APPENDIX IV
EXMON II COMMAND SUMMARY

1. General

*EXMON <byte>	Cold start EXMON, specifying workspace page.
*E <byte>	Cold start EXMON, specifying workspace page
*EXMON	Warm start EXMON, with default workspace
*E	Warm start EXMON, with default workspace.
Break	Warm start EXMON.
Escape	Abort command entry or execution
N <cr>	Reset Basic pointers (NEW).
Q	Quit EXMON, and enter Basic
WS <byte>	Change EXMON workspace
H	Toggle printer on/off
*	Any operating system call
C <expression>	Calculate value of expressron
OB <3 bytes>	JSR OSBYTE
OW <3 bytes>	JSR OSWORD
WI <byte>	Set window height.
P <addr>	Set default panel.
P <cr>	Set panel to default to current PC.
! <byte>	Change currently selected ROM.
<Tab>	Switch format (disassembly/hex. dump)
<cr>	Return to default panel.
#	Decimal values and expressions follow.

2. Editor (hex, Ascii and Assembly)

E <addr>	Edit memory in current format.
<Escape>	Exit editor.
Hex. editor:	
<Tab>	Switch to line assembler.
<Copy>	Switch between hex and Ascii editing.
<cursor-up>	Move up one line
<cursor-down>	Move down one line.
<cursor-left>	Move back one byte.
<cursor-right>	Move forward one byte.
<Shift-cursor-up>	Move up one screenful.
<Shift-cursor-down>	Move down one screenful.
<Shift-cursor-left>	Move to top left.
<Shift-cursor-right>	Move to bottom right.
Line assembler:	
<Tab>	Switch to hex. editor.
<copy>	Restart display at current address.
<cursor-up>	Move back one byte.
<cursor-down>	Move forward one instruction.
<Return>	Move forward one byte.
<Shift-cursor-up>	Move back 16 instructions.
<Shift-cursor down>	Move forward one screenful.

3. Other Memory Editing and Manipulation commands.

D <range>	Disassemble memory.
L <range>	List memory.
K <range> <filename>	Disassemble and save to file.
SS <range> <string>	Search for Ascii string in given range.
SB <range> <bytes>	Search for string of hex. bytes
FS <rangeA> <string>	FiIl memory with Ascii string.
FB <rangeA> <bytes>	FiIl memory with hex. bytes.
M <rangeA> <addrA>	Move memory block.
V <rangeA> <addrA>	Verify that two memory blocks are the same.
R <rangeA> <addrA> <0 to 3 rangeAs>	Relocate memory block, not relocating specified sub-ranges, otherwise adjusting all addresses in the range.

4. Debugging & simulation

@ <addr>	Set program counter.
A <byte>	Set accumulator.
X <byte>	Set X register
Y <byte>	Set Y register
IP <byte>	Set Processor Status register
IS <byte>	Set stack pointer.
IC	Invert carry flag
IZ	Invert zero flag.
II	Invert interrupt flag.
ID	Invert decimal flag.
IB	Invert break flag.
IV	Invert overflow flag.
IN	Invert sign flag.
G <addr>	Execute program (GO).
J <addr>	JSR to <addr>.
<space-bar>	Enter simulation mode. If already simulating then step one instruction. When simulating, step one instruction, treating a JSR as a single instruction.
<Delete>	When simulating, skip next instruction
TB <addr>	Trace from <addr> until next breakpoint.
TSB <addr>	Same, suppressing trace output
TA <byte> <addr>	Trace until A= <byte> or breakpoint.
TSA <byte> <addr>	Same, suppressing trace output.
TX <byte> <addr>	Trace until X=<byte> or breakpoint
TSX <byte> <addr>	Same, suppressing trace output
TY <byte> <addr>	Trace until Y= <byte> or breakpoint.
TSY <byte> <addr>	Same, suppressing trace output
BS <addr>	Set breakpoint
BC <addr>	Clear breakpoint.
BW	Wipe all breakpoints.

5. Dual Screen (BBC only)

ZI <byte>	Initialise dualscreen.
ZC	Cancel dualscreen operation.
<CTRL-Z>	Change simulation display mode between CP; C and P.
<CTRL-Tab>	Display program screen. Escape to return to Control screen.

6. Syntax definitions

<addrA> means an address with default by pressing <cr> not allowed.
<rangeA> means a range with default not allowed on the second address.

<addr>::=<addrA><cr> (<cr> defaults to current PC address)
<addrA>::=<4 hex. digits><1-3 hex. digits><cr>#<expression>
<byte>::=<2 hex. digits><hex. digit><cr><cr>#<expression>
(<cr> defaults to zero, except &FF for 'S' command)
<range>::=<rangeA><addr><cr> (<cr> defaults to 'continuous listing')
<rangeA>::=<addr><addrA><addr>+<addr>
<cr>::=<RETURN>
<expression>::=general expression, involving +, -, *, /, ^, brackets, variable names (e.g. assembly source code labels).