For the BBC Micro

BEEBUG SOFT

# TOOLKIT PLUS

## Advanced Basic Programming Aid

BEEBUG SOFT

# TOOLKIT PLUS

## BASIC PROGRAMMING **AID FOR THE BBC MICRO**

**by Alan Malik**

Dolphin Place, Holywell Hill, St. Albans, Herts., AL1 1EX.



Manual written by Geoff Bains.

# CONTENTS

# INTRODUCTION

Toolkit Plus is a sophisticated piece of software designed to assist Basic programming on the BBC micro. It is supplied on Rom, and provides the user with more than 50 new command features which not only speed up the process of programming, but assist in the task of debugging, and generally streamline the activity of programming in Basic. Toolkit Plus also includes a full screen program editor to help you enter and modify your programs more easily.

For simplicity of use, all commands may be entered with a unique command name directly from the keyboard, or from within your own Basic program. A special option also allows nearly all the commands to be selected from a main menu and several of the routines may be entered directly from the program editor, making Toolkit Plus extremely easy to use.

Extensive error checking routines are also incorporated, and the user is given a range of error messages if commands are incorrectly entered.

## Conventions used in this manual

In this manual specific keystrokes required by Toolkit Plus (such as the 'RETURN', 'SHIFT', and cursor keys) are indicated : **RETURN**, **SHIFT** and ↑ ↓ ← →.

All parameters are shown in this manual enclosed in angled brackets. Single sets of brackets <> are used to indicate essential parameters, while double brackets <<>> indicate optional parameters.

## Fitting Toolkit Plus

Toolkit Plus is supplied on a sideways Rom and can be fitted either in the main circuit board of your BBC micro or in any kind of sideways Rom board. Please refer to the separate fitting sheet for instructions on how to fit Toolkit Plus.

Once Toolkit Plus is fitted in your machine you may use your computer as normal, but the added commands of Toolkit Plus are now available whenever needed. You will probably be using Toolkit Plus to work on a program in Basic. This may now be loaded and run in the normal way from cassette or disc.

## Using Toolkit Plus Commands

All the facilities of Toolkit Plus are accessed using 'star' commands. Each facility has a command word associated with it.

For example if you type the following:

```
*TOOLS RETURN
```

you will see a list of the commands available in Toolkit Plus.

The command word may be entered in either upper or lower case letters or even a mixture of both. All of the following commands are legal with Toolkit Plus.

```
*TOOLS RETURN
*tools RETURN
*Tools RETURN
```

The Toolkit Plus commands may also be entered in an abbreviated form. The shortest abbreviation that you can use will depend on the types and priorities of other Roms fitted in your computer. Each command has a minimum abbreviation that will allow Toolkit Plus to recognise it. In this manual the minimum abbreviation given is the command that will call the Toolkit Plus routine when Toolkit Plus is the highest priority Rom, apart from Basic and the DFS. For the command, *TOOLS this minimum abbreviation is *TO. so this routine could also be entered by typing:

```
*TO. RETURN
```

or even:

```
*to. RETURN
```

Although each Toolkit Plus command has a its own unique command word, you may find that some of these clash with the command words of other Roms which you may have fitted to your machine. Toolkit Plus has a special feature to avoid command name clashes. If any command name clashes, when the Toolkit Plus command is required, simply preface the command name with a B - for Beebugsoft - eg type:

```
*BTOOLS RETURN
```

instead of just:

```
*TOOLS RETURN
```

This will ensure that the command is intercepted by Toolkit Plus rather than any other Rom. There are no clashes of command words with any other Beebugsoft Roms.

If you require the command from the other, clashing, Rom to be executed and that Rom does not have a similar letter prefix facility you can signify to Toolkit Plus that the command is *not* to be obeyed but passed on to the other Rom by prefixing the command with a B or a b and typing the second letter (the first of the command word) in a different case from the prefix.

To clarify these conventions by way of example, the following commands are all legal with Toolkit Plus and have differing effects.

| | |
|---|---|
| `*TOOLS` | will execute the Toolkit Plus routine if that Rom is in a higher priority to any clashing Rom. |
| `*tools` | will execute the Toolkit Plus routine if that Rom is in a higher priority to any clashing Rom. |

| | |
|---|---|
| `*BTOOLS` | will execute the Toolkit Plus command regardless of priority. |
| `*btools` | will execute the Toolkit Plus command regardless of priority. |
| `*bTOOLS` | will execute the routine of a clashing Rom if it is in a lower priority to Toolkit Plus. |
| `*Btools` | will execute the routine of a clashing Rom if it is in a lower priority to Toolkit Plus. |

A further method of calling a Toolkit Plus command is to type:

`*MENU` **RETURN**

This displays a menu on the screen from which almost all of the Toolkit Plus routines may be selected with a single keystroke.

## Parameters

Many of the Toolkit Plus commands require parameters to control their action. These parameters can be either numeric, string, or literal. (Literal parameters are flags that determine the effect of the command and, when required, should be entered as given). If parameters are being entered together with the command, these should follow the command with one or more spaces between the command and the first parameter, and commas before subsequent parameters. Literal parameters should be separated from one another by spaces.

For example, the Toolkit Plus command to renumber a Basic program is ***RENUMBER.** This may be followed by up to four numeric parameters to specify the program lines which are to be renumbered and the manner in which the renumbering is to take place. A typical renumber command issued to Toolkit Plus might be:

`*RENUMBER 1000,10,370,800` **RETURN**

### Optional Parameters

Some parameters, such as the literals used with the ***CRUNCH** command, are optional. Optional parameters are indicated in this manual with a double angle bracket thus:

***CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>>**
**<<T>><<V>><<:>>**

So, to call this command, specifying only the B and R constraints, you would simply type:

`*CRUNCH B R` **RETURN**

## Escape

The **ESCAPE** key may be used at any time to exit any Toolkit Plus routine with the exception of ***CRUNCH**, the menu display and the full screen editor.

On pressing ESCAPE during the execution of the other routines you will see the prompt appear, signifying that you have been returned to Basic.

**ESCAPE** has no effect during the compaction of a program using ***CRUNCH**. When in the full screen editor, **ESCAPE** will take you to the menu display to select further Toolkit Plus routines and a further press of **ESCAPE** will return you to the editor. **CTRL** **ESCAPE** will return you to Basic from either the menu or the full screen editor.

## Screen Modes

All the facilities of Toolkit Plus, with the exception of the full screen editor, will operate correctly in all the display modes available on the BBC micro. The full screen editor is not operative in modes *2* and 5.

When a Toolkit Plus routine is called no changes will be made to the currently selected screen mode or to the currently selected text colours.

### Toolkit Plus Workspace

Toolkit Plus does not raise the value of PAGE. However, it does make extensive use of the following memory areas.

```
PAGE 0    (&0000 - &00FF)
PAGE 5    (&0500 - &05FF)
PAGE 6    (&0600 - &06FF)
PAGE 7    (&0700 - &07FF)
PAGE 9    (&0900 - &09FF)
```

# SPECIAL COMMANDS

There are over fifty new command features provided by Toolkit Plus. The function, syntax, and methods of use of most of these are described in the next section of this manual. However, three of the commands, ***HELP, *TOOLS,** and ***MENU** have a different use from the other commands and so they are considered here, separately.

## *HELP

Like most Roms available for the BBC micro, Toolkit Plus makes use of the Operating System's ***HELP** command. Issued on its own this command will display a list of the Roms present in your machine.

```
>*HELP

DFS 0.9V
  DFS
  UTILS

TOOLKIT PLUS 2.00
  TOOLKIT

EXMON II 1 .03

SPELLCHECK II v1.20

WORDWISE PLUS 1.4A

OS 1.20
>
```

Some Roms are able to provide greater help on aspects of their operation by making use of the extended HELP command. Toolkit Plus is one of these. By following the command ***HELP** with the name **TOOLKIT,** a complete list of all the commands available in Toolkit Plus along with their syntax, as detailed in this manual, is displayed on the screen.

```
>*HELP TOOLKIT
AUTOSAVE <<filename>>
CHECK <filename>
CLEAR
```

```
CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>> <<T>> <<V>> <<:>>
DUPLICATE <start>,<end>,<dest>
EDIT
ENVELOPE <<envno>>
ERASE <<start>>,<<end>>
ERROR <<toggle>>
FORMAT <trks>,<drv> <<drv>> <<drv>> <<drv>>
FKEY <<keyno>>
LMOVE <start>,<end>,<dest>
LOOK <filename>
MENU
MEMORY <<addr>>
MERGE <filename>
PARTSAVE <filename>,<start>,<<end>>
RECOVER <<+>> <<+>> <<+>> <<+>>...
RELOCATE <addr>
RENUMBER <<newstart>>,<<inc>>,<<oldstart>>,<<oldend>>
REPORT
ROMS
SCREEN <filename>
SEARCH <string1>,<<string2>>
SET
SLOMO <<delay>>
STATUS
SYNTAX <<P>>
TOOLS
TRACE <<start>>,<<end>>,<<S>>
VERIFY <<drv>> <<drv>> <<drv>> <<drv>>
XREF <<R>> <<I>> <<S>> <<A>> <<H>> <<F>> <<P>> <<C>>
<<L>>
>
```

The list of commands is long and it will scroll off the screen. This may be prevented by typing **CTRL** **N** before issuing the ***HELP TOOLKIT** command and pressing **SHIFT** to scroll the display, or by pressing CTRL **SHIFT** as the list is displayed.

