Program Development Made Easy

Andrew Rowland describes some utilities which make editing Basic programs much more effective, using Edit on a Master, or View on a model B.

For me, one of the joys of owning a Master is being able to edit programs in a text editor instantly, without all that tedious SPOOLing and EXECing. However, even the Master's Editor has some disadvantages: there is a limit on the length of program that can be transferred to the editor, and once line numbers are removed, you cannot return to Basic without EXECing.

This article presents four utilities: a pair so that BBC owners can edit programs in View with the same convenience as on a Master, and a pair to better facilitate using the Master's Edit. Both pairs may be used to edit without line numbers.

EDITING WITHOUT LINE NUMBERS

The first thing I usually do on entering EDIT is to remove line numbers: it makes things a lot easier when copying or moving blocks around, you don't have to worry about squeezing enough new line numbers between two others, and I have a library of sub-routines saved without line numbers in ASCII format. These can then be inserted anywhere in a program while in the editor. It also encourages good discipline about not using GOTOs! You do have to avoid using RESTORE with line numbers, but with care, it is not necessary.

ENTERING THE PROGRAMS

If you have a Master and use Edit, you will need listings 1 and 2, which produce the utilities XED and XBAS. Model B owners wanting the convenience of the Master (or Master owners wedded to View) will find listings 3 and 4 fit the bill. This time the utilities are called VED and VBAS, and load your program into View for editing.

If you are entering the listings from the magazine, listing 4 is not reproduced as it is similar to listing 2, but with the following modifications:

```
 10 REM Program .>VBASbas (4)
380 INY:INX
940 a$="SAVE VBAS "+STR$~install+" "
    +STR$~P%
```

EDITING WITH LINE NUMBERS

The utilities can be modified as shown below to retain line numbers. For example, you may only want XED in order to avoid the 'No room' error that occurs with lengthy programs (maybe you should consider reducing the length of the program by CHAINing shorter ones or using overlays)! In this case, don't use XBAS - return to Basic in the usual way.

Listings 1 and 3: remove *JMP out* from line 610. Listings 2 and 4: delete lines 640 - 700. Master users will find listing 5 does the same job as listing 4 for less typing!

I do not use XED and XBAS exclusively, but use the normal route when I want line numbers present to find my place. View users may want the choice too. Create VED and VBAS normally, rename them to something appropriate, then perform the above modifications.

USING THE COMMANDS

Both sets of utilities work in the same way: type *XED or *VED in Basic's command mode and they will load the current program into Edit or View in the same way as the Basic command EDIT, but without line numbers. You should always save your work first, as the program will be corrupted if anything goes wrong. And if you are using View, it cannot cope with lines longer than 132 characters, so keep your program lines short - it improves readability anyway!

Then you may edit your program using all the word processor's facilities. Of course, you must ensure that no lines exceed 251 characters and that the last line is terminated by a carriage return. You cannot include Ctrl-@ (which appears black-on-white), as it is used as an end of file marker.

When your editing is over, you need to return to Basic. From Edit, press Shift/f4 (return to language) and enter 'XBAS' where you would usually type 'BASIC'. When the word *Escape* appears, your program is ready to list. It will have line numbers starting at 10 in increments of 10 - if you had any special numbering system, it will have been lost.

From View, you return to Basic by typing *VBAS in View's command page.

HOW THE UTILITIES WORK

Basic programs cannot be edited in a text editor: as you no doubt know, Basic has its own special format for programs, where Basic keywords like PRINT and ENVELOPE are replaced by a one byte code number called a *token*. There are also line numbers and line length bytes. Text editors need files containing only recognisable characters, so to edit an existing program, it must first be converted into plain text (sometimes called ASCII), and afterwards converted back into Basic's internal format - this is called detokenising and tokenising.

Listing a program effectively detokenises it: the program is displayed in readable characters, just as you typed it in. XED and VED make Basic list the program too (by inserting LIST in the keyboard buffer), but instead of sending the output to the screen, the characters produced are stored in memory. A flag is set when a new line starts and five characters are missed, so that line numbers are not stored along with the rest (unless you disabled it). In addition, line feeds (ASCII code 10) which are sent at the end of lines are also skipped.

