print: Like show, but output is directed to the printer.

Info: Opens up a window containing all supported information about the marked file. This consists of the file length, the last update date and so on. Most of the dates are not supported in the QDOS version 1.03. They are yet supported by some extensions, like the CST (R) QDISK (R) controller. Therefore don't be suprised to find the remark 'Not supported' quite frequently.

Options
=======

Show as text: Just after booting, E.A.S.E. will display directories marked with small symbols, refering to the type of the file. If you do not like this form, or if you want to use the options 'Sort by length' or 'Sort by date', you can select this command. All following output will be displayed as text.

Show as icons: The reverse function to the above one. Changing these options will not result in an immediate change of the directory format, but will only work for windows, opened after changing 'Show' mode. This gives you the possibility of displaying

both types of directories simultaneously. The chosen mode is marked in the menu by a little arrow at the lefthand side of the option.

Sort by name: Sorts the uppermost directorywindow alphabethically by the filenames.

Sort by size: Sorts the uppermost directorytextwindow by the filelengths.

Sort by date: Sorts the uppermost directorytextwindow by the date of last access (not supported in QDOS version 1.03).

Sort by marks: Sorts the marked files in the uppermost directorywindow to the top.

Exit
====

System reset: Quits E.A.S.E., sets the QL back to the switch- on status and clears the whole memory.

Exit to Basic: Quits E.A.S.E. and returns to Basic.

Start program: Similar to the 'Start' command. This command does not perform a memory- check, but clears E.A.S.E. from memory, so that the loaded program finds a clear QL. After work with the called program has been finished, E.A.S.E. will be rebooted. This command is necessary for programs which fit in the free amount of memory still available, but which, for themselves, need a much greater amount of memory to work properly. This might be the case for texteditors.

Error messages

Error messages are displayed in a window of its own. They have to be confirmed by using the windowbuttons ('Cancel' or 'Ok'). Most messages explain themselves. In the following part we will focus on a small number of messages.

'Too many windows open'

E.A.S.E. allows a maximum of seven windows to be opened simultaneously. When this message appears, you have tried to open more than seven windows. To go on, you first have to

close an old one.

'Sorry, I've run out of memory'

E.A.S.E. is a very complex program and therefore has a total length of about 64,000 bytes. When running, E.A.S.E. furthermore needs a certain amount of memory for its internal calculations.Additionally, a maximal sized window, for e.g. for a very long directory, can be sized up to 15,000 bytes. An unexpanded QL has only about 84,000 bytes free for the user. This shows that memory can become a problem. The only way to solve this problem is to buy a memory expansion, which will also drastically increase the performance of the Psion (R) programs.

'Directory window not open'

Some functions (e.g. 'Copy') are only possible, if there is an open directory window on the screen.

'No file marked'

There exists an open directory window, but no file is marked in it. This message will also appear when the copying or deletion have been finished.

All this might read very complicatedly. By using E.A.S.E. extensively, you will find, that the program is very easy to work with. E.A.S.E. is very usefull in conjunction with disk- drives, where it is possible to copy E.A.S.E. on every disk, combined with a BOOT- program. We ourselves work with this combination and are very satisfied with its performance. We hope, you will enjoy E.A.S.E. like we do.

Patching E.A.S.E.

To start the program enter:

LRUN mdv1_patch_bas

This implies that a cartridge containing the program is inserted in drive one. This program can be used to adapt E.A.S.E. to your system. E.A.S.E. is normally configurated to work with an unexpanded QL. If you posses Disk- stations, you can configurate E.A.S.E. to have access to these. It is therefore possible to copy E.A.S.E. to all your disks making your work more comfortable.

You should use this program with backup copies only.

Disclaimer

The program has been developed and reproduced carefully, neither the author nor the distributors, however, can guarantee that the program and this guide are free of errors. Therefore GIGA- SOFT under no circumstances will be liable for any direct, indirect, incidental or consequential loss of use, stored data, profit or contracts which may arise from any error, defect or failure of this software.

GIGA- SOFT has a policy of constant development and improvement of their products. We reserve the right to change manuals and software at any time and without notice.

