Please send articles,
letters, comments to:-
C.S.Giles
75 Oakdene Cres,
Mile Oak,
Porstlade,
West Sussex
BN4 2RP
0273 419100

Please send P.D. Software
requests to:-
R.Foulser
20 Wentworth Close
Barnham
West Sussex
0243 553669

Membership Enquiries to
Keith Stokes
Hillcroft
Codmore Hill
Pulborough
West Sussex
RH20 1BQ
07982 2299

Adventure Column to:-
Tony Stansfield
8 Coltsfoot Drive
Altrincham
Cheshire

MEMBERS

J.Raper 14 First Avenue, Worthing, West Sussex
C.Heather, Yew Tree Cott, Tong Rd., Brenchley, Tonbridge, Kent. TN12 7AN
Graham Bettany Ipswich 0473 49507 (Everything)
Mike Smallman 0206 540540
Adrien Cull Woolaston Inn, Woolaston, nr. Lydney, Glos GL15 6PW. (Comms)
Dave Salvage 38 Preistman Point, Rainhill Way, London E3 3EY (Everything)
S.Peterson Herningvej 13, 139 9220 Aalborg Ost Denmark (software Pascal and M/C (University of Aalsborg))
Barry Kidder. 16 Osborne Avenue,Stanwell,Staines,Middlesex,TW19 7AF

UK EUG

JAN 1987

UK EINSTEIN USER GROUP
NEWSLETTER

Volume 2 Number 3 January 1987

CONTENTS

REGULARS

REVIEWS

FEATURES

GAMES

UTILITIES

HINTS AND TIPS

EDITORIAL

We have had a number of letters offering help with serial printers, al of them altering bytes in DOS, but I think that with the article on DOS in this issue will make all clear and work with all versions of DOS. This does apply only to those programs using the DOS bios and won't work with those programs using their own printer output routines.

ARTICLES

When submitting articles and programs it would be helpful if they can be sent in on disc to save us the time taken in typing them in ourselves. Sometimes articles are omitted because we simply have not the time to put them in ourselves, and we are prone to meking mistakes as none of us are typists. (Except the two finger type) A note as to what environment it runs is also useful, i.e. XBAS with DOS 1.11 or BBCBASIC with DOS 2.01 or even LOGO with DOS 1.39. Whether it needs 80 columns or a printer, in fact anything that will help to identify possible areas of incompatability.

Don't let these 'rules' put you off as most articles and programs are published. In fact we are very short of items to put in at present and this is one factor that delays publication. If we have nothing to publish from you we have to write it ourselves and it is not easy to fill the newsletter with relevant information from just two pens!! So get out all those little routines that you have developed to make life easier for yourself and send them in.

E.A.E.A.

What we did last month with NVADERS and TRAXMAN was to introduce, in computer jargon, what is called a PATCH. A software patch is a piece of code that is added to a program to correct or add a facility. Usually a machine code program is a continuous block of code and you cannot just make space for some additional code in the middle of the program. What can be done is to replace a piece of code, usually three bytes that form a complete instruction, and insert a call or a jump which also uses three bytes. Now we can jump or call to a location where there is space to add your new routine, (this can be thought of as a GOSUB in BASIC), this new routine is then written to contain the code replaced by the call we inserted, plus our new routine, on returning to the program from our call we can carry on as before. (Again this equates to the RETURN in BASIC). We must of course ensure that we return to the program without having altered any registers or memory locations that are essential to the running of the program, this can achieved by pushing to and poping from the stack, if needed. Some of the alterations, or patches may be frowned on by the purists but they do work. It would be possible to leave the keyboard working along with the joystick but to keep things simple in most cases this has not been done and the new programs will only work with the joystick, this really is no problem as you should have a keyboard copy as well. Don't forget always work with a copy of the program you are patching, I know you always do!

Chuckie Egg has always been a favourite of mine but using five fingers at once does not come easy, (Mike reckons it's your age that counts!). Now this presented more of a challenge, than last months, but faint heart never won fair software, (Mike reckons I shouldn't bother with the jokes as well!). The disc is copy protected to a certain extent, in that there are no system tracks or directory tracks as would be found on a usual disc. You can see this by loading DOS from another disc and doing a DIR on the Chuckie disc, this gives no files present. Before we can start to modify the program we need to get a copy into memory so it can be saved to disc as a file. In this case it is fairly simple, just reset the machine without a disc in drive 0. We are now in MOS, insert the Chuckie Egg disc, now do a MOS read with

```
    R 0080 416F<E>.
```