## *TOOLS

The Toolkit Plus command, ***TOOLS** has the same effect as the ***HELP TOOLKIT** command and it will display the Toolkit Plus commands and their syntax.

## *MENU

All the Toolkit Plus commands may be entered as 'star' commands directly from the keyboard or from inside a Basic program. Another method of calling the majority of the Toolkit Plus commands is to use the in built menu system.

The Toolkit Plus menu is called with the command, ***MENU.** This will display a menu on the screen from which almost all the Toolkit Plus commands may be called with a single keystroke. One command, ***SCREEN,** is not available from the menu as this would defeat its purpose (see the full description of ***SCREEN** in the next section of this manual).

The commands are arranged in the menu in groups to aid quick selection.

```
                    TOOLKIT PLUS

Diagnostic          Editing             Program

1 SEARCH            7 EDIT              D CLEAR
2 SLOMO             8 DUPLICATE         E CRUNCH
3 TRACE             9 ERASE             F RELOCATE
A REPORT            A LMOVE             G RECOVER
5 XREF              B RENUMBER          H MERGE
6 ERROR             C PSAVE             I SYNTAX

Info                Filing
                                        Escape
J ENVELOPE          P AUTOSAVE          to edit
K STATUS            Q CHECK
L FKEY              R LOOK              CTRL ESCAPE
M MEMORY            S SET               to exit
N ROMS              T FORMAT
0 TOOLS             U VERIFY            BEEBUGSOFT
Enter option:-
```

Any parameters required by Toolkit Plus commands selected from the menu are prompted for. So, for example, selecting option U from the menu will prompt for the disc drive number to be verified before entering the verify routine.

In addition to the Toolkit Plus commands, the menu also allows access to other 'star' commands. These can be entered in the normal way after * has been pressed.

Pressing **ESCAPE** from the menu will select the full screen Basic editor from which programs can be easily entered or altered and pressing **ESCAPE** from within the editor will take you back to the menu. Pressing **CTRL** **ESCAPE** from the menu will return you to Basic in immediate mode.

# THE TOOLKIT PLUS COMMANDS

In this section of the manual, each of the commands provided by Toolkit Plus will be described in alphabetical order. However, there are several commands which deserve special attention. ***HELP, *TOOLS,** and ***MENU** have been described in the previous section. ***EDIT** also requires special treatment and you will find a complete description of the full screen editor in the next section of this manual.

## AUTOSAVE

**Syntax:** ***AUTOSAVE <<filename>>**
**Minimum abbreviation:** ***AU.**
**Menu option:** **P**
**Function:** **Enable the automatic backup facility.**

Issuing this command without a following filename will disable the automatic backup facility. Specifying a filename will cause the program currently in memory to be saved to disc whenever **CTRL** **@** is pressed. The program is saved under a filename which comprises the specified filename with the addition of a single character that is incremented through the numbers 0 to 9, and then the alphabet, on each successive press of **CTRL** **@.**

The filename specified should be less than seven characters. If a filename is specified that is longer than permitted, it will be truncated to six characters.

**For example:**

```
>*AUTOSAVE prog
>
```

## CHECK

**Syntax:** ***CHECK <filename>**
**Minimum abbreviation:** ***CH. Menu option: Q**
**Function:** **Report differences between a program in memory and one on cassette or disc.**

This command performs a comparison of the Basic program in memory with a program on cassette or disc with the specified filename. If the two programs are found to be identical then the message Program OK is issued. At the first difference found between the programs, the routine terminates and the error is reported.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
>*LOOK prog
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus for the BBC micro"
   40 NEXT I%
>*CHECK prog
 Error at line 30
>
```

## CLEAR

**Syntax:** ***CLEAR**
**Minimum abbreviation:** ***CL.**
**Menu option:** **D**
**Function:** **Reset the resident integer variables (A% - Z%).**

This command supplements the Basic command, **CLEAR,** which performs a similar function on all other variables.

**For example:**

```
10 CLS
20 CLEAR
30 *CLEAR
```

## CRUNCH

**Syntax:** ***CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>> <<T>> <<V>> <<:>>**
**Minimum abbreviation:** ***CR.**
**Menu option:** **E**
**Function:** **Compact a Basic program.**

This command invokes a powerful set of routines which remove from a Basic program all unnecessary spaces and **REM** statements, concatenate program lines, and substitute short variable names.

All the various program-shortening functions will be performed if ***CRUNCH** is issued alone. Each of the functions may be *prevented* by following the command with one or more of the following parameters:

| | |
|---|---|
| B | Brackets |
| C | Program line concatenation |
| L | **LET** statement removal |
| R | **REM** statement removal |
| S | Space removal |
| T | **THEN** statement removal |
| V | Variable name contractions |
| : | Colons and null line removal |

If substitution of shortened variable names is performed a list of the new, contracted variable names against the original variable names is displayed.

**For example:**

```
>LIST
   10 REM example
   20 FOR count=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT count
>*CRUNCH

Variable crunching

A — count

Old Program Size = 72
Bytes Saved = 40
New Program Size = 32

>LIST
   20FORA=0T010:PRINT"Toolkit Plus"NEXTA
>
```

## DUPLICATE

**Syntax:** ***DUPLICATE <startline>,<endline>,<destinationline>**
**Minimum abbreviation:** ***DUP.**
**Menu option:** **8**
**Function:** **Copy program lines within a Basic program.**

This command may also be called by pressing **CTRL** **D** from within the full screen editor.

This command will copy one or more program lines between, and including, the specified start line and the specified end line, from their original position in the program to a new position after the specified destination line.

When program lines have been copied in this way the program line numbers will be out of sequence. Toolkit Plus automatically renumbers all the program lines after the destination line number in steps of ten to avoid problems when subsequently running or editing the program.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
   50 REM destination
>*DUPLICATE 20,40,50
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
   50 REM destination
   60 FOR I%=0 TO 10
   70 PRINT "Toolkit Plus"
   80 NEXT I%
>
```

## EDIT

**Syntax:** ***EDIT <<line>>**
**Minimum abbreviation:** ***ED.**
**Menu option:** **Escape or 7**
**Function:** **Enter the full screen Basic program editor.**

Full details of the editor are provided later in this manual.

## ENVELOPE

**Syntax:** ***ENVELOPE <<envelopenumber>>**
**Minimum abbreviation:** ***ENV.**
**Menu option:** **J**
**Function:** **List envelope definitions.**

This command will list the fourteen envelope parameters in the format of the Basic **ENVELOPE** command. This allows the easy editing of the parameters using the **COPY** key.

The command, ***ENVELOPE** on its own will list the envelope definitions for all the envelopes, 1 - 16. Specifying an envelope number will list that envelope definition only.

**For example:**

```
>*ENVELOPE   3
ENVELOPE   3,1,5,5,10,255,255,1,2,2,0,1,126,126
>
```

## ERASE

**Syntax:** ***ERASE <<startline>>,<<endline>>**
**Minimum abbreviation:** ***ERA.**
**Menu option:** **9**
**Function:** **Delete Basic program lines.**

This command may also be called by pressing CTRL E from within the full screen editor.

This command is much faster than the Basic DELETE command. The command alone will delete the whole program. A section of the program may be erased by specifying the start and end line number of that section.

Specifying a single line number will delete just that line. Specifying a single line number followed by a comma will delete all lines starting at that line and a single line number preceded by a comma all line up to and including that line.