Instructions for the Basicextension "GIGA-BASIC V 1.00"

Introduction:

Although the Sinclair QL comes with a real good Basic, some commands are missing, which would offer the full power of the QL. This extension set should increase your motivation to program in Basic. With over 70 commands and functions, Giga-Basic is a useful extension for the QL. Before starting work with Giga- Basic you should read this manual carefully. You should never work with your original copy of Giga- Basic. To obtain working copies a backup program is included. To start this program enter: 'exec_w mdv2_clone_exe'.

Note:

The Originalcartridge of GB displays the message 'PRESS ANY KEY' after booting. If the message 'THIS ISN'T THE ORIGINAL PROGRAM' appears at the screen after booting your cartridge, you probably got an unauthorised copy, which might be faulty in some functions. If this is the case, please note the way you got your copy and send it with y r cartridge to:

Gerd-Uwe Neukamp
Heresbachstraße 1

4000 Düsseldorf 1
Western Germany

We will then send you the original version, if you include a cheque of DM 10,- for p&p.

The extension includes the following groups :

- grafics
- spritehandling and spriteanimation
- base conversion
- fullscreen basiceditor
- direct access to medium
- multitasking clocks
- mousedriven screenoriented menus
- pull-down-menus
- multitasking control commands
- programmable function keys
- others

GRAFICS

PAINT #dev,x,y

Fills an irregularly bordered area of the chosen screen with the ink colour.

dev: devicenumber of the screen
x: x- coordinate
y: y- coordinate

Direct access to medium

GET #dev,variable (,variable)

Gets a value from medium and writes it into the variable. The type of the value depends on the type of the variable.

Example: GET #4,integer%

#dev: devicenumber
variable: any type and number of variables

BGET #dev,byte (,byte)

Gets byte from medium and puts it into the variable.

#dev: devicenumber
byte: any type and number of variables

PUT #dev,variable (,variable)

Writes value to microdrive. Any variabletype is allowed.

#dev: devicenumber
variable: Any type and number of variables.

BPUT #dev,byte (,byte)

Writes byte to microdrive.

#dev: devicenumber
byte: variable which gets a byte

SET_POINTER #dev,pointer

Sets filepointer to new position. With this command it is possible to have direct access to microdrive (or FLP, HDK, FDK and so on).

#dev: devicenumber
pointer: longword containing the pointer

pointer=GET_POINTER (#dev)

Gets the pointer of the selected microdrive.

#dev: devicenumber
pointer: variable containing the pointer

Base conversion

The following functions provide an easy way to convert bases.

hexnum$=CHEX$(decimal)

Converts a decimal value into a hexadecimal string

hexnum$: string which will contain the hexnumber
decimal: variable containing the decimal number

decimal=CHEX(hexnum$)

Converts a hexadecimal string (max. 32 bit) into a decimal number.

decimal: variable which will contain the decimal number
hexnum$: string containing the hexnumber

binary$=CBIN$(decimal)

Converts a decimal number into a binary string (32 bit).

decimal: variable which will contain the decimal number
binary$: string containing the binary number

decimal=CBIN(binary$)

Converts a binary string into a decimal number.

decimal: variable which will contain the decimal number

binary$: string containing the binary number

Multitasking Control Commands

The following commands are intended to control the multitasking capabilities of the QL. Now it is possible to delete, suspend or activate jobs from Basic.

JOB_INF #dev

This command displays a list of all active jobs. A Job is a program working in the background. Additionally you can see the priority, the owner job, the baseaddress and the tagnumber. Job 0 is the Basicinterpreter. For further informations on multitasking refer to the Sinclair User Guide.

#dev: devicenumber

SUS_JOB jobnr,tagnr,timeout.

Suspends a job for a period.

jobnr: jobnumber
tagnr: tagnumber
timeout: Number of frames the job being deactive (-1: infinite).

REL_JOB jobnr,tagnr

Releases a suspended job. This command is the reverse of SUS_JOB.

jobnr: jobnumber
tagnr: tagnumber

PRIOR_JOB jobnr,tagnr,priority

Sets the priority of a job. Priorities are allowed in the range from 0 to 127. 127 is the highest priority. If the priority is high, more time is available for the job.

jobnr: jobnumber
tagnr: tagnumber
priority: priority

Sprites and Animation

Giga-Basic offers a great number of efficient commands for development and animation of sprites. So it is easy to generate actiongames or programs using icons. Sprites are organized in a 32 x 20 matrix and are flickerfree.

