EXDIS

-ON EPROM-

EXTENDED DISASSEMBLER

FOR THE BBC MICRO MODEL A OR B

By ANDREW LORD

INTRODUCTION

EXDIS is a sophisticated machine code disassember written especially for the BBC micro. It provides a wide range of facilities including the following:

(i) Full disassembler allowing disassembly of paged ROMs and files on disc.
(ii) Full labeling capacity, including markers.
(iii) Hex/ASCII dumps and memory editing.
(iv) Line assembler with labels allowed.
(v) Full expression analysis including variable/label names.
(vi) Full sideways ROM support.
(vii) Allows use of all Operating System commands within EXDIS.

EXDIS is useful for debugging machine code programs especially when used with a machine code monitor such as EXMON, or for discovering how someone elses machine code program operates. The function keys con be used to enter frequently used commands. For example *KEY 0 L 8000 807F|M (RETURN) will cause EXDIS to list memory from &8000 to &807f whenever function key f0 is pressed.

NOTE: To take full advantage of all the features of EXDIS your machine should be fitted with Acorn's Basic ROM, version I or II.

STARTING INSTRUCTIONS

(i) Calling EXDIS

Once the EPROM containing EXDIS has been installed, EXDIS con be entered by the command *DIS (RETURN), where (RETURN) means press the Return key. This can be abbreviated to *D (RETURN) for quick access (NOTE that there is no full stop after *D). So that full use of EXDIS can be made, this command should be entered from Basic or a machine code monitor and not, for example, from a wordprocessor EPROM. Pressing break at any time during the running of EXDIS will re-enter EXDIS. If any labels were defined then you will be presented with the option of saving them. To quit EXDIS type Q (or use CRTL/Break). Type OLD (RETURN) to recover any resident Basic (or assembler) program.
NOTE: defining any labels will have the effect of overwriting you Basic program and so it is advisable to save your program before entering EXDIS should you want to use labels.

An address may also be specified (in hex) after the command *DIS, e.g. *DIS 1900. Disassembly will automatically start from this address on entry.

It is possible once in EXDIS to switch into a monitor such as EXMON ,e.g., for single stepping and then return to EXDIS (with *D) for disassembly. Note: answer 'Y' to prompt to keep labels on your return to EXDIS.

Should you accidently leave EXDIS by option Q or CTRL/Break then it is

possible to retrieve labels defined by entering *D as normal and replying 'Y' to the prompt. The labels will still be intact - except possibly the first 2 bytes of the first label will be overwritten by &0D &FF by Basic. This will give the label an address of &FF0D. You can use the command 0 FF0D to find out the name of this variable.

(ii) EXDIS version

Typing *HELP followed by Return will give the normal Help message. This should now include EXDIS will its version number.

GENERAL OVERVIEW

(i) Commands

Commands are entered by means of one or two characters to identify the command, followed by a number of parameters. The command summary at the end of this manual gives a detailed description of the precise syntax and default values.
There is also a HELP facility within EXDIS which will give a brief on-screen summary of the commands, this is activated by typing '?'.

Throughout this manual the following will be used to describe the parameters:

<address> a 2 byte address e.g. 81E5
<range> a range in memory specified by a 2 byte start address and a 2 byte end address, e.g. 8100 82E4.
<byte> a single byte hex value, e.g. 5D.

(ii) Numbers

The numbers used by the commandscan be entered in several ways:

(a) as a simple hex number. Terminate with (RETURN) only if there are less than the maximum number of digits, e.g. when entering a single digit rather than 2 for <byte> or when entering less than 4 characters for <address> etc. Note: do not prefix with &, hex is assumed.

(b) by default i.e. (RETURN) alone is pressed. This has different effects depending on the command. In general a single address or the start address of a range will default to the current address in the memory, and the second address of a range will default to 'infinity' (i.e. the command will continue indefinitely). Note that the second address can be entered relative to the first, using the '+' operator. eg M 9000 +FF is eqivalent to M 9000 90FF.

(c) as an expression, using the '#' option (see below). This allows decimal values to be input.

(iii) Expression Evaluation

(a) The 'C' command causes evaluation of an arithmetic expression, displaying the result as a 4 byte hex integer and decimal equivalent. Negative results are printed in 16 bit complememt form (ie the result -10 is printed as 65526 etc). The expression may contain +,-,*,/,^,brackets to any level, and Basic variable names. In particular this includes labels used in the source code (provided they are still defined).
NOTE: The expression evaluator makes use of routines in the Basic interpreter, and so reverts to the Basic convention of assuming decmal. You must therfore prefix hex numbers by &. If the Basic ROM is not installed, attempting to call the expression evaluator will result in an error message to this effect. EXDIS

is compatible with versions 1 and 2 of BBC Basic in this respect.