If you delete your entire program using the ***ERASE** command, you may recover your program with the Basic command, **OLD.** Sections of the program deleted with this command cannot be recovered by any means.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
```

```
   30 PRINT "Toolkit Plus"
   40 NEXT I%
>*ERASE 20,
>LIST
   10 REM example
>
```

## ERROR

**Syntax:** ***ERROR <<flag>>**
**Minimum abbreviation:** ***ERR.**
**Menu option:** **6**
**Function:** **Enable the advanced error handling facility.**

The command ***ERROR,** alone, will enable the advanced error handling facility. Issuing the command with any following literal parameter, such as ***ERROR D** will disable this facility.

When the advanced error handling facility is enabled, the occurrence of any run time error in a Basic program will cause the editor to be entered at the line containing the statement causing the error, and with the editing cursor close to the statement in error.

**For example:**

```
10*ERROR
```

## FKEY

**Syntax:** ***FKEY <<key number>>**
**Minimum abbreviation:** ***FK.**
**Menu option:** **L**
**Function:** **List the function key definitions.**

This command will print the definitions assigned to the user programmable function keys in the same format as the ***KEY** command requires to program them. This enables the easy editing of the definitions using the **COPY** key.

All the function keys, 0 to 15, are listed whether or not the keys 11 to 15 (the editing keys) are enabled as function keys. If a key number is specified when the command is called, the definition of that function key alone is displayed.

**For example:**

```
>*KEY 10 "OLD¦MMODE3¦MLIST¦M"
>*FKEY 10
 *KEY 10 OLD¦MMODE3¦MLIST¦M
>
```

## FORMAT

**Syntax:** ***FORMAT <tracks>,<drive> <<drive>> <<drive>> <<drive>>**
**Minimum abbreviation:** ***F.**
**Menu option:** **T**
**Function:** **Format a disc.**

This command will format one or more sides of a disc in the specified drive(s) to the specified number of tracks. The drive numbers should all be within the range 0 3 and tracks either 40 or 80. The disc surfaces are formatted in the order specified. This command must be preceded by the command, ***ENABLE,** otherwise you will be asked to confirm the command.

The format routine will verify the formatted disc automatically as it executes. If a track does not verify successfully first time, a question mark is printed against the track number and further attempts are made. Five attempts are made to verify a faulty track and an error message is printed if all attempts are unsuccessful.

**For example:**

```
>*ENABLE
>*FORMAT 80,02
Formatting track 18 ????? Error
Formatting track 19 ??
Formatting track 79
>
```

## HELP

**Syntax:** ***HELP TOOLKIT**
**Minimum abbreviation:** ***H. T.**
**Function:** **Display a list of the Toolkit Plus commands and their syntax.**

This command is identical in operation to the command, ***TOOLS.** Please see the section in this manual on Special Commands for full details on this command.

**For example**

```
>*HELP TOOLKIT AUTOSAVE <<filename>> CHECK <filename>
CLEAR
CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>>
       <<T>><<V>><<:>>
```

```
DUPLICATE <start>,<end>,<dest>
EDIT
ENVELOPE <<envno>>
ERASE <<start>>,<<end>>
ERROR <<toggle>>
FORMAT <trks>,<drv> <<drv>> <<drv>> <<drv>>
FKEY <<keyno>>
LMOVE <start>,<end>,<dest>
LOOK <filename>
MENU
MEMORY <<addr>>
MERGE <filename>
PARTSAVE <filename>/<start>,<<end>>
RECOVER <<+>> <<+>> <<+>> <<+>>...
RELOCATE <addr>
RENUMBER <<newstart>>,<<inc>>,<<oldstart>>,<<oldend>>
REPORT
ROMS
SCREEN <filename>
SEARCH <$>,<<C>> <<D>> <<R>> <<S>> <<G>> <<T>>
       <<">>/<<$2>>
SET
SLOMO <<delay>>
STATUS
SYNTAX <<P>>
TOOLS
TRACE <<start>>,<<end>>,<<S>>
VERIFY <<drv>> <<drv>> <<drv>> <<drv>>
XREF <<R>> <<I>> <<S>> <<A>> <<H>> <<F>> <<P>> <<C>>
     <<L>>
>
```

## LMOVE

**Syntax:** ***LMOVE <startline>,<endline>,<destinationline>**
**Minimum abbreviation:** ***LM.**
**Menu option:** **A**
**Function:** **Move Basic program lines.**

This command may also be called by pressing CTRL L from within the full screen editor.

This command will move one or more Basic program lines, between the specified start line and the specified end line, to a new position after the specified destination line number.

When program lines have been moved in this way the program line numbers will be out of sequence. Toolkit Plus automatically renumbers all the program lines after the destination line number in steps of ten to avoid problems when subsequently running or editing the program.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
>*LMOVE 20,40,1
>LIST
   10 FOR I%=0 TO 10
   20 PRINT "Toolkit Plus"
   30 NEXT I%
   40 REM example
>
```

## LOOK

**Syntax:** ***LOOK <filename>**
**Minimum abbreviation:** ***LOO.**
**Menu option:** **R**
**Function:** **List a Basic program directly from cassette or disc.**

The program is listed on the screen or, after CTRL B, to a printer in the same format as a program in memory is listed with the **LIST** command. Any program in memory is unaffected by this command.

As the program is scrolled off the screen it may be halted by pressing **CTRL** **SHIFT** or by putting the computer into paged mode by pressing **CTRL** **N**, before issuing this command.

An attempt to list a file that is not a Basic program will result in the message `Not Basic`.

**For example**

```
>*LOOK prog
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT Toolkit Plus
   40 NEXT I%
>
```

## MEMORY

**Syntax:** ***MEMORY <<address>>**
**Minimum abbreviation:** ***MEM.**
**Menu option:** **M**
**Function:** **Display a memory dump.**

This command will print a memory dump on the screen or, after **CTRL** **B**, to a printer. The memory contents displayed commences at the specified address or, if this is not defined, at the current value of PAGE. The dump is printed in paged mode and so printing will halt when a page of information has been displayed. The next page may be displayed by pressing the **SHIFT** key. The routine may be exited by pressing **ESCAPE**.

The format of the dump is automatically altered to suit the current display mode and text window.

**For example**

```
>*MEMORY
1900 0D 00 0A 0E 20 F4 20 65 .... . e
1908 78 61 6D 70 6C 65 0D 00 xample..
1910 14 10 20 E3 20 49 25 3D .. . I%=
1918 30 20 B8 20 31 30 0D 00 0 . 10..
1920 1E 15 20 F1 20 22 54 6F .. . "To
1928 6F 6C 6B 69 74 20 50 6C olkit Pl
1930 75 73 22 0D 00 28 09 20 us"..(.
1938 ED 20 49 25 0D FF 00 00 . I%....
1940 00 00 00 00 00 00 00 00 ........
1948 00 00 00 00 00 00 00 00 ........
1950 00 00 00 00 00 00 00 00 ........
1958 00 00 00 00 00 00 00 00 ........
1960 00 00 00 00 00 00 00 00 ........
1968 00 00 00 00 00 00 00 00 ........
1970 00 00 00 00 00 00 00 00 ........
1978 00 00 00 00 00 00 00 00 ........
1980 00 00 00 00 00 00 00 00 ........
1988 00 00 00 00 00 00 00 00 ........
1990 00 00 00 00 00 00 00 00 ........
>
```

## MENU

**Syntax:** ***MENU**
**Minimum abbreviation:** ***ME.**
**Function:** **Display a menu of Toolkit Plus commands.**

This command may also be called by pressing **ESCAPE** from within the full screen editor.

This command displays a menu of most of the Toolkit Plus commands from which they may be called with a single keystroke. Please see the section in this manual on Special Commands for full details on this command.

## MERGE

**Syntax:** ***MERGE <filename>**
**Minimum abbreviation:** ***MER.**
**Menu option:** **H**
**Function:** **Merge two Basic programs.**

This command will join the Basic program with the specified filename, on cassette or disc, with the program in memory. Any program lines in the program in memory, with the same line numbers as those in the program on disc or cassette, will be overwritten.

**For example**

```
>LIST
   10 REM example
   20 PRINT "Toolkit Plus"
   30 END
>*LOOK prog
   10 REM second example
   30 PRINT "for the BBC micro"
    40 END
>*MERGE prog
>LIST
   10 REM second example
   20 PRINT "Toolkit Plus"
   30 PRINT "for the BBC micro"
   40 END
>
```

## PARTSAVE

**Syntax:** ***PARTSAVE <<filename>>,<startline>,<<endline>>**
**Minimum abbreviation:** ***PA.**
**Menu option:** **C**
**Function:** **Save part of a Basic program.**

This command may also be called by pressing **CTRL** **P** from within the full screen editor.

The program lines between and including the specified start line and the specified end line are saved to the current filing system under the specified filename.

