Premier Publications Britain's biggest hobby software specialist
12 Kingscote Road Addiscombe Croydon Surrey Telephone 01-656 6156

TOOLKIT II for UK101 and OHIO systems

TOOLKIT II is supplied in EPROM to locate at 8000hex. When fitting the chip ensure that it is placed the right way round. If you are using it in conjunction with our Mini EPROM/ROM board, TOOLKIT should be placed in the old BASIC 1 slot. If you have just switched on your machine, you MUST cold-start BASIC before entering TOOLKIT.

TOOLKIT is initialised by RESET M 8000G which will produce the on-screen message:-

TK II V1.x

<C> 1981 P.Rihan/PREMIER

TOOLKIT II is now enabled and may be used immediately - the annoying OM error after a warm start of BASIC has been corrected by TOOLKIT. If at any time you exit BASIC for machine code, RESET W will restart TOOLKIT II and BASIC unless the warmstart vector has been altered. If other parts of Page Zero have been affected, a BASIC coldstart may be necessary first, or if you have BASIC 4 installed, use RESET M C and answer memory size with OLD.

TOOLKIT II commands

MC

MC provides a quick exit from BASIC to the machine code monitor at FE00hex. Re-enter BASIC via 0000 G or RESET W.

VIEW

This command allows you to examine the contents of a cassette without loading it to memory. Pressing the SPACE bar (or Q for quit) will cancel VIEW in the usual way.

VIEW is useful where lines of BASIC have misloaded and you wish to attempt a partial reload. Type VIEW, wait until the relevant lines are on the screen, press SPACE, then copy the displayed lines into your program.

TRACE

TRON and TROFF (Trace on and off) gives you a comprehensive trace/delay function. The current line number being executed is displayed at the top of the screen as shown

[1125]

and a delay loop is entered to enable it to be seen. If faster execution is required, CTRL will bypass the delay and the program will run at near normal speed.

TRON and TROFF may be activated from within a program - useful if you are only debugging a subroutine, for instance.

POKE 756,255 enable

POKE756,0 disable

TRON/TROFF is called through the CTRL C vector and any routine calling this will pass through. This includes LIST (but not LIST/ - see later), and thus provides a useful way of controlling the LIST speed.

AUTO xx/yy

AUTO will generate a new line number starting at xx and incrementing by yy. Either or both may be omitted, in which case default is to line 10, incrementing in tens.

Any start or increment that will not cause lines larger than 63999 may be used, but yy may not be zero. If the next line to be created is greater than 63999, a ?OF ERROR (overflow) is generated, and command mode entered.

When using AUTO, the text of the line should be entered after the appearance of the new line number. If a particular line is not required, RETURN will generate the next line. To exit AUTO mode, type CTRL T.

If during AUTO mode, an existing line is encountered, a star will be printed to the left of the line number, thus

```
  230 REM  new line made
  240 REM second new line
* 250  this line already exists !
```

If RETURN is pressed when line 250 is generated, the previous contents of that line would be deleted. To avoid this it is necessary to RUBOUT the line number before pressing RETURN. If the old contents of the line are not required, enter the line in the usual way.

DUPL xx/yy

allows you to copy the entire contents of line xx into line yy. If line yy already exists, its contents are overwritten, and if the line to be copied does not exist a ?LL ERROR will occur and command-mode will be re-entered.

WARNING ! Either xx or yy will default to 10 if not specified.The value yy must not be 0.

LIST xx-yy/zz

This function lists in the normal way, but pauses after zz program lines have been displayed. Pressing any key will continue the list for another zz lines. If zz is not set, default is to 10 lines. Values xx and yy are the start and end requirements for LIST and are optional. For example:-

LIST/ will list the whole program ten lines at a time.
LIST/6 will list the whole program six lines at a time.
LIST100-3479/9 will list the program from lines 100 to 3479 in blocks of 9 lines.

A ?SN ERROR is caused if less than 1 or more than 32 lines are set in zz. To exit at a pause press CTRL T. This routine does not call CTRL C via Trace and is therefore unaffected if Trace is enabled.

DELETE xx-yy

will erase program lines from xx to yy inclusive. Values xx and yy may be specified exactly the same as for LIST/ and default in the same way (xx to 0 and yy to the last program line).

DIRE WARNING !! If DELETE or DELETE- are entered, then RETURN is pressed, your program will be irretrievably lost, so take care in using this command.

RENUM xx/yy

will renumber a program from xx in increments of yy. Values xx and yy will default to 10 as for previous routines, and yy must not be zero. RENUM provides a complete renumber, with all GOTO, GOSUB, THEN, etc being corrected and space being created in a line where the number of digits of a line number has increased. Examples:-

RENUM100/5 will renumber program to start at line 100 and increment in fives.
RENUM1/1 will renumber program to start at line 1 in increments of 1.
RENUM will renumber the program to start at line ten in increments of ten.