Important definitions :

spritedatablock (sprdat): This is a memoryblock which contains the bytes for the shape (mask) of the sprite. A sprite shape contains 160 bytes. Every spritedatablock can be attached to every sprite.

spritenumber (sprnr): A sprite will be activated under a spritenumber. Under this number the sprite can be moved over the whole screen.

SPRDIM spritenr,datanr,anmtenr

Reserve memory for sprites. The defaults are :

SPRDIM 4,16,4

spritenr: number of possible sprites
datanr : number of possible spritedatablocks
anmtenr: number of the sprites which can be animated

SPRCLR

SPRCLR releases the memory allocated by SPRDIM. All defined Sprites are lost.

INVMASK #dev,x,y,sprdat

Prints a spritemask onto the screen. The coordinates are relative to the left upper edge of the selected window. The coordinates have pixel size (This is not a sprite. Only a mask will be drawn.).

dev: devicenumber
sprdat: spritedatablock

SPRON sprnr,sprdat

Activates a sprite with a spritedatablock.
Note: This command does not have any effect on the screen. The sprite will not be visible until it is activated by the MOVESPR command.

sprnr: spritenumber
sprdat: spritedatablock

SPROFF sprnr

Removes the selected sprite.

sprnr: spritenumber

REFRESH

Important after 'CLS'. All active sprites are refreshed.

INVSPRITE sprnr

The chosen sprite is inverted.

sprnr: spritenumber

MOVESPR sprnr,x,y (,sprdat)

Sets a sprite to a new position. The optional spriteparameter is intended to change the appearance of the sprite. If no spritedatablock paramater is given, the sprite image does not change.

sprnr: spritenumber
x,y: absolute pixel coordinates
sprdat: spritedatablock

Several sprites can be moved using only one command. This type of motion is named animation. It is a really easy task to move rockets, men, cars and other things now.

SETANIMATE sprnr,sprdat(,sprdat1)(,sprdat2)

This command has as many parameters as you want to. The given spritedatablocks are connected in series.

Note: Before using ANIMATE, you have to initialize the routine with the SETANIMATE command. A maximum of 16 spritedatablocks may be connected.

sprnr: spritenumber
sprdat: spritedatablock

CLRANIMATE sprnr

The selected spriteentry will not be animated after the use of CLRANIMATE.

sprnr: spritenumber

STEPSPRITE sprnr,xstep,ystep,statx,staty

This command can be used after every SETANIMATE. You can change the direction and speed of the animation in your basic program.

sprnr: spritenumber
xstep: stepsize x
ystep: stepsize y

statx: 0 After reaching the border of the screen invert the x-direction.
1 After reaching the border appear at the otherside.
2 After reaching the border kill the sprite.

staty: Same as statx but refering to the y-direction.

ANIMATE

Moves all sprites which are declared with the SETANIMATE command over the screen.

sprite=COLLISION(sprnr)

Asks whether two sprites are overlaid. If it is true COLLISION returns the spritenumber; otherwise -1.

sprnr: spritenumber
sprite: If the sprite isn't in contact with another sprite -1 will be returned, otherwise the spritenumber.

Spritedefinition commands :

Sprites can be defined for MODE 4 or MODE 8. Following an example of an eightcolour sprite:

```
100 :
110 SPRDEFBLOCK starship
120 :
130 SD8 "................"
140 SD8 "................"
150 SD8 ".......11......."
160 SD8 "......1111......"
170 SD8 "....22222222...."
180 SD8 "..333333333333.."
190 SD8 "...3333333333..."
200 SD8 ".....7....7....."
210 SD8 "....7......7...."
220 SD8 "...7........7..."
```

The fourcolour example .

```
100 :
110 : SPRDEFBLOCK disk
120 :
130 SD4 "................................"
140 SD4 ".##############################."
150 SD4 ".##############################..."
160 SD4 ".##############################."
170 SD4 ".##############################."
180 SD4 ".###############....###############."
190 SD4 ".############........############."
200 SD4 ".##########..........##########."
```

```
210 SD4 ".###########........###########."
220 SD4 ".############....############."
230 SD4 ".################################."
240 SD4 ".###############..###############."
250 SD4 ".###############..###############."
260 SD4 ".################################."
270 SD4 "................................"
```