If no end line is specified, all program lines after and including the start line will be saved. If no start or end line are specified, the entire program is saved.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 PRINT "for the BBC micro"
   50 NEXT I%
>*PSAVE prog,30,40
>*LOOK prog
   30 PRINT "Toolkit Plus"
   40 PRINT "for the BBC micro"
>
```

## RECOVER

**Syntax:** ***RECOVER <<+>> <<+>> << + >> <<+>>...**
**Minimum abbreviation:** ***REC.**
**Menu option:** **G**
**Function:** **Recover a Basic program.**

Basic programs lost with a Bad prog ram error message may be partially or completely restored with this command.

The command ***RECOVER** alone will attempt to recover a program up to the first found logical end point. This may not be the end of the original program. Further program lines can be recovered by issuing the command with following +'s. Each + will recover further program lines up to a subsequent logical end point.

Any VDU control codes or corrupted characters found by the routine in the program being recovered are replaced with the # symbol. These may be easily found with the Toolkit Plus command ***SEARCH** (see below) and dealt with as required.

**For example:**

```
>LIST
Bad program
>*RECOVER
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
>*RECOVER +
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 PRINT "for the BBC micro"
   50 NEXT I%
>
```

## RELOCATE

**Syntax:** ***RELOCATE <address>**
**Minimum abbreviation:** ***REL.**
**Menu option:** **F**
**Function:** **Move a Basic program in memory.**

This command will move the entire Basic program at the current value of PAGE up or down in memory to a new specified address. The value of PAGE is changed to the new address.

The relocation address entered should be in hexadecimal and located at a memory page boundary, eg E00, 1900, 2000, etc. Values of the relocation address not at page boundaries will be truncated to the next lowest page boundary. Eg an address of E80 will be truncated to E00.

Please note that when a program is relocated to a lower memory address than the computer's default PAGE, program corruption or loss can result if **BREAK** is pressed.

Although Toolkit Plus can relocate a program anywhere in Ram, it is not advisable to relocate a program to an address below &0900.

**For example:**

```
>PRINT PAGE
1900
>*RELOCATE E00
>PRINT PAGE
E00
>
```

## RENUMBER

**Syntax:** ***RENUMBER<<newstartline>>,<<increment>>,<<oldstartline>>,<<oldendline>>**
**Minimum abbreviation:** ***RENU.**
**Menu option:** **B**
**Function:** **Wholly or partially renumber a Basic program.**

This command may also be called by pressing CTRL R from within the full screen editor.

All or part of a Basic program may be renumbered with this command. The lines from the specified old start line to the specified old end line are renumbered to begin at the specified new start line with the specified increment.

If no parameters are given, this command will renumber the entire program in increments of 10 starting with line 10. If the new start line is not specified the program will be renumbered to start from line ten. If the increment is not specified the program will be renumbered in steps often. If the old start line is not specified the program will be renumbered from the first line. If the old end line is not specified the program will be renumbered up to the last line.

| | |
|---|---|
| `*RENUMBER 100,5,30,150` | will renumber lines 30 to 150 in steps of 5 starting at line 100. |
| `*RENUMBER ,5,30,150` | will renumber lines 30 to 150 in steps of 5 starting at line 10. |
| `*RENUMBER 100,,30,150` | will renumber lines 30 to 150 in steps of 10 starting at line 100. |
| `*RENUMBER 100,5,,150` | will renumber lines from the start to line 150 in steps of 5 starting at line 100. |
| `*RENUMBER 100,5,30,` | will renumber lines 30 to the end in steps of 5 starting at line 100. |
| `*RENUMBER` | will renumber the entire program in steps of 10 starting at line 10. |

The Toolkit Plus renumber routine will not renumber a section of a program if this will cause an overlap of line numbers with another section, or an overflow above the line number size limit of 32765. Error messages are issued in such cases.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
>*RENUMBER 100,1,20,
>LIST
   10 REM example
   100 FOR I%=0 TO 10
   101 PRINT "Toolkit Plus"
   102 NEXT I%
>
```

## REPORT

**Syntax:** ***REPORT**
**Minimum abbreviation:** ***REP.**
**Menu option:** **4**
**Function:** **Display the last error message issued and the program line number at which it occurred.**

Issuing this command is equivalent to typing:

```
REPORT:PRINT " at line ";ERL
```

**For example:**

```
>*REPORT
Escape   at   line   150
>
```

## ROMS

**Syntax:** ***ROMS**
**Minimum abbreviation:** ***ROMS**
**Menu option:** **N**
**Function:** **Display a list of ROMs present in the computer and enable, disable or enter them.**

The names and types of all Roms present in the computer are displayed against the socket number in which they are located.

Against each Rom is one or more letters to indicate if the Rom has a service (S) or language (L) Rom entry point and whether it has Tube relocation points (R). When the list is displayed, individual Roms may be enabled or disabled to avoid command clashes, reclaim Ram workspace, etc. A disabled Rom is indicated by an X next to its entry in the list.

Roms are enabled and disabled using the keyboard. The cursor keys are used to select a Rom (displayed in blue text on white in mode 7 and in inverse video in the other modes) and the Space key is used to toggle the selected Rom on and off, RETURN will attempt to enter the language Rom currently selected. ESCAPE will end the selection process and exit the routine.

**For example:**

```
>*ROMS
  0 S  HELP II 2.00 ©1985 BEEBUGSOFT
X 1 SL EXMON II 1.03 ©1984 BEEBUGSOFT
  2 SL MUROM 1.2 ©BEEBUGSOFT 1985
  3 S  SLEUTH 1.06 ©1985 BEEBUGSOFT
  4 SL BCPL 7.0 © 1982 RICHARDS COMPUTER PR
  5 SL SPELLCHECK III 2.00 ©1985 BEEBUGSOFT
  6
  7 S  ARIES B20 v2.4 ©1984 David Barnett
  8
  9 SL WORDWISE PLUS 1.4A © 1984 CC
  A S  ICON MASTER 1.00 © 1985 BEEBUGSOFT
  B S  ROMIT 1.00 ©1985 BEEBUGSOFT
  C S  TOOLKIT PLUS 2.00 © 1985 BEEBUGSOFT
  D SL DFS 1.40 © 1983 A.C.Bray
  E LR BASIC ©1982 Acorn
  F
```

## SCREEN

**Syntax:** ***SCREEN <filename>**
**Minimum abbreviation:** ***SC.**
**Function:** **Save the current screen display.**

The current screen display, regardless of display mode, is saved to the currently active filing system (cassette or disc) with the specified filename.

The printing on the screen of filing system messages is suppressed during the operation of this command.

This command is not available from the Toolkit Plus menu.

**For example**

```
10 REM save screen on ESCAPE
20 ON ERROR IF ERR=17 THEN *SCREEN
```

## SEARCH

**Syntax:** ***SEARCH <string1>,<<string2>>**
**parameters <<C>> <<D>> <<R>> <<S>> <<G>> <<T>>**
**Minimum abbreviation:** ***SE.**
**Menu option:** **1**
**Function:** **Perform a string search, or search and replace, through the Basic program.**

This command may also be called by pressing CTRL S from within the full screen editor.

If this command is issued followed only by the search string then the program is listed with all occurrences of this string displayed in blue text on white in mode 7 and in inverse video in the other modes. The search will not find Basic keywords. However, see the control literal, K, below.

So, to highlight all occurrences of the word 'Beebug' in a program, you should issue the command ***SEARCH Beebug.** Note, however, that this command would not find the string, 'BEEBUG', in your program.

If a replace string is specified, at each match the routine halts and the prompt, `Replace Y/N:`, is displayed. Pressing Y will replace that occurrence of the match string with the replace string. Pressing N will continue the search to the next match. The command ***SEARCH Beebug, User Group** will locate every occurrence of the string 'Beebug' and give you the option of replacing it with the string 'User Group'.

After you have specified the search (and replace) parameters and pressed RETURN you will be asked if you wish to enter any parameters. These parameters control the way in which the search is conducted. To enter no parameters, simply press RETURN again.

There are seven possible parameters and each may be entered by keying a single letter. Just enter the appropriate letter/s (they do not need to be separated at all) and press RETURN. The parameters are as follows:

| | |
|---|---|
| C | Ignore the case of string being matched |
| S | Ignore spaces in the string being matched |
| D | Ignore matches in **DATA** statements |
| R | Ignore matches in **REM** statements |
| " | Ignore matches inside quote marks |
| K | Find Basic keywords |
| G | Global replace. `No This one Y/N:` prompt match. |

You may also use a wildcard character as any of the characters in your search string. The wildcard character is # and it may be considered to represent 'any one character'. For example, if you wished to replace all occurrences of `'proc1'` and `'proc2'` with `'proc3'`, you would specify `'proc#'` as your search string, `'proc3'` as the replace string and would selectively approve the alterations as required.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT I%
>*SEARCH 00
   Parameters
RETURN
   30 PRINT "Toolkit Plus"
>*SEARCH I%,count%
   Parameters
G RETURN
>LIST
   10 REM example
   20 FOR count%=0 TO 10
   30 PRINT "Toolkit Plus"
   40 NEXT count%
>
```

## SET

**Syntax:** ***SET**
**Minimum abbreviation:** ***SET**
**Menu option:** **S**
**Function:** **Define the first two user definable function keys for quick access of the menu and editor.**

This command will define the user definable function keys, zero and one, to call up, respectively, the Toolkit Plus full screen Basic program editor and the command menu.

**For example:**