(b) The '#' option. This allows fully general expressions to be entered for any of the parametes for any command, if '#' is entered as the first character after the command letter, e.g. entering D #START #START+22 will disassemble 23 bytes from the memory location given by 'START' (which may be a source code variable). However before the expression is passed to Basic for evaluation EXDIS checks to see if you have entered a label name. If so then the address corresponding to that label will be taken. e.g. L #for #next will list memory between labels 'for' and 'next' assuming the have been defined with the command '#'.

(iv) Operating System Commands

All normal calls to the operating system (commands prefixed by *) are accepted by EXDIS, so that it is possible to set user keys, make *FX calls, perform *LOADs, etc. all from within EXDIS. In particular *KEY can be used from within EXDIS to set up often used command sequences.

(v) Zero-page Memory

EXDIS uses location &40-&6F as workspace and so does not conflict with Basic or EXMON.

(vi) Input correction

<Delete> will cancel the current entry, except on expression or label entry.
(escape> will escape from any command.

1. MEMORY COMMANDS

L=LIST
Syntax: L <range>
Lists memory within given range in both hex and ASCII. The second address can be omitted and the listing will continue indefinitely - press Escape to stop. Page mode is engaged so page through with Shift as usual. Control codes are replaced by a full stop in the ASCII column.

M=ASCII LIST
Syntax: M <range>
As L ,above, only the memory is displayed in ASCII only.

D=DISASSEMBLE
Syntax: D <range>
Disassembles the given range of memory listing standard 6502 mnemonics, hex and ASCII with labels replaceing the appropriate address. '???' signifies an unrecognised opcode. Colour is used to highlight branches, unknown opcodes and jumps. This option can be toggled on/off with the command Z. The disassembler also 'homes in' on labels - see the command I for more details. Page mode is engaged.

K=DISASSEMBLE & SAVE
Syntax: K <range> <filename>
Disassembles the range of memory, spooling to the specified file. The first record of the file will contain the Basic command "AUTO", so that when the file is *EXEC'd from Basic, a Basic program will be created. This can be edited and run to reassemble the original section.
Notes: i) Branch instructions are disassembled with relative addressing e.g. BEQ P%+7
ii) Unknown opcodes are replaced with OPT FNequb(&XX) on BASIC I and EQUB &XX on BASIC II to allow correct reassembly.

N=ASSEMBLE
Syntax: N <address>
  Enters the line assembler at the specified address. Input 6502 mnemonics (only one per line) and these will be assembled at the address displayed. Labels may be used as in Basic.
  <return> moves forward 1 byte.
  <spacebar> <return> moves back 1 byte.
  <escape> terminates assembly.
Note: This calls the assembler in the Basic ROM.

E=EDIT
Syntax: E <address>
  Enters the memory editor at the specified address.
Editor commands:
  <byte>   a byte value entered is stored at the current location, and you move to the next byte.
  <return>   move to the next byte without changing.
  <spacebar> move back one byte.
  <escape>   finish editing.

"=INPUT STRING
Syntax: " <address> <string>
  Stores ASCII characters input from the keyboard starting at the specified address. Control characters can be entered. (a full stop will be displayed) Terminate entry by pressing Escape (the Return key enters the Return character &0D and is not therefore used to terminate this command).

FS=FIND STRING
Syntax: FS <range> <string>
  Searches for an ASCII string in the given memory range. Any occurances will be listed, giving the address of the start of the string.
Note: Do not be confused by unexpected finds, such as the buffer where EXDIS stores the search string or in the screen memory !

FB=FIND BYTE
Syntax: FB <range> <bytes>
  As for FB, but a string of bytes is entered.
Note: the lower 8 bits of the entered byte are taken if you use # <expession>.

B=BASIC LIST
Syntax: B <page>
  Lists the program at the given page in Basic format. The default value is the current PAGE setting.
Note: this command does not decode GOTO or GOSUB linenumbers correctly.

2. LABELING

  EXDIS allows the user to define up to about 3100 labels. These are stored in memory above the default setting of PAGE (normally &E00 on tape machines). Labels can be up to 7 characters in length. They come into their own during disassembly when any referance to a labeled location is replaced by the appropriate label.

```
e.g.    8100  LDA &FF     A9 FF    ..
        8102  BMI escape  30 04    ..
        8104  JMP noesc   4C AB 81 ...
escape  8107  BRK         00       .
         ..    ..         ..       .
```

  Notice that disassembly takes place in five columns : label, address, opcode, hex and ASCII. This is the normal layout used by assemblers and so leads to a very readable code.
  Note that all major operating system calls are already labeled, but do not

take up any RAM. They cannot be redefined or deleted.
  Markers may also be set. Markers are like labels only they don't show up in the disassembly - they just mark an address. They can be used to mark the end of data areas in a program so that the disassembler can 'home in' on them for a better disassembly without giving the address a label (see the command I).
  Details of where the labels are stored in memory is given at the top of the screen. The start and end addresses are given.