The following points should also be noted:-

1/ If the renumbered program would have numbers in excess of 63999 a ?OF (overflow) ERROR is caused and no renumbering takes place.
2/ If, within a program, a call to a non-existent is found, a Lf ERROR is generated and no renumbering takes place.
3/ If a tokenised, renumbered line will not fit into the input buffer for re-insertion, a BF ERROR is printed, but renumbering continues. The offending line will remain in the program and will be renumbered, but its contents will not be changed and will need to be altered manually.
4/ NEVER attempt a RESET during renumbering, or your program will be left in a half-renumbered state.

FIND xx-yy/string

will search the resident program from xx to yy (default as for LIST) for up to a character string. The string must not be empty, and initial spaces are ignored. All lines are searched, but only the first occurence on each line is indicated. The listing will pause when the string is found and any key will continue the search.
For example

FIND 1-999/GOSUB100 will find any GOSUB100 statements between lines 1-999
FIND/B=1:X=2 will find all occurences of B=1:X=2 in the program.

At a pause, CTRL T will exit this mode.

REPL xx-yy/OLD$/NEW$

provides a Global Search and Replace system usually only found in W.P. packages.
The program is searched from xx to yy (defaults as per LIST) for the (up to) 20 character string in OLD$ (syntax as FIND) and indicates its appearance. When found, REPL? is offered, and if Y is pressed, OLD$ is replaced by the string in NEW$ (20 character maximum). Any other key will continue the search without changing, but it is a good idea to use the keys N and Y for answering REPL? For example:-

REPL1-999/GOSU20/GOSUB38

will search through the program from lines 1 to 999, and where GOSUB20 is found will offer REPL? Pressing key Y will effect the change to GOSUB38, relist the line and continue the search. Pressing key N will search for the next GOSUB20 without changing the one just found.

REPL/Spelingg/Spelling

will search through the whole program for 'Spelingg' and will replace it with 'Spelling' where key Y is pressed.
A null string is acceptable for NEW$, which allows a useful block delete.
As for FIND, all lines are searched, but only the first appearance of OLD$ is indicated. However, if REPL? is accepted, the line is re-searched and the next OLD$ pointed out. Should the amended line exceed the input buffer, a ?BF (buffer full) ERROR will occur and restart command mode. If REPL? is rejected, the program will NOT search through the rest of that line for another OLD$.
A certain amount of care is needed in using this powerful function.

1/ If the NEW$ contains the OLD$ as a part of itself, it will be found again after REPL? is done and any other changes to be made on that line cannot be accessed. If you wish to avoid a REPL? early in a line, but make the change later in it, accept the first REPL? and change it back later.

2/ Don't accept all changes offered in the above case if they are not necessary - program can easily degenerate into a time-consuming tangle !

3/ Don't try to replace single letter variables/numbers unless absolutely necessary - they can crop up in the most awkward places. For example, trying to replace the variable G in a program using REPL/G/G7 will find every G in the program (in GOSUB, etc).To get round it you could replace all GOSUBS with G7OSUBS to access the G's further down the line, then REPL/G7OSUB/GOSUB afterwards.

4/ In general, specify the changes to be made as fully and exactly as possible - this will avoid too many unnecessary REPL?'s. Also read the rest of the line carefully.

LIST LINE ON ERROR

All the BASIC error messages have now been corrected and when an error occurs during program execution the offending line is automatically listed, and CEGMON's screen editor turned on (where CEGMON is resident). Note that the edit cursor will not necessarily appear on-screen at the start of the displayed line.

The Stack is now reset on warmstart so that the inevitable and infuriating ?OM ERROR will no longer occur after immediate mode commands on re-entry to BASIC.

INFORMATION

TOOLKIT II commands are entered in full in immediate (command) mode - they cannot be used within program lines. The only function that can be enabled / disabled within program is Trace, by POKEing 756 as indicated above.

If you have BASIC 5 installed, you can initialise TOOLKIT by typing &GO$8000 .

FOR THE MACHINE CODERS !

TOOLKIT II makes extensive use of the Input, Output and CTRL C vectors and most of the routines alter and rewrite them. The warmstart jump is also used. If you wish to connect a machine code routine to these vectors, its initialisation should copy the current contents elsewhere (to be used for your call to the vector), and write its own addresses in. The prompt jump (0003h) should also be copied and a routine linked there should rewrite your entries into the vectors (NOT recopy them) and exit via the old jump. Great care is needed here - it is all to easy to end up exiting to the entry point of your own routine ! Clearly your 'coldstart' initialise routine would be called by X=USR(X) after TOOLKIT II had been initialised. This does not cover all cases as some TOOLKIT functions do not return via the prompt - but RESET W should get round that.

If you have purchased a non-CEGMON version of TOOLKIT and later decide to buy CEGMON from PREMIER, return TOOLKIT with your order and we will replace it with a CEGMON version free of charge.

TOOLKIT II was written by P Rihan and is sold under licence by PREMIER PUBLICATIONS