```
>*SET
>*FKEY
 *KEY 0 *BEDIT¦M
 *KEY 1 *BMENU¦M
 *KEY 2
 *KEY 3
 *KEY 4
 *KEY 5
 *KEY 6
 *KEY 7
 *KEY 8
 *KEY 9
 *KEY 10
 *KEY 11
 *KEY 12
 *KEY 13
 *KEY 14
 *KEY 15
>
```

## SLOMO

**Syntax:** ***SLOMO <<delay>>**
**Minimum abbreviation:** ***SL.**
**Menu option:** **2**
**Function:** **Set the speed of the computer.**

This command will slow down all aspects of the operation of the computer when a Basic program is being executed. The resulting speed of operation can be varied between 100 different speeds by specifying a delay parameter between 0 (normal speed) and 99 (slow).

Specifying no delay will reset the speed of the computer to normal.

**For example**

```
>LIST
   10 TIME=0
   20 FOR I%=0 TO 5000:NEXT
   30 PRINT TIME
>*SLOMO
>RUN
         89
>*SLOMO 50
>RUN
         136
>
```

## STATUS

**Syntax:** ***STATUS**
**Minimum abbreviation:** ***ST.**
**Menu option:** **K**
**Function:** **Display machine status.**

This command will display the state of the most commonly used *FX commands, memory usage, and other features of the computer.

**For example:**

```
>*STATUS
 FX  3  0     FX  4  0
 FX  5  1     FX  6 10
 FX  9 25     FX 10 25
 FX 11 50     FX 12  4

 @% = &0000090A

 LISTO   0  WIDTH 255
 ERR     0  ERL   260

 REPORT : Escape

Free memory =23021 bytes
Program size= 3091 bytes
Next free Location=&2613
PAGE=&1A00   LOMEM=&2613
TOP =&2613   HIMEM=&8000
>
```

## SYNTAX

**Syntax:** ***SYNTAX <<P>>**
**Minimum abbreviation:** ***SY.**
**Menu option:** **I**
**Function:** **Toggle the automatic syntax checking facility.**

If the command, ***SYNTAX,** alone is issued, the automatic syntax checking facility is enabled. The command followed by any literal parameter other than P, such as ***SYNTAX D,** will disable the syntax checker.

When this facility is enabled all program lines are checked for correct syntax as they are entered from the keyboard, from the full screen editor or from an ASCII file with ***EXEC.** The automatic syntax checker will check for the following syntax errors:

Immediate mode command (AUTO, DELETE, LIST, NEW, OLD, RENUMBER, LOAD, SAVE) in a program line.

First item in a program line not a Basic keyword, variable name, indirection operator, =, or *

BGET, BPUT, CLOSE, EXT, and PTR without following #.

CLEAR, CLS, CLG, END, ENDPROC, REPORT, RETURN, RUN, and STOP followed by a Basic keyword except ELSE.

Uneven number of quotes in a statement.

Uneven number of brackets in a statement.

Invalid hex number.

Invalid variable name.

Type mismatch.

ENDPROC without preceding DEF PROC

Combination of keywords not able to be tokenized (eg. FALSEELSE). It should be noted that valid variable names that start with a Basic keyword, such as STOPPER, will also trigger the syntax checker warning to be printed.

If the literal, P, is specified as the parameter of this command, the entire Basic program currently in memory will be checked for syntax errors. Any errors found and the line numbers at which they occur are printed on the screen and, if the command is preceded with CTRL B, to a printer.

The syntax checker is not able to identify assembler mnemonics. It incorrectly identifies these as bad Basic. If the program being checked using the P option contains a lot of assembler, the continual error messages that this causes the syntax checker to issue may be prevented by pressing the SHIFT key as the program is listing. This temporarily disables the syntax checker.

**For example:**

```
>*SYNTAX
Syntax Checker On
>10 REM example

>20 FOR I% = 0 TO 10

>30 PRINT "Toolkit Plus
Missing " at line 30
>
```

## TOOLS

**Syntax:** ***TOOLS**
**Minimum abbreviation:** ***TO.**
**Menu option:** **O**
**Function:** **Display a list of the Toolkit Plus commands and their syntax.**

This command is identical in operation to the command, ***HELP TOOLKIT.** Please see the section in this manual on Special Commands for full details on this command.

**For example**

```
>*TOOLS
AUTOSAVE <<filename>>
CHECK <filename>
CLEAR
CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>> <<T>>
       <<V>> <<:>>
DUPLICATE <start>,<end>,<dest>
EDIT
ENVELOPE <<envno>>
ERASE <<start>>,<<end>>
ERROR <<toggle>>
FORMAT <trks>,<drv> <<drv>> <<drv>> <<drv>>
FKEY <<keyno>>
LMOVE <start>,<end>,<dest>
LOOK <filename>
MENU
MEMORY <<addr>>
MERGE <filename>
PARTSAVE <filename>,<start>,<<end>>
RECOVER << + >> << + >> << + >> << + >>...
RELOCATE <addr>
RENUMBER <<newstart>>,<<inc>>,<<oldstart>>,
         <<oldend>>
REPORT
ROMS
SCREEN <filename>
SEARCH <string1>,<<string2>>
SET
SLOMO <<delay>>
STATUS
SYNTAX <<P>>
TOOLS
TRACE <<start>>,<<end>>,<<S>>
```

```
VERIFY <<drv>> <<drv>> <<drv>> <<drv>>
XREF <<R>> <<I>> <<S>> <<A>> <<H>> <<F>> <<P>>
     <<C>> <<L>>
>
```

## TRACE

**Syntax:** ***TRACE <<startline>>,<<endline>>,<<S>>**
**Minimum abbreviation:** ***TR.**
**Menu option:** **3**
**Function:** **Enable the advanced trace facility.**

When enabled, the advanced trace facility will print, in the top right hand corner of the screen, the program line number currently being executed. A single step feature can be enabled by the S parameter. This will cause the program to pause before the execution of each program line and wait for any key to be pressed before resuming execution.

If a first program line is specified, only line numbers greater than that line will be reported. A second line number will cause the trace facility to report only line numbers between and including the two specified lines.

The trace facility is disabled after any error, including ESCAPE pressed, or by issuing the command with a literal parameter other than S, such as ***TRACE D**.

**For example:**

```
>*TRACE 10,1600,S
>
```

## VERIFY

**Syntax:** ***VERIFY <<drive>> <<drive>> <<drive>> <<drive>>**
**Minimum abbreviation:** ***V.**
**Menu option:** **U**
**Function:** **Verify the contents of one or more discs.**

This command will verify the disc in the specified drive(s). If no drive is specified, drive zero is verified.

Each track is checked for validity of the data on it and reliability to store further data. If a track does not verify successfully first time, a question mark is printed against the track number and further attempts are made. Five attempts are made to verify a faulty track. If all attempts to verify a track are unsuccessful an error message is printed.

**For example:**

```
>*VERIFY 0
Verifying track 15 ????? Error
Verifying track 16 ???
Verifying track 79
>
```

## XREF

**Syntax:** **•XREF <<R>> <<I>> <<S>> <<A>> <<H>> <<F>> <<P>> <<L>>**
**Minimum abbreviation:** ***X.**
**Menu option:** **5**
**Function:** **Print a cross reference listing of the Basic program.**

The numbers of the program lines containing reference to variables, procedures and functions are printed along with the current values of the variables.

The items listed are specified by the parameters following the command. These are:

R List real variables
I List integer variables
S List string variables

A List arrays

F List functions
P List procedures

The command, ***XREF,** alone defaults to listing all variables, procedures and functions.

In addition, three further optional parameters control the listing as follows:

H Contents of variables printed in hexadecimal
L Line numbers not printed

A hard copy of the cross reference may be obtained by typing **CTRL** **B** before issuing this command. You should note that Basic does not assign values to variables until your program has been run.

**For example:**

