```
200LDA&72:CLC:ADC#8:STA&72
210LDA&73:ADC#0:STA&73
215DEX:BNEbigloop
220LDA#129:LDY#255:LDX#&D6
230JSR&FFF4:CPX#&FF:BNEno
240CPY#&FF:BNEno
250LDA&72:CLC:ADC#8:STA&72
260LDA&73:ADC#0:STA&73
262LDA#129:LDY#255:LDX#&FF
264JSR&FFF4:CPX#&FF:BNEno
266CPY#&FF:BNEno
268LDA&72:CLC:ADC#176:STA&72
269LDA&73:ADC#0:STA&73
270JMPcont:.no
280LDA#129:LDY#255:LDX#&C6
290JSR&FFF4:CPX#&FF:BNEcont
```

```
300CPY#&FF:BNEcont
310LDA&72:SEC:SBC#8:STA&72
320LDA&73:SBC#0:STA&73
322LDA#129:LDY#255:LDX#&FF
324JSR&FFF4:CPX#&FF:BNEcont
326CPY#&FF:BNEcont
328LDA&72:SEC:SBC#176:STA&72
329LDA&73:SBC#0:STA&73
330.cont
340LDA#129:LDY#255:LDX#&8F
350JSR&FFF4:CPX#&FF:BNEnxt
360CPY#&FF:BNEnxt
370LDA#31:JSR&FFEE
380LDA#0:JSR&FFEE:LDA#24:JSR&FFEE
390BRK:EQUB27:EQUS"Escape":BRK
890.nxt
```

```
 900LDA&72:SEC:SBC#192:STA&72
 910LDA&73:SBC#0:STA&73
 920JMPgo
1000.prnt STY&75:LDY#0
1010STA(&70),Y:LDA&70:CLC
1020ADC#1:STA&70:LDA&71
1030ADC#0:STA&71:LDY&75:RTS
1040.hexsp JSRhex:LDA#32:JMPprnt
1050.hex PHA:AND#&F0
1060LSRA:LSRA:LSRA:LSRA:JSRohex
1070PLA:AND#&F
1080.ohex
1090CMP#10:BPLalph
1100CLC:ADC#48:JMPprnt
1110.alph CLC:ADC#55:JMPprnt
10000]:NEXT
```

PROGRAMMING: MEMOTECH

Recovery

A F Wilson

Recovery is a utility for the Memotech MTX series of micros which mimics the BBC micro command *Old*. Recovery, as its name suggests, recovers programs which have been accidentally erased. The utility is interrupt driven, thus the program is available at all times. Function key *F1* is used to save the system variables and *F2* to restore the saved system variables.

As you will soon see, the recovery listing is for the disc-based system, but by removing the *USER* command from line 120 of listing this allows the program to work on a tape-based MTX micro. Note MTX 500 owners should change all £4000 references to the equivalent £8000 addresses. The program, when loaded, auto runs itself, and relocates itself in high memory, then sets the interrupt vector and NEW's itself. The program won't affect any Basic programs and is available at all times, just press *F1* and *F2*.

To save the program, type *GOTO 120*. This will save the program. When you come back to reload the program, the Basic OS is ready to execute line 140. The *RUN* command sets the Basic OS to line 10, which goes to line 100. The code at line 100 then moves the relocatable code in line 20, to the top of free space, sets the interrupt vector to point to the relocated code. Once this is done the program NEWs itself, as we don't need the basic listing – just the code at £BF34.

To use the program, load it using *USER LOAD "RECOVERY.BAS"* for disc users, or *LOAD " "* for tape users. Type in the following example.

10 PRINT "TEST"
20 REM

Now press *F1* to save the current program. Type in *PANEL <RET>,D BF97 ,<BRK>*. The hex dump at the bottom of the screen displays the contents of *FIRST8B,SYSVARS*. These should be:

0C 00 0A 00 90 22 54 45 12 40 00 40 12 40 12 00 00 40 12 40 00

Type *NEW <RET>*, then press *F2* to recover.