To do this, the OS routine used to write characters to the screen is diverted to a replacement routine via its vector at &20E, which is responsible for storing the characters in memory. It detects when the listing has ended by the appearance of the '>' prompt, and restores the vector.

Before this happens though, mode 7 is selected and the Basic program moved up in memory as far as it will go. The ASCII text is stored below the program starting at PAGE. This is to permit longer programs to be edited than the EDIT command can cope with (EDIT stores the ASCII text *above* the program so that if all the program cannot be detokenised, it is not corrupted and lost). Detokenised programs take up more room than Basic's internal format, but some overlap will not matter. Nevertheless, save your program first, especially if it *very* long.

To prevent problems, LIST and WIDTH are forced to zero. After this, the editor is

invoked and - if you are using Edit - you will be informed that there is text already in memory.

The way this is done may interest some readers. *EDIT or *EDIT <*filename*> also has an extended form: *EDIT *nn,nn* where *nn* are two hexadecimal digits. These are not the start and end addresses of the text in memory but vectors to them. Thus if you invoke the editor with *EDIT 00,02 then locations &00/&01 and &02/&03 should contain the start and end addresses of the text respectively.

On the other hand, View has to be tricked into thinking it has loaded a file. The command *L ASC* (load a file called ASC - a dummy name I use as a default) is inserted into the keyboard buffer as if it had been typed. The vector used by OSFILE to load files is diverted to a replacement routine. Then a *WORD is issued, and the *L ASC* executed. However, the filing system never gets a chance to see the command: it is dealt with by the replacement routine which tells View how long the 'file' is (which is already in memory), resets the vector and exits.

RETURNING TO BASIC

The utilities which return the ASCII program to Basic also employ a diverted vector, this time of OSWORD 0, the routine Basic uses to obtain a line of input. Whenever you enter a new line of a program at the '>' prompt, this routine is called by Basic. By diverting it, we can copy a line of our ASCII text into Basic's input buffer in page 7, and so fool it into thinking the line has been typed at the keyboard in the usual manner. Moreover, by ensuring the first line so copied is "AUTO", Basic will automatically add the line numbers as it goes. When the last line has been 'entered' like this, the vector is restored to its former contents.

The text can be found easily by XBAS and VBAS because Edit and View always arrange the text in memory to start one page above OSHWM when 'Return to language' is used and place a zero byte at the end. It too, is moved as far up in memory as it will go, because although in general Basic programs are shorter than their ASCII equivalents, the fact that the line numbers are not present means that the tokenised program *can* get longer than the text as they are added, especially with long assembler listings which have little opportunity to benefit from tokenisation.

Finally, a note on listing 5, which is only for Master users who want to use View to create programs *with* line numbers. It uses the same mechanism as Edit does when returning to Basic - it places the address of the start of text in locations &00/01 and calls *BASIC @.

Finally, a reminder that a version of XED that works with Bas128, called *BEDIT*, was published in BEEBUG Vol.9 No.4 on page 40.