The colours are set in the following form:

MODE 4

```
red       : '1'
green     : '2'
white     : '3','#'
black     : all other characters
```

MODE 8

```
blue      : '1'
red       : '2'
magenta   : '3'
green     : '4'
cyan      : '5'
yellow    : '6'
white     : '7','#'
black     : all other characters
```

SPRDEFBLOCK sprdat

Clears the selected spritedatablock and prepares it for a new definition.

sprdat: spritedatablock

D4 defblock$

Command to define a fourcoloursprite. Up to 20 commands can be used after a SPRDEFBLOCK command. The string must be 32 characters long.

SD8 defblock$

Command to define an eightcoloursprite. Up to 20 commands can be used after a SPRDEFBLOCK command. The string must have a length of 16 charactars.

SPRLOAD name$

With this command you can load previously defined spritedatablocks. Before you use this command enough space must be reserved by SPRDIM.
Example : SPRLOAD "MDV1_PACMAN_SPR"

SPRSAVE name$

If you want to save the allocated sprite area you can use this command. Only the area for spritedatablocks will be saved.

flag=SPRACTIVE(sprnr)

With SPRACTIVE you can ask whether a sprite is active. 1 is true and 0 is false.
sprnr: spritenumber

x=SPRXPOS(sprnr)
y=SPRYPOS(sprnr)

With these functions you can find out the location of a sprite.
sprnr: spritenumber

MENU CONTROL COMMANDS

The following commands support userfriendly screenorientated menus. Now you can program mousedriven menus as with the APPLE MACINTOSH (R).
The handling is very simple. With commands like MENUPR or MENUBLOCK you define a BLOCK. This block can be manually inverted or selected with the MOUSE function. Possible inputmedia are the cursorkeys or a mouse with the ABC-interface (included in the big ABC package).

Example :

```
100 :
110 REMark small example menu
120 :
130 CLS
140 PRINT "M E N U"
150 PRINT:PRINT
160 MENUPR 1," Start a program"
170 MENUPR 2," List a program"
180 MENUPR 3," End"
190 :
200 a=MOUSE
210 :
220 SELect on a
230   =1:start
240   =2:LIST
250   =3:STOP
260 END SELect
```

After entering and starting the program, the menupoints appear as if they were printed with the PRINT command. An arrow appears, too. This arrow can be moved over the whole screen. If the arrow is in range of a menupoint this will be inverted. So you can see exactly what you have chosen. By pressing the button or space the selected menunumber will be returned.

SETMDEV mode

Selects inputmedium for the menucommands.

mode: 0: keyboard (cursorkeys/ space)
1: mouse

MENUDIM number

Reserves memory for the menupoints. Space for pull-down-menus will be automatically allocated.

number : the maximum number of menupoints

MENUBLOCK #dev,blknr,x,y,x0,y0

This command marks a block with the chosen menublocknumber.

#dev: devicenumber of a screen
blknr: blocknumber
x,y: size of the block
x0,y0: position relative to the selected window

MENUPR #dev,blknr,text$

Prints a text on the screen similar to the print command and activates it as a menublock.

#dev: devicenumber of a screen
blknr: blocknumber
text$: text

The separator ';' is allowed.

ICON #dev,blknr,sprdat,x,y

Similar to the INVMASK command it displays a spriteblock on the screen. The difference is that ICON marks it as menublock. With this command it is possible to access symbols in a similar way as the MENUPR command. You can define ICONs and use them for defining MACINTOSH (R) style programs.

dev: devicenumber
blknr: menublocknumber
sprdat: spritedatablock
x,y: pixel coordinates relative to the window

INVBLOCK blknr

Inverts a block.

blknr: menublocknumber

CLRBLOCK blknr

Clears a block.

blknr: menublocknumber

nr=MOUSE(x,y)

Displays an arrow which can be moved over the screen by using the mouse. With the arrow you can select an item.

nr: if no menupoint was chosen -1 will be returned,

otherwise the menublocknumber will be returned.

x,y: startcoordinates of the arrow

x=MXPOS,y=MYPOS

These functions return the position of the arrow after pressing the SPACE-key.

