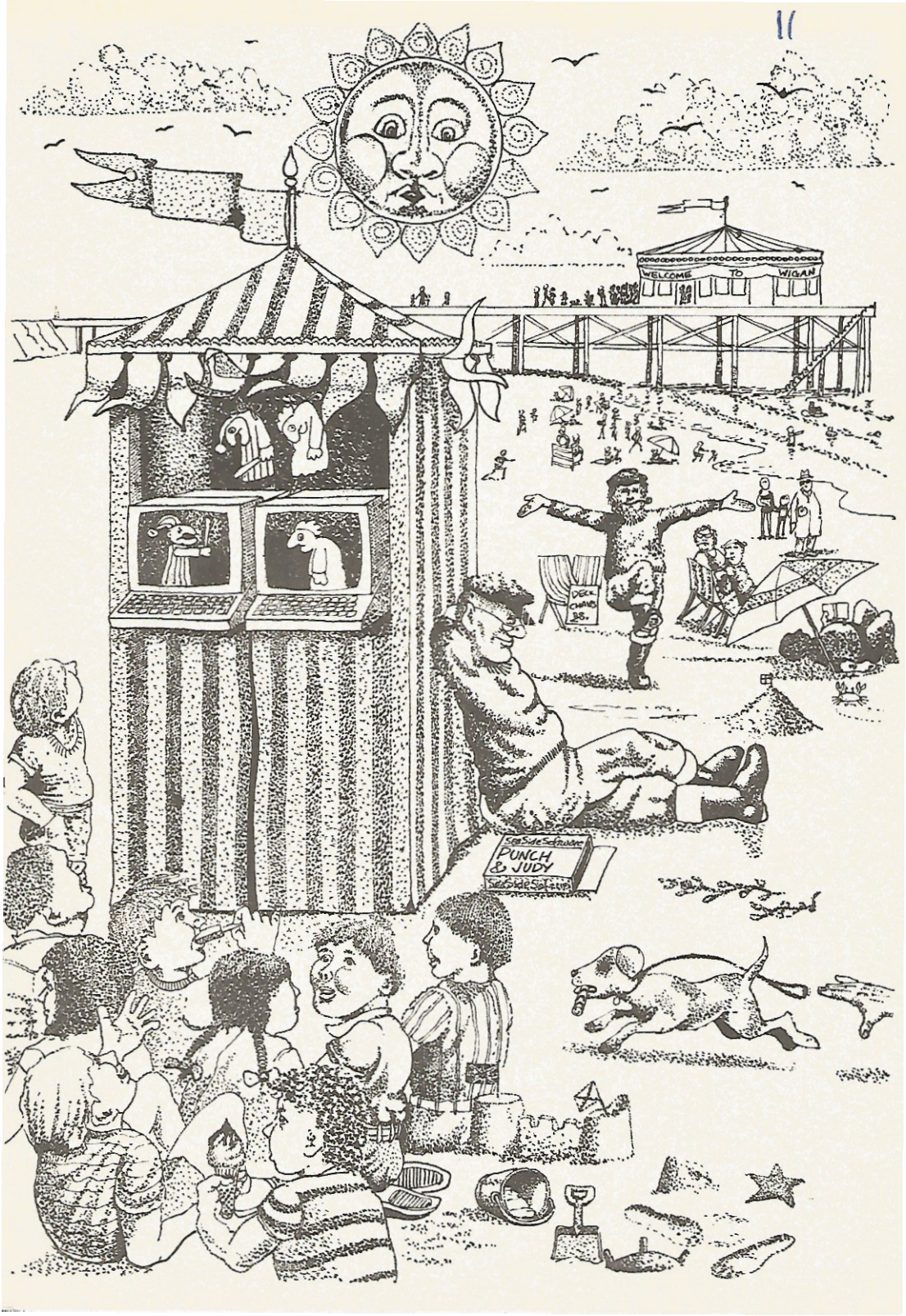

WELCOME TO WIGAN
DECK CHAIRS 25.
PUNCH & JUDY

Issue No.11

C O N T E N T S

SHARPSOFT USER NOTES

Issue No. 11

From your letters, telephone calls etc. we believe that our proposal to publish six issues of these User Notes each year is a step in the right direction to providing SHARP computer owners with an up-to-date information service. We expect that in the future, feedback to readers' letters and comments will be more immediate. The increased number of issues per year does however, rely on a continuing supply of articles, programs and letters from our readers. In each issue we attempt to present a balanced coverage of the various aspects of SHARP personal computing. However, without your contribution it is not always possible for us to include in each issue all our regular, and popular, columns. For example, in this issue we have had to leave out the FORTH and CP/M columns due to lack of material. Looking ahead to future issues we would appreciate contributions on the following topics; MZ-80A hardware and software, PASCAL, FORTH, CP/M and programs for the MZ-80K, A and B in any computer language.

The theme of Issue 11 is "applications". Since I started editing these User Notes I have never ceased to be amazed at the ingenuity of our readers who develop software or modify their computer's electronics. In this issue you will find a wide selection of interesting articles, including a program for enhanced printing on an EPSON MX 80 Printer from B. Gold, modifying the MZ-80K V.D.U. electronics from Alan Vale and Part 2 of adding disk drives to the MZ-80K by Peter Sydenham.

This issue also includes the second part of our "Beginners Guide to PASCAL" and a complete HISOFT PASCAL program called DOUBLE DUTCH. This program is a contribution from Alan Stevens. If you have any similar contribution please send it to us for inclusion in a later issue.

Issue 12 of the User Notes will be published towards the end of May 1984.

MIKE BRINSON

EDITOR

BEGINNERS GUIDE TO PASCAL

PART 2 - Expressions and Statements

16. Operator precedence.

```
()
NOT
*  /  DIV  MOD  AND
+  -  OR
<  <=  <>  =  >=  >
```

17 Predefined functions

Functions return a single value

1×1	ABS(x)
x^2	SQR(x)
$\sqrt{x}$	SQRT(X)
e^x	EXP(x)
lnx	LN(x)
SINX	SIN(x)
COSX	COS(x)
$\tan^{-1}x$	ARCTAN(X)

18 Program Body

```
PROGRAM name;

BEGIN
  statement1;        )
  statement2;        )   BODY
  statement3;        )
      .
      .
      .
END.
```

19. PASCAL statements

1. Assignment statement
2. Sequence statements
   (a) Compound
   (b) GOTO
3. Decision statements
   (a) IF statement
   (b) CASE statement
4. Loop statements
   (a) REPEAT stateent
   (b) WHILE statement
   (c) FOR statement
5. Empty statement
6. WITH statement
7. PROCEDURE and FUNCTION statements.

20. Assignment statement

```
variable   :=  expressions;
```

A variable may only be assigned a value that is "ASSIGNMENT COMPATIBLE"

Y:=X;

Y ↓ / X →	INTEGER	REAL	BOOLEAN	CHAR
INTEGER	✓			
REAL	✓	✓		
BOOLEAN			✓	
CHAR				✓

21. Compound Statement

```
BEGIN
  statement1;      )
  statement2;      )
     .             )Executed in order
     .             )statement 1 - statement n
     .             )
  statement n      )
END
```

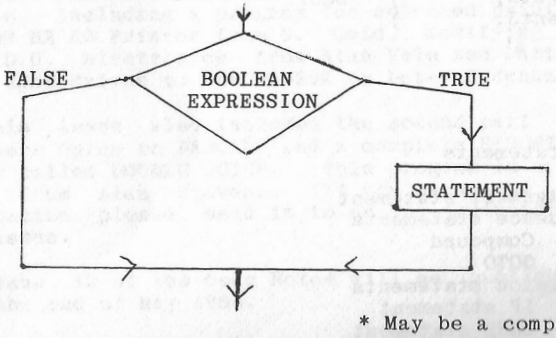
Statements between BEGIN and END are treated as one statement.

22. GOTO - see later notes.

23. IF statement.

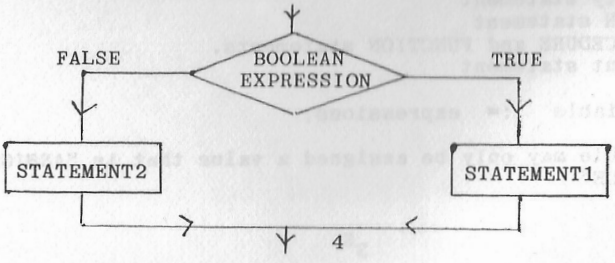
Two forms.

(a) IF <BOOLEAN expression> THEN statement*;

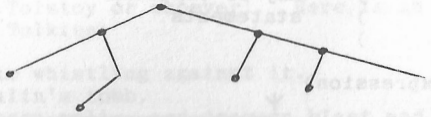


* May be a compound statement

(b) IF <BOOLEAN expression> THEN statement 1* ELSE statement2*;



Nested IF's - 1. Binary decision
2. Compare many types.
3. Reflect tree-type decision.