Listing 1

```
 10 REM Program .>XEDbas (1)
 20 REM Version B1.00
 30 REM Machine Master/Compact
 40 REM Author  Andrew Rowland
 50 REM BEEBUG  July 1991
 60 REM Program subject to copyright
 70 :
100 wrchv=&20E:oswrch=&FFEE
110 osbyte=&FFF4:oscli=&FFF7
120 top=&12:page=&18:himem=&6
130 listo=&1F:width=&23
140 pointer=&70:start=&72
150 :
```

```
160 FOR pass=0 TO 3 STEP 3
170 P%=&900:[OPT pass
180 .install LDA #22:JSR oswrch
190 LDA #135:JSR oswrch
200 \ alter WRCHV
210 LDA wrchv  :STA oldv
220 LDA wrchv+1:STA oldv+1
230 SEI:LDA #entry MOD &100:STA wrchv
240 LDA #entry DIV &100:STA wrchv+1
250 CLI
260 \ move program up
270 LDA #132:JSR osbyte \ read HIMEM
280 STY himem+1:STX himem
290 DEY:LDX #0
300 .over STY pointer+1:STX pointer
310 LDY top:STX top:BEQ zero
320 .loop LDA (top),Y:STA (pointer),Y
330 DEY:BNE loop
340 .zero LDA (top),Y:STA (pointer),Y
350 DEC top+1:DEC pointer+1
360 LDX top+1:CPX page:BCS loop
370 LDX pointer+1:INX:STX page
380 LDA #2:STA pointer:STA start
390 LDA #131:JSR osbyte \ read OSHWM
400 STY pointer+1:STY start+1
410 \ WHENO 0, WIDTH 0
420 LDX #0:STX listo:DEX:STX width
430 \ put "L." in k/b buffer
440 LDA #21:LDX #0:JSR osbyte:LDY #0
450 .loop LDA auto,Y:JSR poke
460 INY:CPY #3:BNE loop
470 RTS
480 .auto EQUS "L."+CHR$13
490 \ perform *FX138,0,n
500 .poke TAX:TYA:PHA:TXA:TAY
510 LDA #138:LDX #0:JSR osbyte
520 PLA:TAY:RTS
530 \ ****************************
540 .entry STY ytemp
550 LDY flag:BNE secondtime
560 CMP #13:BNE eout:INC flag
570 .eout LDY ytemp:RTS
580 .secondtime CMP #10:BEQ out
590 LDY count:BEQ ok
600 CMP #ASC">":BEQ finished
610 DEC count:JMP out
620 .ok CMP #13:BNE notcr
630 LDY #5:STY count
640 .notcr LDY #0:STA (pointer),Y
650 INC pointer:BNE out:INC pointer+1
660 LDY pointer+1:CPY himem+1:BNE out
670 BRK:BRK:EQUS "No room. Press BREAK
":BRK
680 .out LDY ytemp:RTS
690 :
700 .finished SEI
710 LDA oldv  :STA wrchv
720 LDA oldv+1:STA wrchv+1:CLI
730 LDX #string MOD &100
740 LDY #string DIV &100
750 JMP oscli
760 .string EQUS "EDIT "+STR$~start+",
"+STR$~pointer:EQUB 13
770 :
780 .oldv    EQUW 0
790 .flag    EQUB 0
800 .ytemp   EQUB 0
810 .count   EQUB 5
820 ]NEXT
830 a$="SAVE XED "+STR$~install+" "+ST
R$~P%
840 PRINT a$:OSCLI a$
```

Listing 2

```
 10 REM Program .>XBASbas (2)
 20 REM Version B.00
 30 REM Machine Master/Compact
 40 REM Author  Andrew Rowland
 50 REM BEEBUG  July 1991
 60 REM Program subject to copyright
 70 :
100 wordv=&20C:wrchv=&20E
110 osbyte=&FFF4:oscli=&FFF7
120 oswrch=&FFEE
130 zp=&70:block=&72:buffer=&74
140 FOR pass=0 TO 3 STEP 3
150 P%=&900:[OPT pass
160 .install \ disable ESCAPE
170 LDA #200:LDX #1:JSR osbyte
180 LDA #&7C:JSR osbyte
190 JSR setvecs:JSR findend:JSR movup
200 LDA #0:STA flag
210 LDX #string MOD &100
```