However, if you press the RESET keys then you lose the interrupt code. This means you cannot use the function keys. At this stage you can either save the program you are working on and reload the *Recover* program or use the following two commands to save and restore.

To save	: RAND USR(48960)
To recover	: RAND USR(49012)

```
Listing :

10 GOTO 100
20 CODE

4010          LD A,(£FD7D)  ;LET A=THE LAST ASCII KEY PRESSED.
4013          CP £80        ;IS IT 'F1'
4015          JR Z,SAVEPRG  ;IF SO GOTO SAVE VARIABLES
4017          CP £81        ;IS IT 'F2'
4019          JR Z,RESTPRG  ;IF SO GOTO RESTORE VARIABLES
401B          RET           ;RETURN TO CALL ROUTINE
401C SAVEPRG:LD HL,£4000    ;POINT HL TO START OF BASIC
401F          LD DE,£BF97   ;SAVE AT FIRST8B
4022          LD BC,8       ;8 BYTES TO SAVE
4025          LDIR          ;SAVE THE 8 BYTE BLOCK
4027 SARLOOP:LD DE,£BF9F    ;POINT DE TO SYSVARS
402A          LD IX,£BF90   ;POINT  IX TO LSB  OF SYSTEM VARS.
402E          LD B,7        ;SEVEN WORDS TO SAVE.
4030 SYSVLOP:LD A,(£BFAD)   ;LET A=SAVE OR RESTORE FLAG.
4033          LD H,£FA      ;H=MSB OF SYSTEM VARIABLE
4035          LD L,(IX+0)   ;L=LSB OF SYSTEM VARIABLE
4038          PUSH DE       ;SAVE DE AS AFFECTED ON RESTORE
4039          CP 0          ;IS FLAG=0?
403B          JR Z,SAVE     ;IF SO SAVE SYSTEM VARIABLES
403D          CALL £BF87    ;RESTORE SYSTEM VARIABLES
4040          JR LOADED     ;ONCE RESTORED WORD BYPASS SAVE.
4042 SAVE:    CALL £BF88    ;SAVES SYSTEM VARIABLES
4045 LOADED:  POP DE        ;RESTORE CORRUPTED REGISTER PAIR
```

```
4046          INC DE        ;MOVE DE ONTO 2 PLACES
4047          INC DE        ;
4048          INC IX        ;MOVE IX TO NEXT WORD IN VARDISP
404A          DJNZ SYSVLOP  ;IS ALL FINISHED
404C          LD (£BFAD),A  ;RESET FLAG
404F          RET           ;RETURN TO CALLING ROUTINE
4050 RESTPRG:LD HL,£BF97    ;RESTORE FIRST8B AT £4000 ON A MTX
4053          LD DE,£4000   ;512 COMPUTER,£8000  ON A MTX 500.
4056          LD BC,8       ;8 BYTES TO BE RECOVERED
4059          LDIR          ;MOVE THEM BACK.
405B          LD A,1        ;SET FLAG FOR RESTORE
405D          LD (£BFAD),A  ;SAVE IT IN FLAG
4060          JP £BF4B      ;GOTO SARLOOP
4063 RESSYSV:EX DE,HL       ;LET DE=HL & HL=DE ,FOR RESTORING.
4064 SAVSYSV:LD A,(HL)      ;SWAP CONTENTS OF DE AND HL.
4065          LD (DE),A     ;
4066          INC HL        ;REMEMBER SWAPPING TWO BYTES
4067          INC DE        ;
4068          LD A,(HL)     ;
4069          LD (DE),A     ;
406A          XOR A         ;CLEAR REGISTER A AND FLAGS
406B          RET           ;RETURN TO MAIN PART
406C VARDISP:DB £A4,£A7,£AA,£AC,£CC,£CF,£D6
4073 FIRST8B:DS 8
407B SYSVAR: DS 14
4089 FLAG:    DB 0
408A          RET
```

```
100 CODE

420C        LD HL,£4010   ;MOVE THE CODE IN LINE 20 TO
420F        LD DE,£BF34   ;TOP OF FREE SPACE.
4212        LD BC,123     ;JUST  BELOW  THE  BASIC  VARIABLE
4215        LDIR          ;RAM AT £C000 TO £D6FF.
4217        LD A,£C3      ;LET INTERRUPT VECTOR TO JP £BF34
4219        LD (£FA98),A  ;THE JP £BF34 IS STORED IN A
421C        LD HL,£BF34   ;RESERVED 3 BYTE SPACE AT £FA98.
421F        LD (£FA99),HL ;THIS IS CALLED EVERY 64TH OF A
4222        LD A,(£FD5E)  ;SECOND WHEN INTFFF,ADDRESS IS
4225        OR £9F        ;SET BY SETTING BITS 4 AND 7,SEE
4227        LD (£FD5E),A  ;BASIC REFERENCE MANUAL.
422A        RET           ;RETURN TO BASIC

110 NEW
120 USER SAVE "RECOVERY.BAS" :REM FOR DISC USERS
130 REM USE SAVE "RECOVERY" FOR TAPE USERS.
140 RUN
```