Remove the Chuckie disc and insert a disc with the system tracks present, and do a CTL/BREAK to load DOS. Assuming the disc in drive 0 is the disc we want to copy to,

```
SAVE 66 CHUCKJOY.COM<E>.
```

We now have a copy of Chuckie Egg that can be loaded in the normal way. Now for the patches to give joystick control, the keyboard can still be used provided the joystick is left centralised. So you can still move through the screens by holding down all five keys together, as mentioned in an earlier newsletter. If you have just done the above then you do not need to load the program, if not, enter DOS and,

```
LOAD CHUCKJOY.COM<E>.       (66 Blocks).
```

Enter MOS,

```
M 1E51<E>
  1E51 CD004200000000000000000000000000<E>
  1E61 000000000000C900.<E>
M 4200<E>
  4200 AF575F3E05D338061010FEDB38FEFF20<E>
  4210 051101101807FE1030031101083E04D3<E>
  4220 38061010FEDB38FEFF20083E0282571E<E>
  4230 01180AFE1030063E0482571E01DB20CB<E>
  4240 4720083E015F8232489DC97A32489D7B<E>
  4250 FE012001C921309D06080E005ECD1E9D<E>
  4260 CD279D23A620013FCB11232310EE792F<E>
  4270 32489DC9FFFFFFFFFFFFFFFFFFFFFFFF.<E>
M 4080<E>
  4080 06003E05D338DB38FEFF200406101806<E>
  4090 FE1530020608DB20CB47200765018032<E>
  40A0 489DC97832489DC9.<E>
M 08C4<E>
  08C4 204B455953202F204A4F59535449434B<E>
  08D4 202A20532053544152542O2A20522052<E>
  08E4 45444546494E45204B455953.<E>
Y<E>
SAVE 66 CHUCKJOY.COM<E>
```

If you have a favourite program that you would like converted to joystick, then just send it on disk together with 50p in stamps(p&p) to EAEA, 80 Dales Rd, Ipswich, Suffolk, IP1 4JR. and we'll put a joystick version for you on the other side for free.

DISC OPERATING SYSTEM (by EAEA)

On switching on EINSTEIN and pressing control-break DOS is loaded into memory, page 57 of the DOS/MOS handbook gives a memory map of the RAM at this stage. Xtal DOS is loaded as a CPM operating system would be. Xtal DOS uses different names for the modules in use compared to CPM. The DMM (Dos Monitor Module) equates to the CCP (Console Command Processor), the OSM (Operating System Module) to the BDOS (Basic Disc Operating System) and the HDM (Hardware Dependent Module) to the BIOS (Basic Input Output System), the TPA (Transient Program Area) being used for both systems.

Locations 0000 to 00FFH are reserved for the System Parameters, and contain jump vectors to the actual DOS, or MOS, system information, and a buffer area for disc usage. The TPA is the area of RAM that will hold any programs and data that are being executed or used, it extends from location 0100H to the start of the DMM at E100H. The DMM/CCP contains the routines that monitor the console, and the DOS utilities such as DIR, ERA, etc. once a program is loaded this area of RAM can be overwritten, as the loaded program controls console activty. The DMM/CCP is reloaded on a warm start along with the OSM/BDOS. The OSM/BDOS is located from EC00H to F9FFH, and contains the routines used to control the operation of the peripherals. The HDM/BIOS goes from FA00H to the top of memory FFFFH, containing machine specific routines that actually control the hardware devices. The HDM/BIOS module is the one module that will differ between CPM machines, the other routines although not the same should be compatible thus enabling software that is written under CPM to be run on all CPM machines. The OSM/BDOS and HDM/BIOS go together to form the FDOS (Full Disc Operating System). If we load DOS, then enter MOS we can examine the system parameter area of memory by:

```
 1..T 0000 0080<E>
```

The first three bytes contain C303FA hex, meaning of course jump to location FA03H this is the jump vector to the HDM/BIOS warm start location. Execution from location FA03H will cause a fresh copy of the DMM/CCP and OSM/BDOS to be written into memory. The vectors to OSM/BDOS and HDM/BIOS are also re-initialized, (locations 0000, and 0005) The HDM/BIOS and TPA regions of memory are not altered on a warm start.