Continued on page 54

```
220 LDY #string DIV &100
230 JMP oscli
240 .string EQUS "BASIC":EQUB 13
250 :
260 .setvecs LDA wordv:STA oldv
270 LDA wordv+1:STA oldv+1
280 LDA wrchv  :STA oldv+2
290 LDA wrchv+1:STA oldv+3
300 SEI:LDA #entry MOD &100:STA wordv
310 LDA #entry DIV &100:STA wordv+1
320 LDA #rts   MOD &100:STA wrchv
330 LDA #rts   DIV &100:STA wrchv+1
340 CLI:RTS
350 :
360 .findend
370 LDA #131:JSR osbyte \ read OSHWM
380 INY \ add INX for VBAS
390 STX zp:STX zp+2
400 STY zp+1:STY zp+3:LDY #0
410 .loop LDA (zp+2),Y:BEQ foundit
420 INC zp+2:BNE loop
430 INC zp+3:BNE loop
440 .foundit RTS
450 :
460 .movup
470 LDA #&84:JSR osbyte \ read HIMEM
480 TXA:SEC:SBC #16:BCS over:DEY
490 .over STX buffer:STY buffer+1
500 LDY #0
510 .loop LDA (zp+2),Y:STA (buffer),Y
520 OPT FNdec(zp+2):OPT FNdec(buffer)
530 LDA zp+3:CMP zp+1:BNE loop
540 LDA zp+2:CMP zp  :BNE loop
550 LDA (zp+2),Y:STA (buffer),Y
560 LDA buffer+1:STA zp+1
570 LDA buffer  :STA zp:RTS
580 \ ******************************
590 .entry CMP #0:BEQ forme
600 JMP (oldv) \ not OSWORD 0
610 .forme STX block:STY block+1
620 TAY:LDA (block),Y:STA buffer
630 INY:LDA (block),Y:STA buffer+1
640 LDA flag:BNE secondtime:LDY #0
650 .firsttime
660 LDA auto,Y:STA (buffer),Y
670 INY:CPY #4:BNE firsttime
680 INC flag:CLC:RTS
690 .auto EQUS "AU.":EQUB 13
700 :
710 .secondtime LDY #&FF
720 .loop INY:CPY length:BEQ finished
730 LDA (zp),Y:STA (buffer),Y
740 BEQ finished
750 CMP #&0D:BNE loop
760 INY:TYA:CLC \ eol
770 ADC zp:STA zp:BCC rts
780 INC zp+1:CLC
790 .rts RTS
800 :
810 .finished SEI
820 LDA oldv  :STA wordv
830 LDA oldv+1:STA wordv+1
840 LDA oldv+2:STA wrchv
850 LDA oldv+3:STA wrchv+1:CLI
860 \ re-enable ESC
870 LDA #200:LDX #0:JSR osbyte
880 SEC:RTS
890 :
900 .oldv    EQUD 0
910 .length  EQUB 251
920 .flag    EQUB 0
930 ]NEXT
940 a$="SAVE XBAS "+STR$~install+" "+S
TR$~P%
950 PRINT a$:OSCLI a$
960 END
970 :
1000 DEFFNdec(address)
1010 LOCAL jump
1020 IFaddress<&100 jump=4 ELSE jump=5
1030 [OPT pass
1040 DEC address:LDX address
1050 INX:BNE P%+jump:DEC address+1
1060 ]=pass
```