#=SET LABEL
Syntax: # <label> <address>
  This sets the label to the address given.
Note: if the label is less than 7 characters in length then you may press (RETURN) and then enter the address.
e.g. # T (RETURN) A012
  Markers may also be set by pressing TAB after the #. You will then be prompted for an address.

&=DELETE LABEL
Syntax: & <label>
  Deletes the given label from memory. To delete a marker use & and then TAB. You will then be prompted for the address of the marker to be deleted.

0=LIST LABELS
Syntax: 0 <address>
  Lists all labels including markers and operating system labels in order of address starting at the given address. The default address (obtained by entering 0 (RETURN)) is 0000. Listing stops after address &FFFF has been reached. Escape may be used to terminate the command before it reaches this limit.

V=CALCULATE
Syntax: V <label>
  Prints out the value of the given label ,if it exists, in hex.

>=SAVE LABELS
Syntax: > <filename>
  Saves all labels currently in memory to tape/disc under the given filename.

<=LOAD LABELS
Syntax: < <filename>
  Loads labels saved under the previous command.
Note: i) This command clears all labels that were currently defined.
     ii) EXDIS cannot tell the differance between a file containing, say, a Basic program and one containing labels. Thus it is possible to load in a Basic program by accident causing havoc!. If this happens simply issue this command again with the correct filename.

+=APPEND LABES
Syntax: + <filename>
  This operates as for load above, only it adds the labels to those currently in memory.

3. SYSTEM VARIABLES

O=SET OFFSET
Syntax: O <old address> <new address>
  Offsets all addresses entered after this command by a value such that a referance to <new address> gives information on <old address>.
E.g. O 2000 8000 will mean that D 8000 8100 will disassemble the memory at &2000 to &2100 with all addresses reset as if it were at &8000.
  This is useful on tape systems so that a program can be examined without overwriting memory used by labels or Basic variables.

P=CLEAR OFFSET
Syntax: P
  Turns the offset function off.
Note: an indicator is given at the top of the screen as to whether the offset is in operation.

Z=TOGGLE COLOUR
Syntax: Z
  Toggles the colour highlight option on disassembly.
Note: an indicator 'Col' is given at the top of the screen if this option is active.

I=TOGGLE INTELLIGENCE
Syntax: I
  Toggles the ability of the disassembler to 'home in' on any labels defined by the user. This enables the disassembler to skip over data areas and start disassembling from a label on a new line. Data skipped over in this way is given by '?D?' in the opcode column.
E.g. if memory at &2000 reads - 2000 : F5 A2 00 0A ... with a label TEST defined as &2001, then this would normally produce :

```
        2000  SBC &A2,X     F2 A2   ..
        2002  BRK           00      .
        2003  ASL A         0A      .
```

With intelligence ,however, this becomes :

```
        2000  ?D?           F2      .
TEST    2001  LDX #0        A2 00   ..
        2003  ASL A         0A      .
```

with the byte at &2000 being the end of a list of data, for example. This is how it was meant to be read !
Note: 'Int' is displayed at the top of the screen is this operation is engaged.

@=SET MEMORY
Syntax: @ <address>
  This places a value in the memory. It is useful to store an address in the memory when ,for example, studying nested subroutines to keep track of where you started from !
Note: the current value held in memory is given at the top of the screen under 'Mem'.

4. FILES

(SPACE)=GET FILE
Syntax: (SPACE) <filename>
  This allows the user to apply any of the memory commands to a file on disc. The file will 'overlay' the memory addresses as given in the files catalogue on disc. Any commands directed at this area of memory will cause EXDIS to get the appropriate information from the file on disc.
Note: i) because this feature uses random access files it cannot be impliamented on the tape system.
     ii) you must not remove the disc with the program on from the current drive.

A=CLOSE FILE
Syntax: A
  Closes a file opened using (SPACE). It prompts with "Are you sure ?" so that you cannot select this function without knowing it.
Note: the file will also be closed after the command Q or on pressing BREAK.

5. PRINTER

H=PRINTER SELECT
Syntax: H <command>
  The command following the 'H' will be directed at the printer. This will be cancelled before the next command.
Note: you may have to configure your printer before hand by entering the appropriate *FX calls - see User Guide pp.422+

HXD=EXTENDED DISASSEMBLY
Syntax: HXD <range>
  This gives an extended disassembly of the memory in the given range on the printer. In addition to the normal disassembly it also gives a column of the opcodes without any labels giving a quick referance to ,say, the address of a subroutine otherwise hard to find by its label name in 8K of code !. It also spaces out the hex values for a neater appearance.