The fourth byte of the system parameters is the IOBYTE which holds information on the peripherals, console,printer. Byte five contains the current disc drive. The next three bytes hold the vector to the OSM/BDOS, if we look at the first locations again:

```
 1..MOS<E>
 2..T 0000 0080<E>
```

Location 0005 holds a jump to EC00H this being the OSM/BDOS entry point. The next addresses are the restart locations for RST 8 through to RST 38. The region of memory from 005CH is the default File Control Block (FCB) area. Details from the FCB, of the file to be used, from the disc directory are entered here. Location 006CH onwards are set aside as a second FCB area. The general buffer area is from 0080H, it is here that the command line tail will be stored, and any data during disc transfers. To summarise the memory map of the system parameter area looks like this:

Address	Code	Outcome
M 0-2	C303FA	Jump to BIOS warm start vector
3	00	IOBYTE status
4	00	User number
5-7	C300EC	Jump to DOS
8-38	Restart area	
5C-7F	Default FCB area	
80-FF	Buffer	

We can explore the BIOS vectors by examining the area of memory from FA00 as follows:-

BIOS	FA00	Cold Start Vector	C3FA5C
BIOS+3	FA03	Warm Start Vector	C3FAC9
Bios+6	FA06	Console Status Vector	C3FA80
BIOS+9	FA09	Console Input Vector	C3FA34
BIOS+12	FA0C	Console Output Vector	C3FA37
BIOS+15	FA0F	Printer Output Vector	C3FA3B
BIOS+18	FA12	RS232 Output vector	C3FA3F
BIOS+21	FA15	RS232 Input vector	C3FA43
BIOS+23	FA18	Vector to disc routines	C3FA49

If you are using a different version to DOS 1.31 it is a simple matter to work out the locations, remembering that the vector from location 0000 points to the second vector in the BIOS, (Warm Start). It can be seen that is is an easy matter now to redirect the devices connected to the machine. If we wanted to make the centronics printer output go to the serial port, all that is needed is to load DOS, go back to MOS and modify using the M command the vector at BIOS+15 (FA0F) to read the same as the vector at BIOS+18 (FA12). Another method would be to modify the memory that is called by the vector.