Listing 3

```
10 REM Program .>VEDbas (3)
20 REM Version B1.00
30 REM Author  Andrew Rowland
40 REM BEEBUG  July 1991
50 REM Program subject to copyright
60 :
100 wrchv=&20E:filev=&212
```

```
110 oswrch=&FFEE:osbyte=&FFF4
120 oscli=&FFF7:top=&12:page=&18
130 himem=&6:listo=&1F:width=&23
140 pointer=&90:start=&92:block=&94
150 :
160 FOR pass=0 TO 3 STEP 3
170 P%=&900:[OPT pass
180 .install LDA #22:JSR oswrch
190 LDA #135:JSR oswrch
200 \ alter WRCHV
210 LDA wrchv  :STA oldv
220 LDA wrchv+1:STA oldv+1
230 SEI:LDA #entry MOD &100:STA wrchv
240 LDA #entry DIV &100:STA wrchv+1
250 CLI
260 \ move program up
270 LDA #132:JSR osbyte \ read HIMEM
280 STY himem+1:STX himem
290 DEY:LDX #0
300 .over STY pointer+1:STX pointer
310 LDY top:STX top:BEQ zero
320 .loop LDA (top),Y:STA (pointer),Y
330 DEY:BNE loop
340 .zero LDA (top),Y:STA (pointer),Y
350 DEC top+1:DEC pointer+1
360 LDX top+1:CPX page:BCS loop
370 LDX pointer+1:INX:STX page
380 LDA #1:STA pointer:STA start
390 LDA #131:JSR osbyte \ read OSHWM
400 INY:STY pointer+1:STY start+1
405 LDY #0:LDA #13:STA (start),Y
410 \ WHENO 0, WIDTH 0
420 STY listo:DEY:STY width
430 \ put "L." in k/b buffer
440 LDA #21:LDX #0:JSR osbyte:LDY #0
450 .loop LDA auto,Y:JSR poke
460 INY:CPY #3:BNE loop
470 RTS
480 .auto EQUS "L."+CHR$13
490 \ perform *FX138,0,n
500 .poke TAX:TYA:PHA:TXA:TAY
510 LDA #138:LDX #0:JSR osbyte
520 PLA:TAY:RTS
530 \ ******************************
540 .entry STY ytemp
550 LDY flag:BNE secondtime
560 CMP #13:BNE eout:INC flag
570 .eout LDY ytemp:RTS
580 .secondtime CMP #10:BEQ out
590 LDY count:BEQ ok
600 CMP #ASC">":BEQ finished
610 DEC count:JMP out
620 .ok CMP #13:BNE notcr
630 LDY #5:STY count
640 .notcr LDY #0:STA (pointer),Y
650 INC pointer:BNE out:INC pointer+1
660 LDY pointer+1:CPY himem+1:BNE out
670 BRK:BRK:EQUS "No room. Press BREAK
":BRK
680 .out LDY ytemp:RTS
690 :
700 .finished SEI
710 LDA oldv   :STA wrchv
720 LDA oldv+1 :STA wrchv+1
730 LDA filev  :STA oldv
740 LDA filev+1:STA oldv+1
750 LDA #fentry MOD &100:STA filev
760 LDA #fentry DIV &100:STA filev+1
770 CLI
780 \ preserve lowest 2 bytes
790 LDY #0:LDA (start),Y:STA ytemp
800 INY:LDA (start),Y:STA ytemp+1
810 \ push "L ASC" into k/b buffer
820 LDA #21:LDX #0:JSR osbyte
830 LDY #0
840 .loop LDA view,Y:JSR poke
850 INY:CPY #6:BNE loop
860 LDX #string MOD &100
870 LDY #string DIV &100
880 JMP oscli
890 .view   EQUS "L ASC"+CHR$13
900 .string EQUS "WORD":EQUB 13
910 :
920 .fentry CMP #5:BEQ cat
930 CMP #&FF:BEQ load
940 JMP (oldv)
950 .load LDA oldv:STA filev
960 LDA oldv+1:STA filev+1
970 .cat LDA block:PHA
980 LDA block+1:PHA
990 TXA:PHA:TYA:PHA
1000 STX block:STY block+1
```

```
 1010 LDA #0:LDY #2
 1020 .loop STA (block),Y:INY
 1030 CPY #&12:BNE loop
 1040 LDY #&A:LDA pointer:SEC
 1050 SBC start:STA (block),Y
 1060 LDA pointer+1:SBC start+1
 1070 INY:STA (block),Y
 1080 \ restore 2 lowest bytes
 1090 LDY #0:LDA ytemp:STA (start),Y
 1100 INY:LDA ytemp+1:STA (start),Y
 1110 PLA:TAY:PLA:TAX
 1120 PLA:STA block+1:PLA:STA block
 1130 LDA #1:RTS
 1140 :
 1150 .oldv   EQUW 0
 1160 .flag   EQUB 0
 1170 .ytemp  EQUB 0
 1180 .count  EQUB 5
 1190 ]NEXT
 1200 a$="SAVE VED "+STR$~install+" "+ST
R$~P%
 1210 PRINT a$:OSCLI a$
```