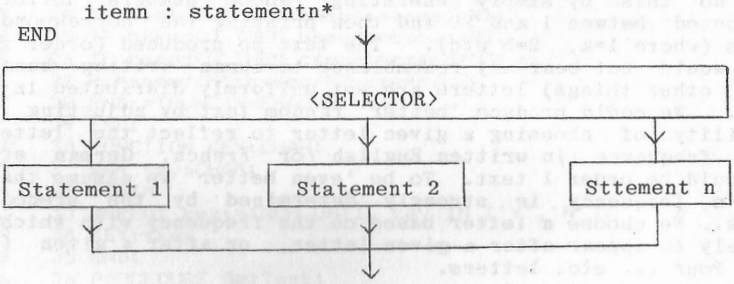


24. IF example program - find the values of L and M at the endof the program.

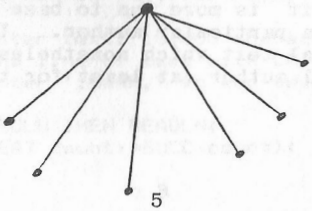
```
PROGRAM LM;
CONST x=1; Y=1; Z=2;
VAR   L,M : INTEGER;
BEGIN
  IF x>=Z THEN
   BEGIN
     L:=0; M:=1
   END
  ELSE IF NOT (Y>Z) THEN
         BEGIN
            L:=2; M:=1
         END
       ELSE M:=4
END.
```

25. CASE statement.

```
CASE <selector> OF

     item1 : statement 1*;          * may be compound
     item2 : statement 2*;            statements
           .
           .
           .
     itemn : statementn*
END
```

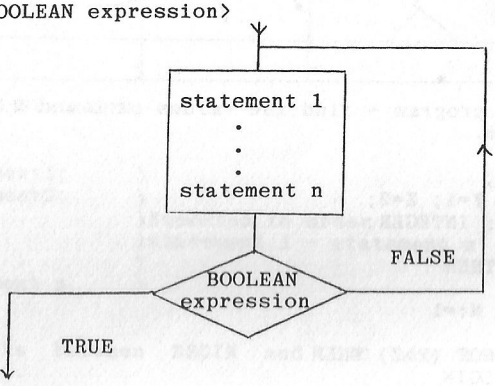


CASE --- 1. Matches ACTION to <Selector>.
2. The <Selector> MUST be an Ordinal type.
3. The choices (ie "items") must be a constant.



26. REPEAT statement.

```
REPEAT
  statement1;          )    may be compound
      .                )    statements
      .                )
  statementn
UNTIL <BOOLEAN expression>
```



Continued next Issue

DOUBLEDUTCH

A Program in PASCAL by Alan Stevens

'DoubleDutch' is a little nonsense written in Hisoft Pascal for the MZ-80K. It was inspired (if that's the right word!) by an article in Scientific American (November 1983), Computer Recreations by Brian Hayes).

The idea is for the program to produce random text. It could do this by simply generating random numbers uniformly distributed betwen 1 and 26 and then printing the corresponding letters (where 1=a, 2=b etc). The text so produced (order zero text) would not bear any resemblance to human writing because (among other things) letters are not uniformly disributed in the latter. We could produce 'better' random text by adjusting the probability of choosing a given letter to reflect the letter's actual frequence in written English (or French, German etc.) This would be order 1 text. To be 'even better' we assume that a letter's frequency is strongly determined by the preceding letters. We choose a letter based on the frequency with which it is likely to appear after a given letter, or after a given two, three, four ... etc. letters.

The letter frequencies for a given 'order' could be obtained by anyalysing an enormous amount of English text and then building the resulting average frequencies into the text generating program. However, it is more fun to base the frequencies on a sample of text from a particular author. In this way one can generate a nonsensical text which nonetheless retains the 'flavour' of the original author (at least for third or fourth order text and above).

By entering some source text when asked to by program DoubleDutch and specifying the 'order' (1 to 9) you too can produce reams of meaningless gibberish in the style of Shakespeare, Tolstoy or whoever! Here is an example of fourth-order, random Tolkien:

Arrows cam whistling against it.
It crash on Balin's tomb.
Boromir anothern wall, and Aragorn blast and hammers was a horn slew many the opening against it.
It cracked and Aragorn blast and hammers were dismayed by the nothern wall, and Aragorn slew many.
When thirteen had sprung up on Balin's tomb.

For a more thorough discussion of the topic see the Scientific American article quoted above. Have fun!

```
6CF6     2 PROGRAM DoubleDutch;
6CF6     3 CONST max=10000;
6CF6     4       null=CHR(0);
6CF6     5       crlf=CHR(13);
6CF6     6       toggle=CHR(16);
6CF6     7       hash=CHR(35);
6CF6     8 VAR text : ARRAY[1..max] OF CHAR;
6CFF     9     part, string : ARRAY[1..9] OF CHAR;
6CFF    10     posn, order, count, line, linmax,
6CFF    11     aseed, bseed, cseed : INTEGER;
6CFF    12     symbol, rpt, hdcopy : CHAR;
6CFF    13     nomore : BOOLEAN;
6CFF    14 FUNCTION RND(N:INTEGER):INTEGER;
6D02    15 CONST N1=30269; N2=30307; N3=30323;
6D02    16       M1=177; M2=176; M3=178;
6D02    17 VAR R : REAL;
6D02    18 BEGIN
6D1A    19 IF N<0 THEN BEGIN aseed:=5271; bseed:=934; cseed:=25859;
6D43    20                   N:=-N END;
6D52    21 aseed:=171*(aseed MOD M1)-2*(aseed DIV M1);
6D83    22 bseed:=172*(bseed MOD M2)-35*(bseed DIV M2);
6DB4    23 cseed:=170*(cseed MOD M3)-63*(cseed DIV M3);
6DE5    24 IF aseed<0 THEN aseed:=aseed+N1;
6E08    25 IF bseed<0 THEN bseed:=bseed+N2;
6E2B    26 IF cseed<0 THEN cseed:=cseed+N3;
6E4E    27 R:=aseed/N1+bseed/N2+cseed/N3;
6E97    28 RND:=TRUNC(N*FRAC(R))+1
6EBC    29 END;
6EC7    30 FUNCTION KEY:CHAR;
6ECA    31 VAR key:CHAR;
6ECA    32 BEGIN
6EE2    33 REPEAT key:=INCH UNTIL key IN ['Y','N'];
6F19    34 KEY:=key
6F19    35 END;
6F26    36 PROCEDURE GetText;
6F29    37 VAR I : INTEGER;
6F29    38     stop : BOOLEAN;
6F29    39 BEGIN
6F41    40 FOR I:=1 TO max DO text[I]:=null;
6F8A    41 PAGE;
6F8F    42 WRITELN('Enter text. 10,000 characters maximum.');
6FC3    43 WRITELN('End line of text with CR');
6FE9    44 WRITELN('Enter ',hash,' to end text.');
701A    45 count:=0;
7020    46 REPEAT IF EOLN THEN READLN;
702D    47        REPEAT count:=SUCC(count);
```

```
7037   48                  READ(text[count]);
7056   49                  stop:=(count=max-1) OR (text[count]=hash)
708B   50          UNTIL EOLN OR stop;
709C   51          IF text[count]<>hash THEN count:=SUCC(count);
70CA   52          text[count]:=crlf
70E7   53 UNTIL stop
70E8   54 END;
70F7   55 PROCEDURE Xtract(posn:INTEGER);
70FA   56 VAR I : INTEGER;
70FA   57 BEGIN
7112   58 FOR I:=1 TO order DO part[I]:=text[posn+I-1]
7184   59 END;
7192   60 FUNCTION Letter(posn:INTEGER):CHAR;
7195   61 VAR match :BOOLEAN;
7195   62     loop : INTEGER;
7195   63 BEGIN
71AD   64 loop:=0;
71B6   65 REPEAT IF posn>=count-order-1 THEN posn:=1;
71E8   66          Xtract(posn);
71F8   67          loop:=SUCC(loop);
7205   68          match:=(part=string);
721E   69          posn:=SUCC(posn);
722B   70          IF loop=count THEN match:=TRUE
7243   71 UNTIL match;
724C   72 Letter:=text[posn+order-1]
7275   73 END;
7281   74 PROCEDURE Update(l:CHAR);
7284   75 VAR I : INTEGER;
7284   76 BEGIN
729C   77 IF order>1 THEN FOR I:=1 TO order-1 DO string[I]:=string[I+1];
7319   78 string[order]:=l
7332   79 END;
733F   80 PROCEDURE Start;
7342   81 VAR I,R : INTEGER;
7342   82 BEGIN FOR I:=1 TO 9 DO part[I]:=null;
73A2   83 string:=part;
73AF   84 R:=RND(count-order);
73D0   85 WHILE text[R]<>crlf DO R:=SUCC(R);
740D   86 IF R=count THEN R:=0;
742B   87 FOR I:=1 TO order DO BEGIN
7455   88                        string[I]:=text[I+R];
749E   89                        WRITE(string[I])
74BD   90                        END;
74C1   91 line:=0
74C5   92 END;
74CF   93 FUNCTION Linecheck(l:CHAR):BOOLEAN;
74D2   94 BEGIN
74EA   95 IF l=crlf THEN line:=SUCC(line);
7502   96 Linecheck:=(line=linmax)
7511   97 END;
751A   98 BEGIN (* Main program *)
7523   99 posn:=RND(-1); (* Initialise seeds *)
7534  100 GetText;
7539  101 REPEAT PAGE;
7541  102        WRITELN('What order? (1 to 9)');
7563  103        REPEAT READ(order) UNTIL order IN [1..9];
75A4  104        WRITELN('How many lines of output?');
75CB  105        READ(linmax);
75D1  106        WRITELN('Hardcopy output? (Y/N)');
75F5  107        hdcopy:=KEY;
7600  108        IF hdcopy='Y' THEN WRITE(toggle);
```

```
7616  109          Start;
761B  110          REPEAT posn:=RND(count-order);
7638  111                 symbol:=Letter(posn);
7647  112                 WRITE(symbol);
764D  113                 Update(symbol);
7657  114                 nomore:=Linecheck(symbol)
765D  115          UNTIL nomore;
766E  116          WRITELN;
7671  117          IF hdcopy='Y' THEN WRITE(toggle);
7687  118          WRITELN('Another go with same text? (Y/N)');
76B5  119          rpt:=KEY
76B6  120 UNTIL rpt='N'
76C7  121 END.
End Address: 76D3
```