```
>LIST
   10 REM example
   20 FOR I%=0 TO 10
   30 A$="Toolkit Plus"
   40 NEXT I%
>RUN
>*XREF IH
I%                      &B      20    40
>
```

# THE FULL SCREEN EDITOR

Although BBC Basic has an effective copy line editor built in, a much easier way of editing your Basic programs is to use a full screen editor. This is the type of editor provided by Toolkit Plus. The Toolkit Plus full screen Basic program editor operates in a manner very similar to a word processor. It allows you to scroll back and forth through a Basic program making alterations and additions as you go, all with the minimum of effort.

## Entering the editor

The full screen editor may be entered by one of three routes. Firstly it may be called up directly from the keyboard with the command, ***EDIT.** For convenience you may choose to issue the ***SET** command and then enter the editor by pressing function key 0.

An alternative method is to enter the editor from the Toolkit Plus menu. The menu is displayed with the command, ***MENU**. This will print *a* table of the Toolkit Plus commands, selectable with a single keystroke. Pressing the ESCAPE key or the 7 key will take you into the full screen editor.

Pressing ESCAPE again from inside the editor will return control to the Toolkit Plus menu, regardless of whether the editor was entered from the menu or not. Pressing CTRL ESCAPE will return you straight to Basic.

If there is no program in memory when the editor is entered, or you have just issued the Basic command, NEW, the editor will display just the default first line number (10), ready for you to start to enter a new program.

The third method of entering the editor is by way of the advanced error handling facility. This is enabled with the command ***ERROR** in a Basic program. The advanced error handling facility will cause entry into the editor whenever an error is found in the running of the program.

## Moving the editing cursor

The editing cursor always occupies the line in the centre of the screen. As the cursor is moved through a Basic program it will remain on this line of the screen and the program will scroll above and below it.

If the editor is entered with the ***EDIT** command, the cursor is initially placed on the first character of the first program line in the program in memory. The cursor is also placed here if the editor is entered from the Toolkit Plus menu (by pressing ESCAPE) directly after a ***MENU** command. A line number may be specified in the command **(*EDIT <<line>>)** or if the editor is entered by way of the menu option 7. This will enter the editor with the editing cursor at the start of that line. If the ESCAPE key is used repeatedly to toggle between the menu and the editor, the cursor will remain on the program line it occupied before last exiting the editor.

If the editor is entered by way of the advanced error handling facility of Toolkit Plus, the editing cursor is placed near the statement which generated the error in the running of the program.

The editing cursor is moved using the four cursor control keys. Pressing the ↑ or ↓ keys will move the cursor up or down one screen display line. The horizontal cursor control keys will move the cursor one character at a time along a program line. Moving the cursor off the right hand end of a line will cause it to reappear at the left hand end of the next line down. Similarly moving off the left end will move it onto the next line up.

At all times the movement of the cursor is limited to the program statements only. Although the program line numbers are displayed these are not *directly* accessible from the editor.

The cursor may be moved to the end of the program by pressing **SHIFT** along with the **DOWN** key and to the start of the program by pressing **SHIFT** ↑.

Similarly, **SHIFT** pressed with ← or → will move the cursor to the left or right hand ends of the screen display line.

The **CTRL** key has a similar function to the **SHIFT** key. **CTRL** with the vertical cursor control keys will move the cursor up or down the program by half of the displayed screen. After pressing **CTRL** ↓, the line that the cursor was occupying will be displayed at the top of the screen and the cursor will now be on a new line half a screenful below it.

The **CTRL** key with the horizontal cursor keys will move the cursor along a line in steps of eight characters.

| | |
|---|---|
| ↑ | up one line |
| ↓ | down one line |
| ← | left one character |
| → | right one character |

| | |
|---|---|
| **SHIFT** ↑ | start of program |
| **SHIFT** ↓ | end of program |
| **SHIFT** ← | start of display line |
| **SHIFT** → | end of display line |

| | |
|---|---|
| CTRL ↑ | up half a page |
| CTRL ↓ | down half a page |
| CTRL ← | left 8 characters |
| CTRL → | right 8 characters |

## Altering Basic programs

When you have moved the editing cursor to the point in your program that is to be altered, new program statements can be entered from the keyboard in the normal way. On entry, the editor is in insert mode and new text will be inserted into the program, shifting up the text already at the cursor position. The insert mode is indicated by a small bar editing cursor. New text will form part of the program as soon as it is typed in. There is no need to press **RETURN**.

The Toolkit Plus full screen editor also supports an overwrite mode. This is selected by pressing **CTRL** **O**. The overwrite mode is indicated by the editing cursor taking a solid block character shape. Pressing **CTRL** **I** will revert to insert mode.

The ten user definable function keys may be programmed with long, or much used, text or Basic keywords for use within the editor.

The **COPY** key may be used to copy sections of a line into another position, in much the same way as it is used in the in built Basic editor. Pressing the **COPY** key automatically enters the copying facility. The cursor will now split into two - a large block cursor and a smaller cursor. The smaller cursor will be either the bar cursor, if you are currently using insert mode, or the smaller block cursor in overwrite mode.

The large block cursor remains on the line that you are currently editing. The smaller cursor is under control of the cursor control keys and may be moved to whichever line the copied text is to come from. Further presses of the **COPY** key will now copy text from the small cursor to the large block cursor. The copied text is inserted in, or overwrites, the editing line dependent on the mode you are currently using. The copying process may be terminated by pressing the **RETURN** key.

Program text may also easily be deleted from your program. The **DELETE** key operates in the normal manner. It will delete the character to the left of the cursor. If the editor is in insert mode, it will also close up any gap created in the program line. In overwrite mode the gap is left. If the **CTRL** **DELETE** is pressed, the character at the cursor position is deleted. Again, the gap is closed in insert mode.

At all times, it is worthwhile to keep an eye on program line length. Lines that cannot be compressed by the editor to within Basic's line length limit of 255 characters (after tokenization) will be truncated by Basic.

## Adding program lines

When it is desired to create a new line because, for example, the line you are editing is full, the **RETURN** key should be pressed. This will create a new line after the line currently being edited with a line number of one greater than that line.

If no program statement is typed in at that line it will be removed when the program is scrolled or **RETURN** is pressed again.

So, to create a new line numbered, say, 105 after line 100, you should move the cursor to line 100 and press **RETURN** five times and then enter the statements as required.

If the line currently being edited and the next program line use successive program line numbers it is not possible to insert a new line. In this case pressing **RETURN** will automatically cause Toolkit Plus to offer to renumber the program. If you approve this action, the entire program will be numbered in units of 10.

## Syntax checking within the editor

When the automatic syntax checking facility of Toolkit Plus is enabled (see the section on the ***SYNTAX** command in this manual), and a program is being edited using the full screen editor, any additional lines entered or changes to existing lines are checked for correct syntax.

If an altered program line is in error, this is detected when the editor updates its display, eg when it scrolls. An error message is displayed in the bottom line of the screen along with the line number of the program line in error.

## Issuing commands from within the editor

The Toolkit Plus full screen Basic program editor not only enables the easy editing of Basic program text but also allows several of the powerful Toolkit Plus commands to be called while editing a program, without leaving the editor.

Many of these commands require sections of the program to be selected -for deletion, moving, copying, and so on. Sections of the program in the editor are selected with either one or two special markers. These appear in the editor as asterisks between the program line number and program line itself. A marker is inserted at the line the editing cursor is occupying by pressing the TAB key. A marker may be removed by moving the editing cursor to the marked line and pressing the TAB key a second time. Only two markers are allowed in a program at any one time. Attempts to place a third marker in a program will be ignored by the editor.

Although you may cause a marker to be inserted on a line regardless of the position of your cursor, markers will in fact only be placed at the beginning of a program line. In other words, you may only mark entire program lines, not partial lines.

As you place markers, an indicator will appear at the top of the screen to remind you that they have been set.

The cursor can be quickly moved to a marked line by pressing CTRL TAB**.**

The commands that may be called from within the Toolkit Plus editor are all invoked by pressing the CTRL key along with the Toolkit Plus command's initial letter.

| | |
|---|---|
| **CTRL D** | Duplicate program lines (***DUPLICATE**) |
| **CTRL E** | Erase program lines (***ERASE**) |
| **CTRL L** | Move program lines (***LMOVE**) |
| **CTRL P** | Save all, or part, of the program (***PARTSAVE**) |
| **CTRL R** | Renumber the program (***RENUMBER**) |
| **CTRL S** | Perform a string search, or search and replace (***SEARCH**) |

There is also an additional command that may be entered from within the editor. This is **CTRL G**(for Go) which runs the Basic program in the editor.

These **CTRL** commands may also be entered by way of the user definable function keys. So, for example, to program function key two to run a program from within the editor, you would type (from Basic or the Toolkit Plus menu):