Listing 5

```
  10 REM Program .>VBASbas(M)  (5)
  20 REM Version B1.00
  30 REM Machine Master/Compact
  40 REM Author  Andrew Rowland
  50 REM BEEBUG  July 1991
  60 REM Program subject to copyright
  70 :
 100 osbyte=&FFF4:oscli=&FFF7
 110 FOR pass=0 TO 3 STEP 3
 120 P%=&A00:[OPT pass
 130 LDA #180:LDX #0:LDY #&FF
 140 JSR osbyte \ read OSHWM
 150 INX:STX 1:STZ 0
 160 LDX #string MOD &100
 170 LDY #string DIV &100
 180 JMP oscli
 190 :
 200 .string EQUS "BASIC@":EQUB 13
 210 ]NEXT
 220 a$="SAVE VBAS A00 "+STR$~P%
 230 PRINT a$:OSCLI a$
```

Personal Ads

BEEBUG members may advertise unwanted computer hardware and software through personal ads (including 'wants') in BEEBUG. These are completely free of charge but please keep your ad as short as possible. Although we will try to include all ads received, we reserve the right to edit or reject any if necessary. Any ads which cannot be accommodated in one issue will be held over to the next, so please advise us if you do not wish us to do this. We will accept adverts for software, but prospective purchasers should ensure that they always receive original copies including documentation to avoid any abuse of this facility.

We also accept members' Business Ads at the rate of 40p per word (inclusive of VAT) and these will be featured separately. Please send us all ads (personal and business) to MEMBERS' ADS, BEEBUG, 117 Hatfield Road, St. Albans, Herts AL1 4JS. The normal copy date for receipt of all ads will be the 5th of each month.

Prism modem 2000 for sale (no software or instructions) - Offers? Elite 5.25", Revs 5.25" boxed, complete offers, **WANTED:** BEEBUG magazines Vol.1 No.1 to Vol. 2 No.9 and BEEBUG discs before May '86 or will swap for above. Tel. (0705) 678410 anytime.

Epson RX80 printer £60, BBC Master ROMs, Printmaster £8, ADT £10, Interword £25, Master £12, on disc; (Superart, Pagemaker, Mouse) £30, Elite £8, Hyperdriver £5, Astaad £5, Master Reference manuals 1&2 £15, Voltmace joystick and splitter box £8, Watford User Port Splitter £10, 3 Quad cartridges £7 each. Tel. (0475) 38502.

BBC B issue 7, with Torch Z80 CP/N double disc attachment, almost unused Torch perfect, comes with full software perfect writer, perfect filer etc. BBC A issue 7, with upgrade kit of chips to bring up to B specification (unused). Electron, used, with joystick attachment and modem connector (serial-port type attachment) £ . Tel. 091-271 5966.

WANTED: I've been given an ATPL board that I want to fit to my old BBC B Micro, the trouble is someone cut all the wires taking it out! and I have no instructions. Could anyone owning an ATPL ROM/RAM board please photocopy and send me the fitting and operating instructions? Also does anyone own a ROM/RAM board made by Peartree Ltd, (it has a socket to take the O.S ROM, four empty sockets and four yellow wires with a blue plug on the end of each). Once again I have no paper work, can anyone help with a photo copy of the fitting instructions. I will be happy to pay any cost involved if anyone can help on this. Tel. (0322) 664761 eves or 071-494 1365 office hours.

Archimedes 310 colour system, 1Mb RAM, 2x 3.5" disc drive, RISC OS, PC Emulator, Amstrad SM2400 modem and Hearsay, will split £100 o.n.o. Tel. (0494) 764643 after 6pm.

WANTED: BEEBUG Vols.1-8 complete, plus Vol.9 Nos.1,2 and 3, reasonable prices paid. Tel. (0792) 201848.

A3000 colour system, 1Mb, RISC OS, E-Type, Fun School 2, little used, mint condition, must sell £490 or offer. Tel. 081-360 8076 or 071-956 5488 office hours.