LOW COST VIDEO MODIFICATION FOR THE MZ-80K

by

ALAN VALE

I have been using my MZ-80K with this modification for nearly a year with no ill effects to the computer. This modification uses unassigned addresses under software control. Any standard configurations i.e. F/disc & printer etc. can be used with this modification.

NOTE: IF YOU DO NOT HAVE A SUITABLE BACKGROUND IN ELECTRONICS DO NOT ATTEMPT TO DO THIS MODIFICATION YOURSELF.

This mod takes about 1.5 hours if you have the correct tools and equipment.

Equipment Required

Soldering Iron with a small bit and solder
1 metre PCB wire (I used wire wrap wire which you can buy from places such as Tandy or RS Components (Approx cost #1.50)

Suitable:
Screwdrivers
Wire Cutters
Wire Strippers
Long Nose Pliers

One of the most annoying features of the standard MZ-80K is the video display, it's the complete inverse of the printer (i.e. dark characters on a light background).

Having previously desinged a VDU controller for a particular application I decided to apply some of my experience to the MZ-80K. After going through the circuit diagrams of the MZ-80K manual I found that there are quite a few spare IC (integrated ciruit) gates on the main PCB of the MZ-80K.

To control the VDU I found some chip select positions which are unused at hexadecimal addressed E00C and E01C (Note there are several others).

IC 30 which is a 74LS42 provides address E00C on pin 4 & E01C on pin 9.

There is a spare flip flop on IC2 (74LS74)
There is a spare exclusive or gate IC26(74LS86).

The Modification

1. Switch your computer and printer etc. off.

2. Disconnect from the mains.

3. A copy of the service manual is helpful but not essential.

4. Unscrew the top of the computer which hinges up and can be clipped open by a catch on the left hand side.

5. Carefully remove the various PCB plugs from main PCB only.

6. Unscrew and unclip the 6 PCB mountings.

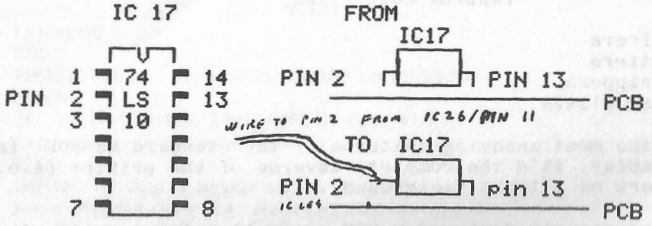
7. You should now be able to CAREFULLY remove the PCB, you should not have to force it out else 5, 6 & 7.

8. The video signal is output from a shift register IC29 (74LS165) pin 9. Pin 9 of IC29 is connected to IC17 (74LS10) pin 2.

This connection IC29 PIN 9 & IC17 PIN 2 must be broken.
Although I cut the track between the Plated through hole on the component side of the PCB AND PIN 2 OF IC17, I advise you to do an alternative as if you cut the wrong track on the PCB repairs may be costly.

The alternative is carefully cut pin 2 of IC17 (74LS10) as close to the PCB as possible (On the component side).
Lever the IC Pin up until it is nearly parallel with the PCB without breaking the pin or damaging the PCB.

See diagram below:-



9. Wiring List

TAKE CARE NOT TO SPLASH SOLDER BETWEEN TRACKS ON THE PCB.

CAREFULLY solder the following wire links:-

FROM			TO		
IC30	PIN	4	IC2	PIN 13	!ADDRESS E00C HEX
IC30	"	9	IC2	" 10	!ADDRESS E01C HEX
IC2	"	11	IC46	" 26	!Z80 RESET TO CLOCK INPUT OF IC2.
IC2	"	12	IC2	" 14	!D INPUT TO 5 VOLT RAIL (SEE NOTE 15)
IC2	"	9	IC26	" 13	!O/PUT OF LS74 TO XOR GATE INPUT OF IC26
IC29	"	9	IC26	" 12	! " LS165 TO XOR " " " "
IC26	"	11	IC17	" 2	! " LS86 TO 3 INPUT NAND GATE
					!PIN 2 WHICH YOU SHOULD HAVE LIFTED
					!SEE SECTION 8.

CHECK YOUR WIRING CAREFULLY AS ANY ERROR MAY TURN OUT TO BE COSTLY. WIRING SHOULD BE DONE ON THE SOLDER SIDE OF THE PCB EXCEPT FOR THE CONNECTION BETWEN IC26 PIN 11 & IC17/2 WHICH SHOULD BE DONE ON THE COMPONENT SIDE IF YU HAVE CHOSEN THE OPTION OF LIFTING PIN 2 OF IC17 IN SECTION 8.

10. FIT THE PCB BACK INTO THE COMPUTER AND CONNECT ALL THE PCB CONNECTORS WHICH YOU REMOVED.

11. CLOSE THE LID OF THE COMPUTER.

12. YOUR COMPUTER SHOULD NOW BE READY FOR USE.

13. IF NOTHING HAPPENS AFTER SWITCH ON RECHECK YOUR MODIFICATION

14. Software Selection

ON POWER UP THE SCREEN WILL USUALLY POWER UP IN THE REVERSE VIDEO CONDITION. THE NORMAL OR REVERSE VIDEO OPTION CAN BE SELECTED FROM MACHINE CODE BY VARIOUS COMMANDS SUCH AS:

```
CD 0C E0 & CD 1C E0
32 0C E0 & 32 1C E0
```

IN BASIC
POKE 57356,0 FOR SELECTION OF NORMAL VIDEO.
POKE 57372,0 FOR SELECTION OF REVERSE VIDEO.

15. Additional Hardware Selection

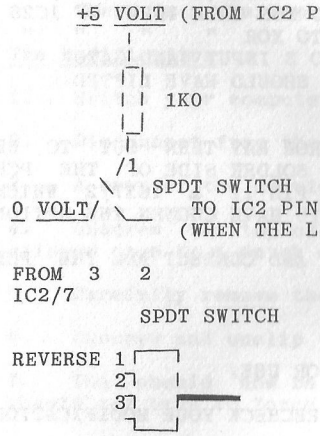
IF YOU HAVE A PUSH BUTTON RESET SWITCH ON YOUR COMPUTER (INSTALLATION OF WHICH WAS DESCRIBED IN AN EARLIER EDITION OF USER NOTES), ANOTHER SWITCH MAY BE FITTED. THIS SWITCH MUST BE A TOGGLE TYPE SWITCH 'SPDT'.

THEN IN PLACE OF THE CONNECTION BETWEEN IC2/12 & IC2/14, THE LINK BETWEEN IC2 PIN 12 & IC2 PIN 14 MAY BE REMOVED AND A WIRE CONNECTED BETWEEN:-

HAVING MOUNTED THE SPDT SWITCH

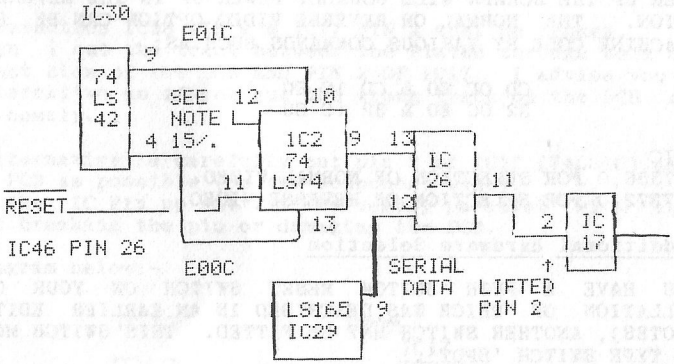
FROM	TO
IC2 PIN 12	CENTER (LEVER) OF THE SWITCH.
IC2 PIN 8	ONE OF THE REMAINING CONTACTS ON THE SWITCH.
1000 OHM RESISTOR (BROWN BLACK RED)	ONE LEG OF THE RESISTOR TO PIN 14 OF IC2.

THE OTHER SIDE OF THE RESISTOR TO THE REMAINING CONNECTION OF THE SWITCH.



FOR HARDWARE SELECT
THE REQUIRED SWITCH POSITION &
PUSH THE RESET BUTTON.