```
*KEY 2 ¦G
```

In the following command descriptions, **markline1, markline2,** and **cursorline** refer, respectively, to the line numbers of the first marked program line, the second marked line, and the line occupied by the editing cursor.

**Duplicating program lines**
**CTRL D**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command, ***DUPLICATE.**

The effect of the command will depend upon the number of markers present in the program.

No markers: This command has no effect.

One marker: Copy the marked program line to a position after the program line containing the editing cursor. This is equivalent to the Toolkit Plus command, **"DUPLICATE markline1,markline1,cursorline.**

Two markers: Copy the program lines between and including the marked lines to a position after the program line containing the editing cursor. This is the equivalent of the Toolkit Plus command, ***DUPLICATE markline1,markline2,cursorline.**

In all cases, after line copying the program lines after and including the line occupied by the cursor are automatically renumbered in steps of ten.

After this command, any markers that may have been set up will remain in position.

**Erasing program lines**
**CTRL E**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command,***ERASE.**

The effect of the command will depend upon the number of markers present in the program.

No markers: Erase the program line containing the cursor. This is equivalent to the Toolkit Plus command, ***ERASE cursorline.**

One marker: Erase the marked program line. This is equivalent to the Toolkit Plus command, ***ERASE markline1.**

Two markers: Erase the program lines between and including the marked lines. This is equivalent to the Toolkit Plus command, ***ERASE markline1,markline2.**

After this command, any markers that were set up will have been removed.

**Move program lines**
**CTRL L**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command, ***LMOVE.**

The effect of the command will depend upon the number of markers present in the program.

No markers: This command has no effect.

One marker: Move the marked program line to a position after the program line containing the editing cursor. This is equivalent to the Toolkit Plus command, ***LMOVE markline1,markline1,cursorline.**

Two markers: Move the program lines between and including the marked lines to a position after the program line con taining the editing cursor. This is equivalent to the Toolkit Plus command, ***LMOVE markline1,markline2,cursorline.**

In all cases, after line moving the program lines after and including the line occupied by the cursor are automatically renumbered in steps of ten.

After this command, any markers that were set up will have been removed.

**Saving all, or part, of the program**
**CTRL P**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command, ***PARTSAVE**.

The effect of the command will depend upon the number of markers present in the program.

No markers: Save the entire program to cassette or disc. This is equivalent to the Basic command, **SAVE "filename"** and the Toolkit Plus command, ***PARTSAVE filename .**

One marker: Save to cassette or disc the program lines after and including the marked program line. This is equivalent to the Toolkit Plus command, ***PARTSAVE filename,markline1,.**

Two markers: Save to cassette or disc the program lines between and including the marked lines. This is equivalent to the Toolkit Plus command, ***PARTSAVE filename,markline1,markline2.**

In all three cases, a prompt for the filename required is displayed in the bottom line of the screen.

After this command, any markers that may have been set up will remain in position.

**Renumbering the program**
**CTRL R**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command, ***RENUMBER.**

The effect of the command will depend upon the number of markers present in the program.

No markers: Renumber the entire program in steps of ten to start from line ten. This is equivalent to the Basic command, **RENUMBER** or the Toolkit Plus commands, ***RENUMBER**.

One marker: Renumber the program lines after the marked program line in steps of ten. This is equivalent to the Toolkit Plus command, ***RENUMBER markline1,10,markline1.**

Two markers: Renumber the program lines between and including the marked lines in steps of ten. This is equivalent to the Toolkit Plus command;
*__RENUMBER markline1,10,markline1,markline2.__

After this command, any markers that may have been set up will remain in position.

**Performing a string search or search and replace**
**CTRL S**

Issuing this command from within the full screen editor will enter a Toolkit Plus routine with a similar function to that entered with the command, ***SEARCH.**

This command will display a prompt for the search string in the bottom line of the screen. The search string, replace string and, if required, the control parameters should be entered together, in the same format as used when the ***SEARCH** command is issued in immediate mode. Please see the section on ***SEARCH** for further details on the format required.

When a match for the search string is found in the program, the editing cursor is moved to the first letter of the matching string and a prompt. This one Y/N:, is displayed in the bottom line of the screen. Pressing Y will now exit the search routine leaving the editing cursor at the matched string. Pressing N will continue the search for the next match.

When a replace string has been specified, and a search and replace operation is therefore initiated, the prompt, `Replace Y/N:` , is displayed in the bottom line and the matched string is only replaced with the replace string if the **Y** key is pressed. After replacement, or after **N** is pressed, the search continues for the next match. You may press **ESCAPE** at any time to terminate this operation.

The effect of the command will depend upon the number of markers present in the program.

No markers: The search will be made through the entire program.

One marker: The search will be made from that marker to the end of the program.

Two markers: The search will be made between the two markers only.

**Running the program**
**CTRL G**

Issuing this command from within the full screen editor will run the program in the same manner as though the editor had been exited with **CTRL ESCAPE** and the Basic command, **RUN** issued.

**Goto a line number**
**CTRL N**

Issuing this command from within the full screen editor will cause a prompt to ask for a line number. Specify a line number, press **RETURN** and the cursor will move to that line ready for editing in the normal way.

# COMMAND SUMMARY

## AUTOSAVE

**Syntax:** *AUTOSAVE <<filename>>
**Minimum abbreviation:** *AU.
**Menu option:** P
**Function:** Enable the automatic backup facility.

## CHECK

**Syntax:** *CHECK <filename>
**Minimum abbreviation:** *CH.
**Menu option:** Q
**Function:** Report differences between a program in memory and one on cassette or disc.

## CLEAR

**Syntax:** *CLEAR
**Minimum abbreviation:** *CL.
**Menu option:** D
**Function:** Reset the resident integer variables (A% - Z%).

## CRUNCH

**Syntax:** *CRUNCH <<B>> <<C>> <<L>> <<R>> <<S>> <<T>> <<V>> <<:>>
**Minimum abbreviation:** *CR.
**Menu option:** E
**Function:** Compact a Basic program.

## DUPLICATE

**Syntax:** *DUPLICATE <startline>,<endline>,<destinationline>
**Minimum abbreviation:** *DUP.
**Menu option:** 8
**Function:** Copy program lines within a Basic program.

**Called from the editor with CTRL D.**

## EDIT

**Syntax:** *EDIT <<line>>
**Minimum abbreviation:** *ED.
**Menu option:** Escape or 7
**Function:** Enter the full screen Basic program editor.

## ENVELOPE

**Syntax:** *ENVELOPE <<envelopenumber>>
**Minimum abbreviation:** *ENV.
**Menu option:** J
**Function:** List envelope definitions.

## ERASE

**Syntax:** *ERASE <<startline>>,<<endline>>
**Minimum abbreviation:** *ERA.
**Menu option:** 9
**Function:** Delete Basic program lines.

**Called from the editor with CTRL E.**

## ERROR

**Syntax:** *ERROR <<flag>>
**Minimum abbreviation:** *ERR.
**Menu option:** 6
**Function:** Enable the advanced error handling facility.

## FKEY

**Syntax:** *FKEY <<key number>>
**Minimum abbreviation:** *FK.
**Menu option:** L
**Function:** List the function key definitions.

## FORMAT

**Syntax:** *FORMAT <tracks>,<drive> <<drive>> <<drive>> <<drive>>
**Minimum abbreviation:** *F.
**Menu option:** T
**Function:** Format a disc.

## HELP

**Syntax:** *HELP TOOLKIT
**Minimum abbreviation:** *H. T.
**Function:** Display a list of the Toolkit Plus commands and their syntax.

## LMOVE

**Syntax:** *LMOVE <startline>,<endline>,<destinationline>
**Minimum abbreviation:** *LM.
**Menu option:** A
**Function:** Move Basic program lines.

**Called from the editor with CTRL L.**

## LOOK

**Syntax:** *LOOK <filename>
**Minimum abbreviation:** *LOO.
**Menu option:** R
**Function:** List a Basic program directly from cassette or disc.

## MEMORY

**Syntax:** *MEMORY <<address>>
**Minimum abbreviation:** *MEM.
**Menu option:** M
**Function:** Display a memory dump.

## MENU

**Syntax:** *MENU
**Minimum abbreviation:** *ME.
**Function:** Display a menu of Toolkit Plus commands.

## MERGE

**Syntax:** *MERGE <filename>
**Minimum abbreviation:** *MER.
**Menu option:** H
**Function:** Merge two Basic programs.

## PARTSAVE

**Syntax:** *PARTSAVE <<filename>>,<startline>,<<endline>>
**Minimum abbreviation:** *PA.
**Menu option:** C
**Function:** Save part of a Basic program.

**Called from the editor with CTRL P.**

## RECOVER

**Syntax:** *RECOVER <<+>> <<+>> <<+>> <<+>>...
**Minimum abbreviation:** *REC.
**Menu option:** G
**Function:** Recover a Basic program.

## RELOCATE

**Syntax:** *RELOCATE <address>
**Minimum abbreviation:** *REL.
**Menu option:** F
**Function:** Move a Basic program in memory.

## RENUMBER

**Syntax:** *RENUMBER <<newstartline>>,<<increment>>, <<oldstartline>>,<<oldendline>>
**Minimum abbreviation:** *RENU.
**Menu option:** B
**Function:** Wholly or partially renumber a Basic program.

**Called from the editor with CTRL R.**

## REPORT

**Syntax:** *REPORT
**Minimum abbreviation:** *REP.
**Menu option:** 4
**Function:** Display the last error message issued and the program line number at which it occurred.

## ROMS

**Syntax:** *ROMS
**Minimum abbreviation:** *ROMS
**Menu option:** N
**Function:** Display a list of ROMs present in the computer and enable, disable or enter them.

## SCREEN

**Syntax:** *SCREEN <filename>
**Minimum abbreviation:** *SC.
**Function:** Save the current screen display.

## SEARCH

**Syntax:** *SEARCH <string1>,<<string2>>
parameters <<C>> <<D>> <<R>> <<S>> <<G>> <<T>>