Pull-Down-Menus

This a new type of menutechnique. On top of the screen you can see a headline holding the menupoints. If you move the arrow to one of the points, a window will be opened with a submenu. Now you can choose the point you want in the submenu. With the Pull- Down- Menus you can handle a great number of menupoints on a very small room.

Example :

```
100 SPRDIM :REMark Reserves space for the arrow
110 MENUDIM :REMark Allocates space for the pull-down-menu
120 :
130 MENU 0,0,1,"Addresses"
140 MENU 1,0,1,"Clear"
150 MENU 2,0,1,"Input"
160 MENU 3,0,1,"Edit"
170 :
180 MENU 0,1,1,"File"
190 MENU 1,1,1,"Load"
200 MENU 2,1,1,"Save"
210 :
220 MENU 0,2,1,"Exit"
230 MENU 1,2,1,"Reset"
240 MENU 2,2,1,"Basic"
250 :
260  SETMENU :REMark Clears the screen and shows the
menuheadline
270 :
280 GETMENU :REMark Shows the arrow and gets the menupoint
290 x=HMENU
300 y=VMENU
310 :
```

MENU vnr,hnr,active,string$

Command to define a pull-down-menue.

vnr: Vertical coordinate. The headline has the coordinate zero.

Note: The menupoints within the headline (vnr=0) must be defined in ascending order. Every headlinepoint must have a submenu. A maximum of 10 items can be

defined in the vertical direction.

hnr: Horizontal coordinate. The number of horizontal items is restricted to a maximum of 8. The total length of the items in the headline must be selected to fit according to the selected screenmode. This is important for compatibility between mode 256 and mode 512.

active: Flag which selects whether you can access the menupoint or not.

string$: Text of the menupoint. The length is restricted to 14 characters.

SETMENU paper1,paper2,actcol1,pascol

Clears the whole screen. Displays the headline.

paper1: Screencolour
paper2: Bordercolour of the headline
actcol: Colour of the active menupoints
pascol: Colour of the passive menupoints

GETMENU x,y

Displays the arrow and allows the user to select menupoints.

x,y: startposition of the arrow

Default: GETMENU 256,100

ACTIVE vnr,hnr,active

Activates and deactivates menupoints.

vnr: vertical Position of menupoint
hnr: horizontal Position of menupoint
active: flag, 1-active, 0-inactive

x=HMENU
y=VMENU

With these functions you can get the position of the chosen menupoint.

possible range :
HMENU (0-7)
VMENU (1-9)

Programmable function keys

Directly after starting the Basic extension, the functionkeys

are programmed. Information about the assignments can be gained by pressing "F1". This assignment can easily be changed by the user. Furthermore the functionkeys can be switched off if they would disturb the function of other programs.

KEYS #dev

Lists all functionkey assignments to the specified device.

dev: devicenumber (default is 1)

KEY keynr,string$

Allows the user to change the functionkey assignment.

keynr: Number of the functionkey (1 to 10, numbers greater than 5 are aktivated by pressing the shiftkey simultaniously.

string$: String containing the command (max. 32 characters). Example: KEY 1,'LIST' & chr$(10)

KEYSON

Turns funktionkeys on.

KEYSOFF

Turns functionkeys off.

CLOCKCOMMANDS

It is possible to display either a digital or an anologue-clock on the screen. There is also the possibility of changing colour and size to adapt the clocks to own programs.

DCLOCK on,x,y,paper,ink1,ink2

Displays a digital clock.
Default : DCLOCK 1,340,0,2,7,4

on: flag, 0-removes the clock, others-activates the clock
x,y: right upper coordinate of the clock in pixel coordinates
ink1: inkcolour
ink2: bordercolour

ACLOCK on,x,y,size,paper,ink1,ink2,ink3,ink4

Displays an analogue- clock.
Default : ACLOCK 1,0,0,40,0,2,2,4,6

on: flag, 0-removes the clock, others-activates the clock
x,y: right upper coordinate of the clock in pixel coordinates
size: vertical ize of the clock
paper: papercolour.
ink 1-4: colour for the hands of the clock an the circle around it

OTHER COMMANDS

CAT #mdvnr

Displays the directory of the specified drive in a formatted form. Furthermore it displays the number of blocks (512 bytes) each program uses.