CIRCUIT DIAGRAM FOR THE NORMAL & REVERSE VIDEO MOD.



Last of all you will most probably have to alter your brightness & contrast to get the optimum balance between normal and reverse video.

I hold no responsibility for any damage to any computers 'or the users' while implementing or using this mod. You do it at your own risk.

Uses of Reverse Video

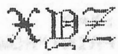
As well as the advantages when reading text and program listings etc. some very interesting results can be obtained with graphics when alternating between normal & reverse video when simulating explosions etc. on various games.

I have several other Hardware improvements for the MZ-80K but they are for later issues, if you are interested please let me know as soon as possible.

NOTE
If you cannot do this mod yourself I am prepared to modify any MZ-80K for twenty pounds plus carriage costs (i.e. Securicor etc.) Telephone 0273 563212.

BIT PRINTING TO EPSON MX 80

by
B. GOLD



The program is written in DISK BASIC, and is fairly self explanatory. First plot the design that is required to be printed, on a graph. Then rule the graph into lines of eight rows, read the lines upward putting 1 to print and 0 for a space, for each of the lines from left to right, this will produce binary numbers in reverse, which the program will change into decimal numbers to poke into memory.

Illustration letter 'B' double size.

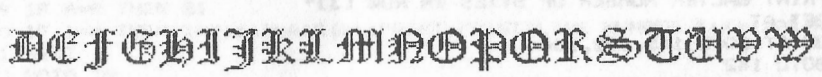
```
B. GOLD
|1 |1 |1 |1 |1 |1 |1 |1 |1 |0 |
|1 |1 |1 |1 |1 |1 |1 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |1 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |0 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |0 |0 |1 |1 |
|1 |1 |0 |0 |0 |0 |0 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |1 |1 |1 |0 |
|1 |1 |1 |1 |1 |1 |1 |1 |0 |0 |
--------------------------------
|1 |1 |1 |1 |1 |1 |1 |1 |0 |0 |
|1 |1 |0 |0 |0 |0 |1 |1 |1 |0 |
|1 |1 |0 |0 |0 |0 |0 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |0 |0 |1 |1 |
|1 |1 |0 |0 |0 |0 |0 |1 |1 |1 |
|1 |1 |0 |0 |0 |0 |1 |1 |1 |1 |
|1 |1 |1 |1 |1 |1 |1 |1 |1 |1 |
|1 |1 |1 |1 |1 |1 |1 |1 |1 |0 |
--------------------------------
```

```
1 REM BIT PRINTING TO EPSON MX 80
2 REM BY B.GOLD JANUARY 1984
3 LIMIT 53182
4 REM CONTROL ROUTINE
5 RESTORE
6 DATA 62,0,205,216,207
7 DATA 58,192,207
8 DATA 211,255,62,128
9 DATA 211,254,62,1
10 DATA 205,216,207
11 DATA 175,211,254,201
12 DATA 213,87,219,254,230,13,186,32,249
13 DATA 209,201,0,0,0
14 PR=53185:DT=PR-1
15 FOR I=PR TO PR+35
16 READ W:POKE I,W
17 NEXT
18 PRINT"■":CURSOR15,10:PRINT"MENUE"
19 PRINTTAB(15);"―――――■"
20 PRINTTAB(5);"[1] INPUT NEW LAYOUT"
21 PRINTTAB(5);"■[2] SEND BUFFER TO PRINTER"
22 PRINTTAB(5);"■[3] SAVE LAYFOUT TO DISK"
23 PRINTTAB(5);"■[4] GET LAYOUT FROM DISK"
24 PRINTTAB(5);"■[5] ALTERATION TO BUFFER"
25 PRINTTAB(5);"■[6] FINISHED WITH PROGRAM"
26 GET M:IF M=0 THEN 26
27 ON M-1 GOTO 83,105,120,138,195
28 PRINT"■■■■INPUT NEW LAYOUT"
29 PRINT"■■―――――――――――――――――――■"
30 PRINT"■■DENSITY -- NORMAL [1] OR DOUBLE [2]";:INPUT DN
31 IF DN=1 THEN DD=480:DA=75:GOTO 34
32 IF DN=2 THEN DD=960:DA=76:GOTO 34
33 GOTO 30
34 PRINT"■■NUMBER OF ROWS";:INPUT R3
35 DIM R(R3-1),F(R3-1):R=0
36 FOR I=0 TO R3-1
37 PRINT"■■■NUMBER OF BYTES IN ROW";I+1;:INPUT B
38 IF B>DD THEN PRINT"TOO MANY ":GOTO 37
39 R(I)=B:R=R+B
40 PRINT"■■■IS LINE FEED REQUIRED  1 = YES"
```

```
41 PRINT"                    0 = NO";:INPUT F
42 IF(F<>1)*(F<>0)THEN PRINT"■■■":GOTO40
43 F(I)=F
44 NEXT I
45 LM=53180-R:LIMIT LM-1
46 FOR I=LM TO 53180:POKE I,0:NEXT I
47 REM MAIN LOOP***************
48 PRINT"■INPUT BITS NOW   1 = 1"
49 PRINT"                  2 = 0"
50 PRINT"                  3 = 00000000"
51 PRINT"                  4 = 11111111":PRINT
52 A=R(0):X=0
53 PRINT:PRINT"ROW NUMBER 1":PRINT
54 FOR I=1 TO R:A$="":P$=""
55 IF I>A-R(X) THEN PRINTI-A+R(X);" ";:GOTO 57
56 PRINTI;" ";
57 FOR J=1 TO 8
58 GET P$:IF P$=""THEN 58
59 IF(P$="1")+(P$="2")+(P$="3")+(P$="→")+(P$="4")THEN 61
60 GOTO 58
61 IF P$="2"THEN P$="0"
62 IF(P$="3")*(J=1)THEN A$="00000000":PRINTA$;:R1=0:GOTO 77
63 IF(P$="4")*(J=1)THEN A$="11111111":PRINTA$;:R1=255:GOTO 77
64 IF(P$="3")+(P$="4")THEN 58
65 IF(P$="→")*(J=1)THEN I=I-1:PRINT"■■■■";SPC(15):PRINT"■"; :GOTO 56
66 IF P$="→"THEN J=J-1:PRINT"■ ■";:GOTO 58
67 A$=A$+P$:PRINTP$;
68 NEXT J
69 GOSUB 70:GOTO 77
70 R1=0:L=8:J=1
71 FOR K=1 TO L
72 IF K<>1 THEN J=J*2
73 R2=VAL(MID$(A$,K,1))
74 R1=R1+R2*J
75 NEXT K
76 RETURN
77 PRINT R1
78 POKE LM+I,R1
79 IF A=R THEN 81
80 IF I=A THEN X=X+1:A=A+R(X):PRINT:PRINT"LINE NUMBER ";X+1:PRINT
81 NEXT I
82 GOTO 18
83 POKE DT,27:USR(PR)
84 POKE DT,65:USR(PR)
85 POKE DT,8:USR(PR)
86 V=0:W=0
87 FOR L=0 TO R3-1
88 X=0:Y=0
89 IF R(L)<256 THEN X=R(L):Y=0:GOTO 93
90 X=R(L)
91 X=X-256:Y=Y+1
92 IF X>256 THEN 91
93 POKE DT,27:USR(PR)
94 POKE DT,DA:USR(PR)
95 POKE DT,X:USR(PR)
96 POKE DT,Y:USR(PR)
97 V=W+1:W=W+R(L)
98 FOR P=V TO W
99 POKE DT,PEEK(LM+P):USR(PR)
100 NEXT P
101 IF F(L)=0 THEN 103
102 POKE DT,13:USR(PR)
103 NEXT L
104 GOTO 18
```

```
105 PRINT"■":CURSOR 5,10:PRINT"SAVE LAYOUT TO SCREEN"
106 PRINTTAB(5);"──────────────────────"
107 INPUT"■NAME FOR FILE ";N$
108 WOPEN #1,FD1,N$
109 PRINT #1,R3,DA
110 FOR I=0 TO R3-1
111 PRINT #1,R(I),F(I)
112 NEXT I
113 PRINT #1,R
114 LM=53180-R
115 FOR I=LM TO 53180
116 PRINT #1,PEEK(I)
117 NEXT I
118 CLOSE #1
119 GOTO 18
120 PRINT"■":CURSOR 5,10:PRINT"INPUT LAYOUT FROM SCREEN"
121 PRINTTAB(5);"──────────────────────"
122 INPUT"■NAME OF FILE ";N$
123 ROPEN #1,FD1,N$
124 INPUT #1,R3,DA
125 DIM R(R3-1),F(R3-1)
126 FOR I=0 TO R3-1
127 INPUT #1,R(I),F(I)
128 NEXT I
129 INPUT #1,R
130 LM=53180-R:LIMIT LM-1
131 FOR I=LM TO 53180:POKE I,0:NEXT
132 FOR I=LM TO 53180
133 INPUT #1,A
134 POKE I,A
135 NEXT I
136 CLOSE #1
137 GOTO 18
138 PRINT"■ALTERATION TO BUFFER"
139 PRINT"■ALTER BYTES                    [1]"
140 PRINT"■CHANGE LINE FEED SWITCH        [2]"
141 PRINT"■ALTER NUMBER OF BYTES IN ROW [3]"
142 GET AT
143 ON AT GOTO 145,172,184
144 GOTO 142
145 PRINT"■WHICH LINE IS TO BE ALTERED";
146 INPUT WL:IF WL>R3-1 THEN 138
147 PRINT"■■■     WHICH BYTE NUMBER ";:INPUT WB
148 PRINT"■USE '0' and '1' ONLY for alteration."
149 PRINT"■LINE:ROW"
150 WA=0
151 FOR I=0 TO WL-2
152 WA=WA+R(I)
153 NEXT I
154 WA=WA+WB
155 WX=PEEK(LM+WB):WA$="":WY=WX
156 FOR I=1 TO 8
157 WA$=WA$+STR$(WY-INT(WY/2)*2):WY=INT(WY/2)
158 NEXT I
159 PRINT"■";WL;"    ";WB;TAB(12);WA$;" ";WX
160 PRINT"■";TAB(10);
161 INPUT A$
162 A$=LEFT$(A$,8)
163 GOSUB 70
164 PRINTTAB(21);"■";R1;"    "
165 PRINT"■O.K. NOW ?"
166 GET O$
```

```
167 IF O$="N"THEN 138
168 IF O$="Y"THEN 170
169 GOTO 166
170 POKE LM+WB,R1
171 GOTO 18
172 PRINT"■ALTERATION TO LINE FEED SWITCH"
173 PRINT"■WHICH LINE IS SWITCH TO BE ALTERED  ";
174 INPUT SW:IF SW>R3-1 THEN 172
175 PRINT"■LINE NUMBER SW ";F(SW-1)
176 PRINT"■";TAB(14);:INPUT NF
177 PRINT"■O.K. NOW ?"
178 GET S$
179 IF S$="N"THEN 175
180 IF S$="Y"THEN 182
181 GOTO 178
182 F(SW-1)=NF
183 GOTO 18
184 PRINT"■ALTERATION TO NUMBER OF BYTES PER LINE"
185 INPUT"LINE NUMBER ";LN:IF LN>R3-1 THEN 184
186 PRINT"■LINE NUMBER LN ";R(LN-1)
187 PRINT"■";TAB(14);:INPUT NL
188 PRINT"■O.K. NOW ?"
189 GET S$
190 IF S$="N"THEN 186
191 IF S$="Y"THEN 193
192 GOTO 189
193 R(LN-1)=NL
194 GOTO 18
195 PRINT"■":CURSOR 5,10
196 PRINT"HAVE YOU SAVED LAYOUT TO DISK ?"
197 GET S$
198 IF S$="Y"THEN 201
199 IF S$="N"THEN 18
200 GOTO 197
201 END
```