6. ROMS

  Although EXDIS resides at the same address as the other paged ROMs, it is capable of listing and disassembling them. To do this first select the required ROM by entering ! <rom-id> where rom-id is the number between 0 and &F specifing the position of the required ROM. Command T may be useful here. With the standard machine, C,D,E and F refer to the 4 sockets from left to right (though there is redundancy so that 0,4,8 and C all refer to the leftmost socket). Initially (i.e. after *DIS or Break) the Basic ROM will be selected regardless of where it resides.

!=SELECT ROM
Syntax: ! <rom-id>
  Select current ROM which will reside between &8000 and &BFFF in memory.
Syntax: ! <return>
  This gives the number of the currently selected ROM.
Note: the current ROM number is also displayed at the top of the screen.

T=LIST ROMS
Syntax: T
  This gives a list of the ROMs situated in the machine. Any ROMs which are aot active i.e. not recognised by the operating system are shown in magenta.

Y=ROM DETAILS
Syntax: Y <rom-id>
  This supplies details on the given ROM including its name and copyright message. It also gives its language entry print, its service entry point and second processor relocation address if appropriate. For example Beebugs Toolkit is a service ROM and so does not have a language entry point and so none will be given.
Note: This command only works on 'active' ROMs.

EXDIS COMMAND SUMMARY

1.GENERAL

*EXDIS (or *D) (RETURN)	Enter EXMON.
?	Display HELP summary
BREAK	Re-initialise EXDIS
Q	Quit EXDIS.
*	Permits all normal calls to operating system
H <command>	Send output of command to printer.

2. MEMORY LISTING AND EDITING

```
L <range>                 List memory (hex. & ASCII).
M <range>                 List memory (ASCII).
D <range>                 Disassemble memory.
K <range> <filename>      Disassemble and save to file.
N <addr>                  Assemble.
E <addr>                  Edit memory:
  editor commands:
        <byte>      change byte, move to next byte.
        <return>    move to next byte.
        <space-bar> move to previous byte.
        <escape>    exit editor.
" <addr> <string> <escape> Input ASCII string to memory.
FS <range> <string>       Find ASCII string in given range.
FB <range> <string>       Find string of hex bytes.
B <page>                  List Basic program.
```

3. LABELS

```
# <label> <addr>          Set label.
& <label>                 Delete label.
V <label>                 Calculate label value.
0 <addr>                  List labels.
< <filename>              Load labels.
> <filename>              Save labels.
+ <filename>              Append labels.
```

4. ROMS

```
! <rom-id>                Select ROM.
! <return>                Display number of current ROM.
R <rom-id>                Save ROM.
T                         List ROMs.
Y <rom-id>                Give ROM details.
```

5. OTHERS

```
I                         Toggle inteligence.
Z                         Toggle colour highlight.
@ <addr>                  Set memory.
O <addr> <addr>           Set offset.
P                         Clear offset
(SPACE)                   Get file.
A                         Close file.
C <expression>            Calculate value of expression.
```

6. SYNTAX DEFINITIONS

<addr> An address in hex of up to four digits. Terminate with Return is less than 4 hex digits. Return on empty input defaults to memory (except command 0).

<range> Two addresses (a start address and an end address) in hex of up to 4 digits. Both addresses as <addr> above. Return on an empty second address defaults to continuous printing (except on commands K and H). An alternate form is <addr> +<addr> where the second "addr" is an increment. (This is achieved in such a way that, for example, the range 8100 +9 is taken to be the range 8100 8109 inclusive). As suggested above <addr> may be replaced by #<expression>.

<byte> A byte of data input as two hex digits (i.e. 00 to FF), or a single hex. digit followed by Return, or #<expression>.

<expression> A general expression involving +,-,*,/,^, brackets, variable names. Expressions must be proceeded by #. You may enter a defined label as an expression as #<label>.

<filename> A name to be used when saving or loading to or from tape/disc.

7. ERROR MESSAGES

The following may be produced at one time of other in the use of EXDIS.

escape -- escape pressed during input or listing/disassembling.

No such label -- when trying to delete or calculate an non-existant label.

No BASIC ROM -- when using a command requiring the Basic ROM if the ROM is not fitted in your machine.

Bad program -- when using the command B and an error has occured in the line length of a particular Basic line.

O.S. label -- produced if you try to delete/redefine an Operating System label e.g. osword.

Bad filename -- if no filename is given or the filename exceeds the maximum number of characters (7 on a disc system).

No such marker -- if you try and delete a non-existant marker.

Disc system only -- if you try and use the command (SPACE) with the tape/ROM filing system.

File not found -- if you try to load a non-existant file from Disc.

Already closed -- using command A if a file has not been opened by command (SPACE)

Can't open -- if the filing system cannot open a file for some reason.

Bad Hex -- if you give a bad hex. number on the entry line e.g. *D 89FG.

Continuous not allowed -- if you try and list/disassemble memory continuously when using the printer or the command K.

-- THE END --