#mdvnr: number of drive (default : 1)

DUMP #dev

Displays all variables with contents, procedures and functions with linenumbers.

#dev: outputdevice (default is 1)

COMMANDS #dev

Lists all new Basiccommands with their startaddress on the outputdevice.

#dev: outputdevice (default is 1)

HRDCOPY inv

Prints hardcopy on EPSON-compatible printers. Through technical restrictions, it is only possible to print a maximum of 480 horizontal points.

inv: flag, 1-inverted print, 0-normal print

SYSTEM #dev

Displays the systemvariables on screen.

#dev: outputdevice (default is 1)

a=FREE

Returns the amount of free Basicmemory.

SCREEN #dev,linenr,tab

default : SCREEN #1,1,3

This command enters the screeneditor. It allows the user to edit Basicprograms in a way similar to QUILL. Unlike a normal ASCII-Editor all entered lines are syntactically checked by the interpreter.

Note : The interpreter will not accept lines after a programbreak if the functions and procedures are not reinitialized. This is possible by using the CLEAR command, which will produce the message 'PROC/DEF CLEARED'. After this message the work with the screeneditor can go on.

dev: windownumber to edit in
linenr: linenumber which will be displayed first
tab: stepsize of the inbuilt tabulator

The editor accepts the following keysequences:

cursup	
cursdown	
cursright	
cursleft	
ESC	leaves the editor
TABULATE	tabulator
SHIFT&ALT&UP	jumps to start of program
SHIFT&ALT&DOWN	jumps to end of program
ALT&UP	page up
ALT&DOWN	page down
CTRL&RIGHT	deletes character under cursor
CTRL&LEFT	deletes character at the left of the cursor
CTRL&ALT&LEFT	clears Basicline
CTRL&ALT&RIGHT	deletes all characters at the right of the cursor
SHIFT&UP	jumps to the first line of the screen
SHIFT&DOWN	jumps to the last line of the screen
ALT&LEFT	jumps to start of line
ALT&RIGHT	jumps to end of line

SETFONT #dev,fount1,fount2

Gives the user the possibility of using a selfdefined Characterset. It is possible to define up to two charactersets at one time, in which case a character is displayed from the first characterset, if defined there, if not defined, it is taken from the second and, if it also is not defined there, the first defined character of the

second set is displayed. To select the inbuilt fonts of the Ql, just enter zero for the startaddress of the font.

fount1 : startaddress of the first font
fount2 : startaddress of the second font

Example: (Using the supported characterset 'BIG_CST'.)

```
100 a=RESPR (1024)              :REMark Reserve space for font.
110 LBYTES 'mdv1_BIG_CST',a     :REMark Load new font.
120 FOR channel = 0 TO 2        :REMark Loop
130   SETFONT #channel,a,0        :REMark Activate new font for
140   CLS #channel                :REMark every window.
150 END FOR channel             :REMARK End loop
```

MONSCR mode

Activates the switch-on-status of the windows for the monitormode.

mode: Selects 4 or 8 colourmode.

TVSCR mode

Activates the switch-on-status of the windows for the televisionmode.

mode: Selects 4 or 8 colourmode.

SETMON #dev, xsize, ysize, x0, y0, paper, strip, ink, borderwidth, bordercolour

Changes a defaultwindow in the monitormode.

SETTV #dev, xsize, ysize, x0, y0, paper, strip, ink, borderwidth, bordercolour

Changes a defaultwindow in the televisionmode.

mode=GETMODE

Returns the screenmode.
4= fourcolour, 8= eightcolour.

Windowcommands

The windowcommands allow the user to work with 'real' windows. With these commands it is possible to save the background of a window before writing to it and to restore this background after closing the window. This technique is known as 'refreshing'.

~~SCRSAVE~~ nr,xs,ys,x,y
SCRSTORE

Saves an area of the screen.

nr: Number from 0 to 15. Tis number represents the label for the saved screen. It has to be specified in the other commands refering to the saved screen area.

xs,ys: size of the window
x,y: left upper position of the window

SCRLOAD nr

Redisplays an saved area of the screen.

nr: Labelnumber (0 to 15)

SCRCLEAR nr

Clears the part of the memory containing the saved screen.

nr: Labelnumber (0 to 15)