Starting on Page 17 is PART 2 of

MZ-80K GETS SHUGART DRIVES

by

Peter Sydenham

```
PAGE    1

                        ;CODE LOADED AT 9800 ONWARDS
                        ;BY INITIAL BOOT
                        ;AS SOON AS THIS CODE IS LOADED
                        ;EXECUTION STARTS AT 9800
                        ;THE FIRST BYTE MUST BE C3 FOR
                        ;THE INITIAL BOOT CODE TO TRANSFER
                        ;CONTROL
                        ;IF IT IS NOT C3 THE DISC IS NOT
                        ;REGARDED AS A MASTER

                        ;15 9 83 P SYDENHAM
                        ;PREPARED USING DIASSEMBLER
                        ;BY R TANSWELL AND 'ZEN' ASSEMBLER

                        ;TO MAKE DISCS BOOT WITH OTHER
                        ;SOFWARE THE VARIOUS CHECKS
                        ;MUST BE ALTERED TOGETHER
                        ;WITH THE LENGTH,START AND
                        ;EXECUTE ADDRESSES

                        ;NOTE THE LOADING IS FASTER
                        ;THAN THE NORMAL PROGRAM LOAD
                        ;BECAUSE THE DISC READ ROUTINE
                        ;READS WHOLE TRACKS

                        ;REFERENCE TO TOSHIBA DATA ON
                        ;DISC CONTROLLER CHIP IS ADVISED


                                ORG   9800H

9800 C30398                     JP    START2            ;TO GIVE C3 AS FIRST BYTE
9803 3A1110     START2:         LD    A,(1011H)         ;GET DRIVE NO.
9806 325298                     LD    (BUFF1),A         ;FOR (IX+D) USE
9809 210401                     LD    HL,0104H
980C 225398                     LD    (BUFF1+1),HL      ;SET TRK 4,SECT 01
980F 210012                     LD    HL,1200H          ;LOADING ADDR.
9812 225798                     LD    (BUFF1+5),HL
9815 DD215298                   LD    IX,BUFF1          ;SET IX FOR DISC READ
9819 CD8799                     CALL  DSCREAD
981C 3A5998                     LD    A,(BUFF1+7)       ;CHECK ALL SECTORS
981F 47                         LD    B,A               ;HAVE BEEN READ OK
9820 3A5398                     LD    A,(BUFF1+1)       ;BEFORE JUMPING TO
9823 C60A                       ADD   A,0AH             ;EXECUTE ADDRESS
9825 B8                         CP    B
9826 200D                       JR    NZ,EXIT
9828 3A5A98                     LD    A,(BUFF1+8)
982B FE01                       CP    01H
982D 2006                       JR    NZ,EXIT
982F CD7298                     CALL  MOTOFF
9832 C3F721                     JP    21F7H             ;'BASIC'EXECUTE ADDR.

9835 CD7298     EXIT:           CALL  MOTOFF            ;SHUT DRIVES DOWN
9838 CD0900                     CALL  CRLF              ;AND RETURN TO MONITOR
983B 114498                     LD    DE,MESS1
983E CD1500                     CALL  MESSAGE
9841 C38200                     JP    MAINLP
```

```
PAGE   2

  61 9844 45523A43 MESS1:       DB    "ER:CAN'T BOOT"
  61 9848 414E2754
  61 984C 20424F4F
  61 9850 54
  62 9851 0D                    DB    0DH
  63
  64 9852 00        BUFF1:      DB    00H                 ;FOR DRIVE NO.
  65 9853 0401                  DB    04H,01H             ;TRACK 04 SECTOR 01
  66 9855 0032                  DB    00H,50              ;LENGTH IN BYTES
  67 9857 0012                  DB    00H,12H             ;LOAD ADDRESS
  68 9859 0701                  DB    07H,01              ;CHECKS
  69                ;*******************************
  70
  71 985B C5        MOTON2:     PUSH  BC                  ;ROUTINE NOT USED
  72 985C 01F808                LD    BC,08F8H
  73 985F ED78                  IN    A,(C)
  74 9861 010000                LD    BC,0000H
  75 9864 0B        MT2A:       DEC   BC
  76 9865 00                    NOP
  77 9866 00                    NOP
  78 9867 78                    LD    A,B
  79 9868 B1                    OR    C
  80 9869 20F9                  JR    NZ,MT2A
  81 986B 3E01                  LD    A,01H
  82 986D 320210                LD    (MOTFLG),A
  83 9870 C1                    POP   BC
  84 9871 C9                    RET
  85                ;*****************************
  86
  87
  88                ;MOTOFF STOPS DISC MOTORS
  89
  90 9872 C5        MOTOFF:     PUSH  BC
  91 9873 CD489A                CALL  LNGDEL
  92 9876 01F800                LD    BC,00F8H
  93 9879 ED78                  IN    A,(C)               ;TURN MOTOR OFF
  94 987B AF                    XOR   A
  95 987C 320210                LD    (MOTFLG),A
  96 987F 320310                LD    (1003H),A           ;CLEAR BUFFERS
  97 9882 320410                LD    (1004H),A
  98 9885 320510                LD    (1005H),A
  99 9888 320610                LD    (1006H),A
 100 988B C1                    POP   BC
 101 988C C9                    RET
 102               ;-----------------------------
 103
 104
 105               ;SKZERO MAKES DISC HEAD SEEK TRACK
 106               ;ZERO POSITION
 107
 108 988D CD9898   SKZERO:     CALL  SHTSTS
 109 9890 AF                   XOR   A
 110 9891 D3F9                 OUT   (0F9H),A            ;CLEAR TRACK REG.
 111 9893 320010               LD    (OPSTORE),A
 112 9896 D3FA                 OUT   (0FAH),A            ;CLEAR SECTOR REG.
 113 9898 C5       SHTSTS:     PUSH  BC                  ;SHORT STATUS
 114 9899 010000               LD    BC,0000H
 115 989C DBF9     SST1:       IN    A,(0F9H)            ;GET DRDY,CRDY,RQM
 116 989E E603                 AND   03H                 ;LEAVE DRDY & CRDY
 117 98A0 FE02                 CP    02H
```