**Minimum abbreviation:** *SE.
**Menu option:** 1
**Function:** Perform a string search, or search and replace, through the Basic program.

**Called from the editor with CTRL S.**

## SET

**Syntax:** *SET
**Minimum abbreviation:** *SET
**Menu option:** S
**Function:** Define the first two user definable function keys for quick access of the menu and editor.

## SLOMO

**Syntax:** *SLOMO <<delay>>
**Minimum abbreviation:** *SL.
**Menu option:** 2
**Function:** Set the speed of the computer.

## STATUS

**Syntax:** *STATUS
**Minimum abbreviation:** *ST.
**Menu option:** K
**Function:** Display machine status.

## SYNTAX

**Syntax:** *SYNTAX <<P>>
**Minimum abbreviation:** *SY.
**Menu option:** I
**Function:** Toggle the automatic syntax checking facility.

## TOOLS

**Syntax:** *TOOLS
**Minimum abbreviation:** *TO.
**Menu option:** O
**Function:** Display a list of the Toolkit Plus commands and their syntax.

## TRACE

**Syntax:** *TRACE <<startline>>,<<endline>>,<<S>>
**Minimum abbreviation:** *TR.
**Menu option:** 3
**Function:** Enable the advanced trace facility.

## VERIFY

**Syntax:** *VERIFY <<drive>> <<drive>> <<drive>> <<drive>>
**Minimum abbreviation:** *V.
**Menu option:** U
**Function:** Verify the contents of one or more discs.

## XREF

**Syntax:** *XREF <<R>> <<I>> <<S>> <<A>> <<H>> <<F>> <<P>> <<L>>
**Minimum abbreviation:** *X.
**Menu option:** 5
**Function:** Print a cross reference listing of the Basic program.

**FULL SCREEN EDITOR COMMANDS**

| | |
|---|---|
| **ESCAPE** | Exit to Toolkit Plus menu |
| **CTRL ESCAPE** | Exit to Basic |
| | |
| **↑** | Move editing cursor up one line |
| **↓** | Move editing cursor down one line |
| **←** | Move editing cursor left one character |
| **→** | Move editing cursor right one character |
| | |
| **SHIFT ↑** | Move editing cursor to start of program |
| **SHIFT ↓** | Move editing cursor to end of program |
| **SHIFT ←** | Move editing cursor to start of display line |
| **SHIFT →** | Move editing cursor to end of display line |
| | |
| **CTRL ↑** | Move editing cursor up half a page |
| **CTRL ↓** | Move editing cursor down half a page |
| **CTRL ←** | Move editing cursor left 8 characters |
| **CTRL →** | Move editing cursor right 8 characters |
| | |
| **CTRL I** | Insert mode |
| **CTRL O** | Overwrite mode |
| **COPY** | Invoke character copy procedure |
| **RETURN** | Create new program line |
| | |
| **TAB** | Insert marker |
| **CTRL TAB** | Move to next marker |
| | |
| **CTRL D** | Duplicate program lines |
| **CTRL E** | Erase program lines |
| **CTRL L** | Move program lines |
| **CTRL P** | Save all, or part, of the program |
| **CTRL R** | Renumber the program |
| **CTRL S** | Perform a string search, or search and replace |
| **CTRL G** | Run the Basic program |
| **CTRL N** | Goto a line number |

# TOOLKIT PLUS ADDENDUM

## COMMAND ABBREVIATIONS

The minimum abbreviation for Toolkit Plus commands is now two characters, this will help reduce command conflicts.

## COMMAND NAME CONFLICTS

We would draw your attention to the section concerning Toolkit Plus commands on page 3 of the manual. A number of other Roms will inevitably have similar commands to some of those in Toolkit Plus. For example, Disc Doctor has its own *EDIT, *VERIFY, *RECOVER and *MENU commands. In such a case you should insert your Toolkit Plus into a socket with higher priority than Disc Doctor; Now duplicated commands such as *EDIT, will be intercepted by Toolkit Plus. When you wish to use Disc Doctor's command of the same name, simply prefix it with the letter B in the opposite case. For example, *bEDIT

Users of both Toolkit Plus and Murom need to be aware of a command name conflict. *ENVELOPE has been used in both products, and once again we advise you to insert Toolkit Plus into *a* higher priority socket. Then, to access the Toolkit Plus ENVELOPE command use *ENVELOPE and to use Murom's command enter *bENVELOPE (Note the lower case b).

## *SEARCH AND BASIC TOKENS

To save memory, Basic stores each keyword in memory as a single byte code called a token. For example, the keyword PRINT is stored as the number &F1. It is therefore not possible to search for part of a keyword (such as PRIN).

Certain Basic keywords must always be followed by an opening bracket, and so this bracket is included by Basic as a part of the token. Therefore, if you wish to search for any of the following keywords, the opening bracket must also be included as a part of the search string:
INSTR( LEFT$( MID$( POINT$( RIGHT$( STRING( TAB(

The Basic keywords HIMEM, LOMEM, PAGE, PTR and TIME are each stored as two different tokens by Basic, depending upon how they are used. One token is used in cases where the keyword is assigned a value by the statement (such as TIME=0), a different token is used in all other cases (such as NOW%=TIME).

The *SEARCH command will always locate the first situation, where the keyword is given a new value. To search for occurrences of the other keywords, for example to locate TIME in 'NOW%=TIME', you should use the wildcard character in your search string. -ie a search string of #TIME (see page 27 of the Toolkit Plus manual for a description of wildcards).

## OUT OF SEQUENCE LINE NUMBERS

It is possible to create a program with line numbers which are out of sequence. Programs of this nature must be renumbered before being used in the editor.

## SYNTAX CHECKER

The syntax checker may on very rare occasions draw your attention to a line of Basic that it suspects to be in error, but which after inspection proves to be correct. This will occur rarely and is caused by the complexity of the Basic language.

The syntax checker should be regarded as a warning system, and once you have acknowledged its message you may choose whether to alter the statement in question or leave it unchanged. However, in the overwhelming number of cases, the statement in question will be a genuine error.

## MARKERS

Markers placed in the editor will be removed by the following commands: GOTO, DUPLICATE, ERASE, LINEMOVE, RENUMBER and ESCAPE.

## EPROM FITTING INSTRUCTIONS

## WARNING!

Please read through these instructions completely before starting. EPROMs are very delicate, never place the pins of the ROM near or in contact with plastic or artificial fibres or any other source of electricity.

1. Disconnect the computer from the mains socket.

2. Unscrew the pair of screws at the top of the back panel, which might be labelled FIX.

3. Unscrew the pair of large screws, similar to those removed from the back, positioned near the front by the feet of the computer. They may be labelled FIX.

4. Lift the lid of the computer.

5. Carefully unscrew the two (or maybe three) nuts and washers holding the keyboard to the computer case. The screws are located about two inches behind the ones removed in stage three above.

6. Lift the keyboard and carefully turn it over and lay it on the back part of the computer.

7. On the very bottom right of the circuit board there are five 28-pin sockets. These contain the operating system, EPROMs and ROMs.

8. Locate the Basic ROM. This is usually in the second socket from the left, and can be identified by a serial number on it's surface ending with either B01 or B05.

9. Remove the Basic ROM, unless it is in the far right socket, by placing a screwdriver under the end of the chip and gently twisting. Never force a chip into or out of a socket. Be careful not to scratch the printed circuit board or bend any of the pins. Always pull the chip out vertically and not from one end as this can fracture the pins at the other end.

10. Insert the Basic ROM into the far right socket, making sure that the small notch at one end of the chip is pointing towards the back of the computer.

11. Insert the new EPROM in any of the spare sockets, once again make sure the notch points towards the rear of the computer.

12. Put the keyboard back in it's normal position, and switch the computer on. It should now function as normal, however if it does not, switch it off and check that all the pins on the ROM and EPROM are inserted correctly. It is possible that a pin is bent beneath the chip and therefore out of sight.

13. Screw the keyboard and lid back on the computer. Turn the machine on and type *HELP <return> which will display the title of all the ROMs in your computer (except Basic).

14. Your new EPROM is now ready for use, please refer to the manual for a full explanation of the facilities.

N.B. We have been informed that some model B's and upgraded Model A's do not switch between ROM's. This is because I.C.76 (a 74LS163) is not fitted, or wire links S12 and S13 are not cut (open circuit). Therefore to enable the paging of ROMs, I.C.76 has to be fitted and links S12 and S13 have to be cut. This should already be done on a Model B.

# Serious Software For The BBC Micro


Beebugsoft offer a unique range of programs for BBC micro users. Our titles include:

- Toolkit Plus
- RomIt
- Studio Eight
- Icon Master
- Wordease
- Program Builder
- Spellcheck III
- Help II
- Sleuth
- Exmon II
- Discmaster
- Dumpmaster
- Masterfile II
- Murom
- Quickcalc
- Starter Pack
- Hershey Characters
- Billboard
- Design
- Paintmaster
- Teletext
- Sprites


For further information on any of these products, please write to **Beebugsoft**, PO Box 50, St. Albans, Herts.

Telephone Orders 049481 6666
Telephone Queries 0727 40303