Micromon

P A Fairclough

Here's part three of Micromon, the multi-function machine code programming utility.

M – Memory.
Format : *M addr* or
M addr, addr

Memory will convert memory into hexadecimal bytes. Any value may be changed by overtyping the original and pressing Return.

"Micromon is a multi-function machine code programming utility for the Commodore 64"

N – Number.
Format : *N addr,addr,offset,addr,addr* or
N addr,addr,offset,addr,addr,W

Number allows all absolute addresses in a machine code program to be changed. If *W* is specified then the code is assumed to be a word table consisting of an iteration of addresses in low byte/high byte format.

The first two addresses specify the block code to be numbered. The last two addresses specify the old block of the code. Offset is a hexadecimal value indicating how much is to be added to each absolute address to make the addressing correct.

O – Out.
Format : *O value* or
O value,value

Out will tell the monitor how the *Roms* are set before memory access. The command may consist of one or two hexadecimal values.

The first data value used is to show how the memory is set up before access. Only the first 3 bits are used and have the same function as location $01. The images seen are:

Value	I/O ($D000)	Kernal ($E000)	Basic ($A000)
00	Ram	Ram	Ram
01	I/O	Ram	Ram
02	Chr	Ram	Ram
03	Chr	Ram	Ram
04	Ram	Ram	Ram
05	I/O	Ram	Ram
06	I/O	Ram	Ram
07	I/O	Ram	Ram

I/O are the VIC, SID and CIA chips. Chr is the Character Rom.

The second value is used by the *G*, *Q* and *W* commands to show how the Basic Rom (at $A000) is always set. Only the first bit is used. The first value has priority over this one. The byte has the following function:

Value	Basic ($A000)
00	Ram
01	Rom

P – Print.
Format : *P value data*

Print allows the user to send data bytes to the printer. The printer must have been opened by using the relevant function key.

The value tells the monitor what Ascii code to send along with a carriage return. Data may consist of one or more hexadecimal bytes.

Q – Quick.
Format : *Q or Q addr*

Quick runs a machine code program starting at the PC or the address. Each instruction is checked to see if a breakpoint should occur. Pressing the Stop key will display the registers. Program execution will be passed to the *W* command if a breakpoint occurs.

R – Registers.
Format : *R*

Registers will display the current register values of the 64. They are:

PC – Program Counter
SR – Status Register
N – Negative Sign Bit
V – Overflow Bit
– – Unused Bit
B – Break Bit
D – Decimal Bit
I – Interrupt Bit
Z – Zero Bit
C – Carry Bit
AC – Accumulator
XR – X Register
YR – Y Register
SP – Stack Pointer

Any of the values may be changed by typing over the old value and pressing *Return*.

S – Save.
Format : *S "filename",device,addr,addr ,sec* or
S @addr,device,addr,addr,sec

Save will store a block of memory as a file on a device.

The filename must be enclosed in quotation marks. If the filename supplied is an '@' sign with an address then the filename will be taken from the 187 bytes commencing at the address.

The device must be 01 for cassette, or 08 for diskette. The secondary address must be 00 for a relocatable file, or 01 for an unrelocatable file.