PAGE 3

```
 118 98A2 2002              JR   NZ,SST2          ;WAIT FOR BOTH DRDY&CRDY
 119 98A4 C1                POP  BC
 120 98A5 C9                RET                   ;CORRECT EXIT ALL OK
 121
 122 98A6 0B        SST2:   DEC  BC
 123 98A7 78                LD   A,B
 124 98A8 B1                OR   C
 125 98A9 20F1              JR   NZ,SST1
 126 98AB C1                POP  BC
 127 98AC 3E32              LD   A,32H            ;EROR CODE
 128 98AE 320810            LD   (1008H),A        ;KEEP IT
 129 98B1 37                SCF
 130 98B2 CD7298            CALL MOTOFF
 131 98B5 C30B10            JP   100BH            ;ERROR ESCAPE
 132                ;*****************************
 133
 134 98B8 1A        UNUSMSS: LD   A,(DE)           ;A MESSAGE
 135 98B9 E67F              AND  7FH              ;ROUTINE THAT IS NOT USED
 136 98BB FE0D              CP   0DH
 137 98BD C8                RET  Z
 138
 139 98BE CD1200            CALL VIDEO
 140 98C1 13                INC  DE
 141 98C2 18F4              JR   UNUSMSS
 142                ;*****************************
 143
 144 98C4 DBFA      STATUS: IN   A,(0FAH)         ;READ STATUS
 145 98C6 E6F0              AND  0F0H             ;MASK OUT RUBBISH
 146 98C8 07                RLCA                  ;CRDY INTO CARRY
 147 98C9 30F9              JR   NC,STATUS        ;WAIT FOR CRDY
 148 98CB E6F0              AND  0F0H
 149 98CD 0F                RRCA                  ;MOVE UNTIL S3 IN BIT 0
 150 98CE 0F                RRCA
 151 98CF 0F                RRCA
 152 98D0 0F                RRCA
 153 98D1 B7                OR   A                ;CLEAR FLAGS
 154 98D2 C8                RET  Z                ;Z=ALL OK
 155 98D3 FE06              CP   06H              ;6=S1 & S2 HIGH
 156 98D5 2812              JR   Z,ST3
 157 98D7 FE0C              CP   0CH              ;0C=CRDY & S1 HIGH
 158 98D9 2004              JR   NZ,ST1
 159 98DB 3E32              LD   A,32H            ;ERROR CODE
 160 98DD 180A              JR   ST3
 161 98DF FE04      ST1:    CP   04H
 162 98E1 2004              JR   NZ,ST2
 163 98E3 3E36              LD   A,36H            ;ERROR CODE
 164 98E5 1802              JR   ST3
 165 98E7 3E29      ST2:    LD   A,29H            ;ERROR CODE
 166 98E9 320810    ST3:    LD   (1008H),A
 167 98EC 37                SCF
 168 98ED C9                RET
 169
 170                ;-----------------------------
 171
 172
 173                ;PRMDRV PRIMES DISC DRIVES
 174
 175 98EE C5        PRMDRV: PUSH BC
 176 98EF E5                PUSH HL
 177 98F0 3A0210            LD   A,(MOTFLG)
```

PAGE 4

```
 178 98F3 0F                    RRCA
 179 98F4 D46E99                CALL  NC,MOTON
 180 98F7 DD7E00                LD    A,(IX+00H )          ;GET DRIVE NO.
 181 98FA E603                  AND   03H                  ;MASK IT
 182 98FC F61C                  OR    1CH                  ;TND,MOTOR,SELECT BITS HI
 183 98FE 320110                LD    (DRVSTRE),A          ;KEEP DRIVE CODE
 184 9901 E60F                  AND   0FH                  ;MASK OUT SELECT BIT
 185 9903 47                    LD    B,A
 186 9904 0EF8                  LD    C,0F8H
 187 9906 ED60                  IN    H,(C)                ;SEND DRIVE TO USE TO FDC
 188 9908 3E32                  LD    A,32H
 189 990A CD489A    PRM1:       CALL  LNGDEL
 190 990D 3D                    DEC   A
 191 990E 20FA                  JR    NZ,PRM1
 192 9910 010000                LD    BC,0000H
 193 9913 DBF9      PRM2:       IN    A,(0F9H)             ;GET DRDY,CRDY,RQM
 194 9915 E607                  AND   07H                  ;MASK OUT RUBBISH
 195 9917 FE06                  CP    06H                  ;DRDY AND CRDY OK ?
 196 9919 2017                  JR    NZ,PRM5              ;NZ=KEEP TRYING
 197 991B DD4E00                LD    C,(IX+00H )          ;GET DRIVE NO.
 198 991E 0600                  LD    B,00H
 199 9920 210310                LD    HL,1003H             ;IS HEAD POSITION
 200 9923 09                    ADD   HL,BC                ;KNOWN FOR DRIVE ?
 201 9924 CB46                  BIT   0,(HL)
 202 9926 2803                  JR    Z,PRM4               ;Z=NOT KNOWN
 203 9928 E1        PRM3:       POP   HL
 204 9929 C1                    POP   BC
 205 992A C9                    RET                        ;CORRECT EXIT
 206
 207
 208 992B CD8D98    PRM4:       CALL  SKZERO
 209 992E CBC6                  SET   0,(HL)               ;HEAD POSITION NOW KNOWN
 210 9930 18F6                  JR    PRM3
 211 9932 0B        PRM5:       DEC   BC
 212 9933 78                    LD    A,B
 213 9934 B1                    OR    C
 214 9935 20DC                  JR    NZ,PRM2
 215 9937 3E32                  LD    A,32H                ;ERROR CODE
 216 9939 320810                LD    (1008H),A
 217 993C 37                    SCF
 218 993D CD7298                CALL  MOTOFF
 219 9940 C30B10                JP    100BH                ;ERROR ESCAPE
 220
 221                ;-----------------------------------
 222                ;VALDAT VALIDATE DATA TO BE USED
 223                ;FOR DISC ACCESS
 224
 225 9943 DD7E00    VALDAT:     LD    A,(IX+00H)           ;GET DRIVE NO.
 226 9946 FE04                  CP    04H                  ;LESS THAN 4 ?
 227 9948 3018                  JR    NC,VAL1              ;NC=FAULT
 228 994A DD7E01                LD    A,(IX+01H )          ;GET TRACK NO.
 229 994D FE46                  CP    46H                  ;MUST BE < 46H
 230 994F 3011                  JR    NC,VAL1
 231 9951 DD7E02                LD    A,(IX+02H )          ;GET SECTOR NO.
 232 9954 B7                    OR    A
 233 9955 280B                  JR    Z,VAL1               ;ZERO NOT VALID
 234 9957 FE11                  CP    11H                  ;01 TO 10 ARE VALID
 235 9959 3007                  JR    NC,VAL1
 236 995B DD7E03                LD    A,(IX+03H )          ;GET LENGTH TO LOAD
 237 995E DDB604                OR    (IX+04H )            ;CHECK > ZERO
```

PAGE 5

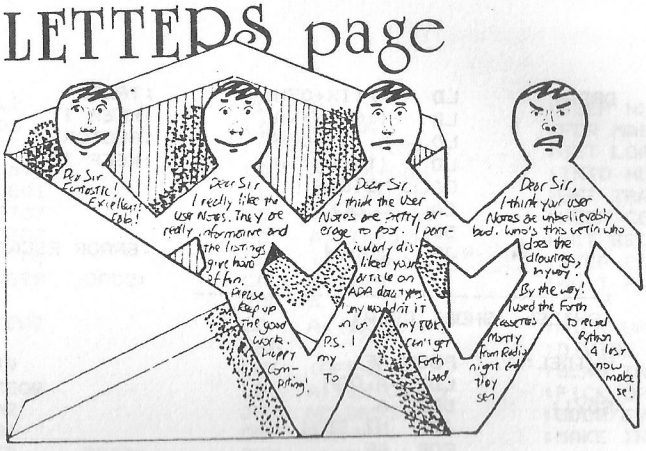
```
 238 9961 C0                    RET  NZ                  ;NZ=OK
 239 9962 3E38      VAL1:       LD   A,38H               ;ERROR CODE
 240 9964 320810                LD   (1008H),A
 241 9967 CD7298                CALL MOTOFF
 242 996A 37                    SCF
 243 996B C30B10                JP   100BH               ;ERROR ESCAPE
 244
 245                ;-------------------------------
 246
 247                ;MOTON   TURNS ON DRIVE MOTORS
 248                ;NOTE USE OF IN A,(C) TO SEND
 249                ;DATA TO THE DISC CONTROLLER
 250
 251                ;EXAMINE CIRCUIT OF CONTROLLER
 252                ;CARD FOR DETAILS
 253
 254
 255 996E F5        MOTON:      PUSH AF
 256 996F C5                    PUSH BC
 257 9970 CD489A                CALL LNGDEL
 258 9973 0EF8                  LD   C,0F8H
 259 9975 0608                  LD   B,08H
 260 9977 ED78                  IN   A,(C)               ;TURN MOTOR ON
 261 9979 C1                    POP  BC
 262 997A F1                    POP  AF
 263 997B C9                    RET
 264
 265
 266                ;-------------------------------
 267
 268                ;CLOCK RE-ENABLES INTERRUPTS FOR
 269                ;      CLOCK
 270 997C F5        CLOCK:      PUSH AF
 271 997D 3A9C11                LD   A,(119CH)
 272 9980 FEF0                  CP   0F0H
 273 9982 2001                  JR   NZ,CLC1
 274 9984 FB                    EI
 275 9985 F1        CLC1:       POP  AF
 276 9986 C9                    RET
 277
 278                ;-------------------------
 279
 280                ;DSCREAD  READS DISC WHOLE TRACK
 281                ;AT A TIME USING TND SIGNAL TO
 282                ;CONTROLLER CHIP TO GET TRACK
 283                ;END INDICATION
 284
 285
 286 9987 CD4399    DSCREAD:    CALL VALDAT
 287 998A 3E0A                  LD   A,0AH               ;NUMBER OF TRIES TO READ
 288 998C 320710                LD   (1007H),A
 289 998F CDEE98    DRD1:       CALL PRMDRV
 290 9992 3A0110                LD   A,(DRVSTRE)         ;GET DRIVE NO.
 291 9995 47                    LD   B,A
 292 9996 0EF8                  LD   C,0F8H
 293 9998 D9                    EXX
 294 9999 0EFB                  LD   C,0FBH              ;PORT
 295 999B DD5E03                LD   E,(IX+03H )         ;GET LENGTH & CALC.
 296 999E DD5604                LD   D,(IX+04H )         ;NO. OF SECTORS TO
 297 99A1 CB13                  RL   E                   ;READ
```

