MTX EXTENDED BASIC

This package was designed to aid the BASIC programmer by filling in the gaps in MTX basic which result in you diving into the assembler section of the operators guide. All the new commands are checked for syntax on entry, and will not be accepted until correct. All the usual MTX syntax rules apply, spaces must follow keywords e.t.c. In addition, the appalling error messages have been replaced by more meaningful ones; never again will you see SE.B! The other major criticism of the MTX is it's poor cassette facilities, so this package has additional commands to make saving and loading of screens and blocks of memory available from basic. The new commands are listed below. Note each one starts with a period (.) and ends with an underscore (_).

.SSNEW_

This command is used to clear out any program resident in memory without corrupting the code for the new commands.If by mistake you type NEW instead, immediately enter .SNEW_ and hopefully the code will be unaffected. There is no way of recovering the code after a hard reset and therefore it must be reloaded from tape.

.BSAVE_ "filename",start of block,length of block

This is a binary save which saves a block of code to tape under the name "filename".The filename can be up to fourteen characters long, any additional characters being ignored.
e.g .BSAVE_ "ROMcode",0,8192

This will save the first 8k of ROM.

.BLOAD_ "filename",start of block

This is the load operation for the above code. Note it is not necessary to specify the length since this will be calculated for you.

e.g. .BLOAD_ "ROMcode",0

This will load back the block saved in the above binary save operation.

.BVERI_ "filename",start of block

This is identical to .BLOAD_ except the code is not loaded into memory but is checked with what is already there.

.SSAVE_ "Filename"

This is a screensave, and will save the contents of the graphics screen to tape under the specified filename.

.SLOAD_ "Filename"

This is the reverse of the above operation, and will load the graphics screen saved using the above command. The screen builds up directly onto the screen as it is loaded, bringing the MTX into line with other micros on the market which have this facility.

.PAINT_ x,y,colour

This will fill an area on the screen with the desired colour starting at the x,y coordinates specified. Due to certain restrictions placed by us in order to obtain speed this will work on a 32*24 graphics screen, and if the current screen is not of this type then an "Incorrect virtual screen type" error will be given. If the point specified by x,y is already set to ink the machine will give an "Point already PAINTed" error. It is also possible to obtain an "Out of free space" error when filling extremely complicated shapes but this happens very rarely and the space is reclaimed for use again once the error has been generated.

.DPOKE_ address,value

This is a two byte poke i.e. value is put into address and address+1 in the following format:

```
POKE address,value-256*INT(value/256)
POKE address+1,INT(value/256)
```

This simplifies calculating LSB/MSB values when attempting to put 16-bit values into memory.

.SLINE_ x1,,y1,x2,y2

This is the same as the MTX LINE command, however it draws more accurate lines. The syntax is the same as MTX LINE.

Just try the following program which draws two parallel lines and compare them.

```
10 VS 4:CLS:.SLINE_ 0,0,198,191:
LINE 4,0,202,191:DSI
```

Note that PLOT SPRITES will not follow the points plotted by .SLINE_ however the line follows the normal rules governed by the ATTR command.

Additionally, this package caters for more advanced programmers with the following extensions to PANEL.

Fill. Pressing "F" now gives the prompt "Fill" which requires the start address of the block you wish to fill. You will then be given the prompt "To" which requires the end address then "With" which requires the value to fill the block with. As soon as this is entered the block will be filled with the required value.

VRAM read. This is used to copy VRAM into RAM so that it can be manipulated via PANEL. Pressing "V" gives the prompt "Read" which requires the VRAM address of the block to copy. This is followed by "To" for the end address then "Put" for the RAM address to put the data.

Write
RAM to VRAM. This is used to write blocks of RAM into VRAM. On pressing "W" the prompt "Write" is given which requires the start address of the block to write, followed by "To" which requires the end address then "VRAM" which asks for the start address in VRAM.

Care must be taken when using these commands so as not to overwrite any valuable system information, or corrupt the VRAM pattern table thus corrupting the character set and making the results totally unreadable.

Important points to note

When a program written using Extended Basic is saved to tape the Extended Basic code is saved alongside and placed high up in memory when loaded back. However this code will not be active so in order to initialise it, it is advisable to make the first line of your program thus:

```
10 POKE 64853,0: POKE 64854,238: POKE 64852,195
```

When the program is subsequently run the Extended Basic commands are initialised and will execute correctly.

The Extended Basic commands occupy space normally reserved for a sound buffer, and unfortunately the use of continuous sound will most likely corrupt the code. This is, however, a small price to pay for the extra features, but it is a point worth remembering.