PAGE 6

```
 298 99A3 CB12          RL   D              ;KEEP NUMBER TO READ IN D
 299 99A5 1E03          LD   E,03H          ;FOR MASKING
 300 99A7 DD6E05        LD   L,(IX+05H )    ;GET LOAD ADDRESS
 301 99AA DD6606        LD   H,(IX+06H )    ;INTO HL REG.
 302 99AD DD7E01        LD   A,(IX+01H )    ;GET TRACK TO START
 303 99B0 DD7707        LD   (IX+07H),A     ;KEEP COPY
 304 99B3 DD7E02        LD   A,(IX+02H )    ;GET SECTOR TO START
 305 99B6 DD7708        LD   (IX+08H),A     ;KEEP COPY
 306 99B9 CD9898  DRD2: CALL SHTSTS         ;WAIT FOR CRDY
 307 99BC AF            XOR  A
 308 99BD DD7E07        LD   A,(IX+07H )    ;PICK UP TRACK
 309 99C0 1F            RRA                 ;DIVIDE BY 2
 310 99C1 D3F9          OUT  (0F9H),A       ;SEND TRACK TO FDC
 311 99C3 DD7E08        LD   A,(IX+08H )    ;PICK UP SECTOR
 312 99C6 3002          JR   NC,DRD3        ;ODDS/EVENS FOR SIDES
 313 99C8 F680          OR   80H            ;MAKE INTO SIDE 2
 314 99CA D3F8    DRD3: OUT  (0F8H),A       ;SECTOR & SIDE
 315 99CC CD409A        CALL SHTDEL         ;WAIT A WHILE
 316 99CF 3E70          LD   A,70H          ;SEEK & READ CODE
 317 99D1 320010        LD   (OPSTORE),A    ;KEEP IT
 318 99D4 F3            DI
 319 99D5 D3FA          OUT  (0FAH),A       ;SEEK & READ TO FDC
 320 99D7 0680    DRD4: LD   B,80H          ;BYTES/SECTOR
 321 99D9 DBF9    DRD5: IN   A,(0F9H)       ;GET DRDY,CRDY,RQM
 322 99DB A3            AND  E              ;MASK OUT RUBBISH
 323 99DC 28FB          JR   Z,DRD5
 324 99DE 0F            RRCA                ;RQM INTO CARRY
 325 99DF 300F          JR   NC,DRD6
 326 99E1 EDA2          INI                 ;TAKE BYTE INTO (HL)
 327 99E3 C2D999        JP   NZ,DRD5        ;DO WHOLE SECTOR
 328 99E6 DD3408        INC  (IX+08H )      ;NEXT SECTOR
 329 99E9 15            DEC  D              ;DEC SECTOR COUNTER
 330 99EA C2D799        JP   NZ,DRD4        ;NZ=MORE TO DO
 331 99ED D9            EXX
 332 99EE ED78          IN   A,(C)          ;SEND TND HI
 333 99F0 CDC498  DRD6: CALL STATUS
 334 99F3 3817          JR   C,DRD8
 335 99F5 F5            PUSH AF
 336 99F6 DD7E08        LD   A,(IX+08H )    ;SECTOR
 337 99F9 FE11          CP   11H            ;END OF TRACK ?
 338 99FB 3807          JR   C,DRD7         ;C=NOT END
 339 99FD DD360801      LD   (IX+08H),01H   ;RESET TO SECTOR 1
 340 9A01 DD3407        INC  (IX+07H )      ;INC TRACK
 341 9A04 F1      DRD7: POP  AF
 342 9A05 CD6E99        CALL MOTON          ;WHY ? SHOULD BE MOTOFF?
 343 9A08 CD7C99        CALL CLOCK
 344 9A0B C9            RET                 ;CORRECT EXIT
 345
 346
 347 9A0C FE06    DRD8: CP   06H            ;LAST SECTOR OF TRACK ?
 348 9A0E 200A          JR   NZ,DRD9        ;NZ=NOT LAST SECTOR
 349 9A10 DD3407        INC  (IX+07H )      ;INC TRACK
 350 9A13 DD360801      LD   (IX+08H),01H   ;RESET SECTOR
 351 9A17 C3B999        JP   DRD2           ;DO NEXT TRACK
 352 9A1A 3A0710  DRD9: LD   A,(1007H)      ;1 TRY GONE
 353 9A1D 3D            DEC  A
 354 9A1E 320710        LD   (1007H),A
 355 9A21 CA2A9A        JP   Z,DRDA         ;TRY AGAIN
 356 9A24 CD8D98        CALL SKZERO
 357 9A27 C38F99        JP   DRD1
```

PAGE 7

```
9A2A DD7E07   DRDA:     LD   A,(IX+07H)      ;TRACK
9A2D 320910             LD   (1009H),A       ;KEEP IT
9A30 DD7E08             LD   A,(IX+08H)      ;SECTOR
9A33 320A10             LD   (100AH),A       ;KEEP IT
9A36 CD7298             CALL MOTOFF
9A39 CD7C99             CALL CLOCK
9A3C 37                 SCF
9A3D C30B10             JP   100BH           ;ERROR ESCAPE

                ;-----------------------------------
                ;SHTDEL  SHORT DELAY

9A40 F5       SHTDEL:   PUSH AF
9A41 3E0A               LD   A,0AH
9A43 3D       SDY1:     DEC  A
9A44 20FD               JR   NZ,SDY1
9A46 F1                 POP  AF
9A47 C9                 RET
                ;-----------------------------

                ;LNGDEL  LONG DELAY


9A48 F5       LNGDEL:   PUSH AF
9A49 3E0A               LD   A,0AH
9A4B CD409A   LDY1:     CALL SHTDEL
9A4E 3D                 DEC  A
9A4F 20FA               JR   NZ,LDY1
9A51 F1                 POP  AF
9A52 C9                 RET

                ;-----------------------------


                CRLF:     EQU  0009H
                MESSAGE:  EQU  00015H
                MAINLP:   EQU  00082H
                MOTFLG:   EQU  01002H
                VIDEO:    EQU  00012H


                OPSTORE:  EQU  1000H           ;FDC CODE STORE
                DRVSTRE:  EQU  1001H           ;DRIVE NO. STORE

                          END
```



Dear S.U.N.

Having a particular interest in data storage, retrieval and analysis, it was gratifying to find the article by Drs. A.K. Black and G. R. Glover on this very subject in SUN Issue 8 pages 20-21.

I tried the method shown initially with a test sample of 50 items of data each 35 bytes in length and subsequently (doubting my own calculations) with 435 items of data each 30 bytes in length as shown in the article.

I taped the data I had poked into RAM on a clear tape as directed. On attempting to retrieve the data in the manner described on P.21 (after 'switch on') with a simple retrieval program I could not get this method to work as shown. The area of RAM that should have contained my data only held its original contents. However I did find that:-

1) Load BASIC
2) LIMIT nnnnn (in this case 40197)
3) LOAD tape containing previously stored data
4) LOAD simple retrieval program
5) RUN

worked beautifully. At no time did I have to return to MONITOR.

I did take this a little further by writing a small retrieval program which peeked my data area of RAM and with the appropriate calculations to convert the values obtained from ASCII code toi DISPLAY code poked the data direct to Video RAM which is obviously a great deal faster.

The letter from M BELLAMY of SUSSEX was also of interest where he writes about the 'look up' tables in Monitor and I shall be trying his method in my data retrieval program (SUN 9 P.61).

Keep up the good work. Your 'goodies' for everyone policy is terrific.

J.R. GREGORY
SOUTH YORKS.

****

Dear Sirs,

Whilst writing a home accounts program recently I had to write a routine to express all values with two digits affter the decimal point and to print them so that the decimal points were all underneath eath other, regardless of the size of the number.

Before calling the routine the value to be printed must be converted into a string, P$, using the STR$ command. Also the position at which printing is to end must be stored in the variable P. Line 100 in the listing is not part of the routine but is an example of what must be done if the value of a variable V is to be printed finishing at position 60. The listing uses the PRINT/P command but of course thiw may be replaced with the PRINT command for on screen printing.

```
90   REM * CALL FROM MAIN PROGRAM *
100  P=60:P$=STR$(V):GOSUB500
490  REM
500  REM * SUBROUTINE TO PRINT VALUES TO 2 DEC PLACES *
510  IFLEN(P$)<3THENP$=P$+".00"
520  IFMID$(P$,(LEN(P$)-2),1)="."THEN550
530  IFMID$(P$,(LEN(P$)-1,1)="."THENP$=P$+"0":GOTO550
540  P$=P$+".00"
550  PRINT/PTAB(P-LEN(P$);P$
560  RETURN
```

R. WHEELER
SURREY

****

Dear Sir,

I attach some suggestion for WP/2 users (SUN No 6).

- deleting PRINT"H" from line 780 will save one empty page
- for SP-5025 users small changes must be done
  - line 1600 shuld be changed to:
    IF(MID$(A$(X+1),I,1)=" ")THEN1610:NEXTI
  - PRINT@0,5 must be removed from line 830
- I found NAME FILE idea useless so I use placed as first character in line for printing NEW PAGE. To do so:
  - change line 730 to:
    730 FORI=XTOY:IFLEFT$(A$(I),1=" "THENGOTO1860
  - change line 740 to:
    740 PRINT/PA$
  - change line 1860 ro
    1860 PRINT"H":GOTO760
  - delete lines 750 & 1870-2020
- function Replace is inconvenient if only small changes should be done. I following new Update function placed in lines 1960-2020. Unfortuantely U function is removing all leading spaces and text after coma.

```
1960 PRINT"CUPDATE"
1970 INPUT"LINE NUMBER ? "I:IFI>LTHEN160
1980 PRINT" UPDATE TEXT  ":PRINTSPC(15);" ":PRINT:PRINT"";A$(I)
1990 IFLEN(A$(I))<39THENPRINT"↑↑";GOTO2020
2000 IFLEN(A$(I))<79THENPRINT"↑↑↑";GOTO2020
2010 IFLEN(A$(I))<119THENPRINT"↑↑↑↑"
2020 INPUT " ";A$:PRINT:A$(I)=A$:GOTO1970
```

J. ROEHR
KUWAIT

Dear Sharpsoft,

Enclosed please find 2 programs for the MZ-80K written by my 11 year old son, Peter.

"Kaleidoscope" uses "reflections" in four quadrants & symmetrical graphics symbols to emphasise the kaleidoscope effect.

"Secret Messages" uses a 26 x 26 array and a keyword to cipher messages - even the same letter produces different letters when the message is ciphered.

I enjoy your Magazine and believe I have the only MZ-80K in Dar es Salaam! I also have an RX80 printer and a Powerbank, which is absolutely essential as we experience frequent voltage variations. Many thanks.

G. ROUNCE
TANZANIA

```
1 PRINT'ﾋ: GOSUB500
10 PRINT 'ﾋ
20 T=INT(RND(1)*11)+10
30 FOR NT=1 TO T
40 RESTORE
50 READ N
60 R=INT(RND(1)*N+1)
70 FOR I=1 TO R
80 READ D1,D2,D3,D4
90 NEXT
100 PX=INT(RND(1)*20+1)
110 PY=INT(RND(1)*12+1)
120 POKE 53248+PX+PY*40,D1
130 POKE 53248+(40-PX)+PY*40,D2
140 POKE 53248+PX+(24-PY)*40,D3
150 POKE 53248+(40-PX)+(24-PY)*40,D4
160 NEXT
170 GET A$:IF A$="" THEN 170
175 IFA$="E"THEN END
180 USR(62)
190 RUN 10
200 DATA 5
210 DATA 208,239,239,208
220 DATA 209,210,209,210
230 DATA 211,211,212,212
240 DATA 212,212,211,211
250 DATA 246,249,249,246
500 PRINT@12,10;"K A L E I D O S C O P E"
510 PRINT@14,10;"          BY"
520 PRINT@16,10;"  PETER J. ROUNCE."
530 PRINT@22,10;"Press any key to continue."
540 PRINT@18,10;"Press any key to change pattern except 'E'"
550 PRINT@20,10;"Press 'E' to end."
555 FORA=1TO5000:NEXT
560 RETURN
```

```
1 PRINT"ᑕ"
2 GOSUB1000
5 PRINT"ᑕ":TEMPO 7
7 PRINT @11,13;"**************"
8 PRINT @12,13;"*PLEASE WAIT*"
9 PRINT @13,13;"**************"
10 DIM A$(26,26):XD=0:QD=5
20 XD=XD+1:FOR X=XD TO 26:QD=QD+2
22 IF QD>26 THEN QD=1
23 Q=QD
25 FOR Y=X TO 26
27 Q=Q+1
30 IF Q>26 THEN Q=1
40 A$(X,Y)=CHR$(Q+64)
42 A$(Y,X)=CHR$(Q+64)
60 NEXT
70 NEXT
75 PRINT"ᑕ"
80 INPUT "WHAT IS THE KEYWORD ?";K$
90 K=LEN(K$)
100 PRINT:PRINT
110 INPUT "WHAT IS THE MESSAGE ?";M$
120 M=LEN(M$)
130 DIM B(M)
135 LET J=1
140 FOR I=1 TO M
150 LET B(I)=ASC(MID$(K$,J,1))-64
155 IF MID$(M$,I,1)=" " THEN LET B(I)=0:GOTO 170
160 LET J=J+1:IF J>K THEN J=1
170 NEXT
180 LET C$=""
190 FOR I=1 TO M
195 IF B(I)=0 THEN C$=C$+" ":GOTO 210
200 LET C$=C$+A$(B(I),ASC(MID$(M$,I,1))-64)
210 NEXT
220 PRINT:PRINT
230 PRINT "CODED MESSAGE IS :";C$
240 PRINT @24,0;"AGAIN (Y/N) ? ";
250 GET A$:IF A$<>"" THEN 300
260 PRINT "ᑫ";
265 FOR I=1 TO 100:NEXT
270 GET A$:IF A$<>"" THEN 300
280 PRINT " ";
285 FOR I=1 TO 100:NEXT
290 GOTO 250
300 IF A$="Y" THEN 330
310 IF A$="N" THEN 340
320 GOTO 250
330 PRINT"ᑫ";A$;:MUSIC "R":PRINT "":GOTO 80
340 PRINT"ᑫ";A$;:MUSIC"R":PRINT"":END
350 END
1000 PRINT@5,5;"S E C R E T  M E S S A G E S"
1010 PRINT@7,14;"BY"
1020 PRINT@9,7;"PETER J. ROUNCE (AGE 11)
1030 FORA=1TO3000:NEXT:RETURN
```

Dear Sirs,

I enclose a small program which calculates VAT, adds or subtracts, which may be of interest to your Members.

```
100 REM WRITTEN BY LARRY GALLIFORD ON 10/12/83 FOR SHARP MZ-80K
110 REM ____________________________________________________
120 REM VAT CALCULATOR @ 15%:PRINT"C"
130 INPUYT"PLEASE ENTER YOUR AMOUNT.";A
140 IF A=0 THEN 220
150 B=A*3/23:C=A-B
160 PRINT" VAT SUBTRACTED";TAB(20);"VAT ADDED "
170 PRINT " ";A:PRINT" ";B
180 PRINT " ";C
190 B=A*3.45/23:E=A+B
200 PRINTTAB(19);"   ";A:PRINTTAB(19);B:PRINTAB(19);" ";E:PRINT
210 FORX=1TO33:PRINT"-";:NEXT X:PRINT:PRINT:GOTO 130
220 PRINT "C":END.
```

Dear Sharpsoft,

I have received the Sharp Service manual and the Word Power Package.

After working on the program I seem to have a problem with Wordpower when I require to input a text file.

After selecting Option 1 the message READ CASSETTE is displayed and the Tape recorder is started when the file is found the Message "THIS FILE(Y/N)" is displayed, if I select Y the program runs properly but if I select N the Program just loops.

A.F. WHITER
ESSEX

This problem has now been fixed. It only occurs in the MZ-80A version. Those who own a machine code monitor program like PA-MON can do the modification by changing the contents of address 330AH from EEH to 88H and save the new WORDPOWER to tape. WORD-POWER starts from address 1200H and ends at 3500H with an execute address at 1200H.

SHARPSOFT

Reference - Issue No.9 (p.3).

FORTH PUBLIC DOMAIN SOFTWARE - EDMUND RAMM

Any member wishing to have a copy of Mr. Edmun Ramm's disc-based Z80 fig-FORTH running under CP/M on the MZ-80K (we have to date not received any other versions) - send in six pounds to cover the cost of Diskette, copying, post & pack.

SHARPSOFT LIMITED
CRISALLEN HOUSE
86-90 PAUL STREET
LONDON EC2A 4NE
TEL: 01-739-8559

SHARPSOFT

Sharpsoft Ltd., 86-90 Paul Street, London EC2A 4NE
Printed by Oldham Press (T.U.), Chatham, Kent.