ATOMIC

A game that is simple in essence but that requires logic, and speed of reaction along with enough persistence to keep trying when you get it wrong. The idea is to arrange your route around the screen to get to eight numbered locations without backtracking or crossing over the path you have already been on. Well worth typing it in.

```
  10 REM******************************
  20 REM* COPYRIGHT G.M.BETTANY AND  M.E.SMALLMAN FREE TO UKEUG *
  50 REM* WORLD BY MARTIN PAGE   ******************************
 100 GOSUB 10000 REM***INSTRUCTIONS***
 110 GOSUB 1000REM***INITIALIZATION***
 120 GOSUB 2000REM***DISPLAY***
 130 GOSUB 3000REM***GAME***
1000 REM***INITIALIZATION***
1010 SHAPE 130,"FEFE383F3F38FEFE":SHAPE 131,"7F7F1CFCFC1C7F7F"
1030 SHAPE 132,"C3C3FFFFFFDDDD18":SHAPE 133,"1818DDFFFFFFC3C3"
1050 LET T=0000:REM *** TIME ***
1060 LET XT=0:LET YT=1:REM *** TANK ***
1065 LET OXT=0:OYT=1
1070 LET D=0:LET G=0:LET F=0
1080 LET RODS=01
1090 SHAPE 142,"1028444428100001":SHAPE 143,"1030101010380002"
1110 SHAPE 144,"3844043840 7C0003":SHAPE 145,"3844180444380004"
1130 SHAPE 146,"1828487C08080005":SHAPE 147,"7C40780444380006"
1150 SHAPE 148,"3840784444380007":SHAPE 149,"7C04081020200008"
1170 SHAPE 150,"3844384444380009"
1180 REM IOM3,0
1900 RETURN
2000 REM***DISPLAY***
2010 TCOL15,4:BCOL4
2020 CLS32
```

```
2030 PRINT @ 25,3,"TIME"
2040 PRINT @ 25,19,"CODE"
2050 DRAW 0,15 TO 0,191 TO 191,191 TO 191,15 TO 0,15
2060 DRAW 7,23 TO 7,184 TO 184,184 TO 184,23 TO 7,23
2070 FILL 20,20,9
2080 IOM 5,0
2090 IOM 4,0
2100 TCOL15,6
2110 FOR A=0 TO 8
2120 LET X=RND (20)+2:LET Y=RND (18)+2
2130 IF VPEEK(X*8+8192+Y*256)= 246 THEN GOTO 2120
2140 PRINT @ X,Y,CHR$(142+A)
2150 NEXT A
2160 PRINT @ 0,1,CHR$(131)
2170 TCOL15,1
2180 PRINT @ 25,20,"    "
2190 RETURN
3000 REM *** GAME ***
3010 TCOL15,1
3020 PRINT @ 25,4,T
3030 M=KBD
3040 IF M=75 THEN OXT=XT:OYT=YT
3050 IF M=76 THEN OXT=XT:OYT=YT
3060 IF M=83 THEN OXT=XT:OYT=YT
3070 IF M=87 THEN OXT=XT:OYT=YT
3080 IF M=75 AND D=2 THEN G=10
3090 IF M=75 AND D=3 THEN G=26
3100 IF M=76 AND D=2 THEN G=11
3110 IF M=76 AND D=3 THEN G=27
3120 IF M=83 AND D=1 THEN G=26
3130 IF M=87 AND D=0 THEN G=11
3140 IF M=87 AND D=1 THEN G=10
3150 IF M=83 AND D=0 THEN G=27
3160 IF M=75 AND D=0 THEN G=0
3170 IF M=76 AND D=1 THEN G=0
3180 IF M=83 AND D=2 THEN G=4
3190 IF M=87 AND D=3 THEN G=4
3200 IF M=75 THEN LET XT=XT-1:F=0:D=0:ELSE IF M=76 THEN LET
     XT=XT+1:F=1:D=1
3210 IF M=87 THEN LET YT=YT-1:F=2:D=3:ELSE IF M=83 THEN LET
     YT=YT+1:F=3:D=2
3220 IF M = 76 OR M =75 OR M =83 OR M =87 THEN 3230ELSE 3290
3230 IF VPEEK(8195+XT*8+YT*256)=241 THEN 5000
3240 IF VPEEK(8195+XT*8+YT*256)=144 THEN 5000
3250 IF VPEEK(7+XT*8+YT*256)<> 0  THEN GOSUB 4000
3260 IF XT<0 THEN GOTO 5000
3270 PRINT @ XT,YT,CHR$(130+F)
3280 PRINT @ OXT,OYT,CHR$(164+G)
3290 LET T=T+1:IF T=1000 THEN GOTO 5000
3300 GOTO 3000
4000 REM *** CODE ***
4005 BEEP
4010 IF VPEEK(7+XT*8+YT*256)<>RODS THEN 5000
4020 LET CODE=RND(8999)+1000
4030 PRINT @ 25,20,CODE
4040 FOR A=0 TO 100:NEXT A
4050 PRINT @ 25,20,"    "
4060 PRINT
4070 INPUT A
4080 IF A=CODE THEN PRINT @ 25,20,"     ":ELSE GOTO 5000
4090 PRINT@ 0,22,"     "
4100 IF RODS =9  THEN GOTO 6000
4110 LET RODS=RODS+1:RETURN
```

```
 5000 FORA=1TO 20
 5010 BCOL1:TCOL15,9
 5020 PRINT@ 10,10,"MELT DOWN"
 5030 BCOL9
 5035 CLS
 5040 PSG6,31:PSG7,71
 5050 PSG8,16:PSG9,16:PSG10,16
 5060 PSG12,100
 5070 PSG13,0
 5080 NEXT A
 5090 GOTO 7380
 6000 REM *** WINNER ***
 6010 FOR Z=0 TO 25
 6020 CLS40:TCOL15,RND(15):PRINT @ 5,10;"YOU HAVE SAVED THE
      WORLD":PRINT @ 5,12;"IN "T" HALF LIFE PERIODS"
 6030 NEXT Z
 6040 RESTORE
```

YOUR COPY OF WORLD by Martin Page goes in here renumbered 7000,10
If you have not typed it in yet WHY NOT!

```
10000 REM***INSTRUCTIONS***
10010 BCOL6:TCOL15,1:CLS40
10015 PRINT@12,2;"ATOMIC RESCUE"
10020 PRINT@5,10;"HELLO SUPER-HERO"
10030 PRINT@5,12;"DO YOU REQUIRE INSTRUCTIONS?"
10040 PRINT@32,12: A=INCH
10050 IF A=89 OR A=121 THEN 10100:ELSE RETURN
10100 CLS40
10110 PRINT"YOU ARE THE ONLY PERSON WHO CAN SAVE THEWORLD.THE  ATOMIC
      REACTOR AT SELASCALE ISIN A CRITICAL STATE."
10120 PRINT" UNLESS YOU CAN COOL THE ATOMIC RODS A   MELT DOWN  WILL
      OCCUR.THE RODS MUST BE   COOLED IN THE SET ORDER 0 TO 8."
10130 PRINT"IT IS TOO DANGEROUS TO ENTER THE REACTOR.YOU MUST USE THE
      REMOTE CONTROL COOLER."
10135 PRINT"THE  CONTROL  KEYS  ARE..W-UP,S-DOWN,K-LEFT,L-RIGHT.YOU
      CANNOT BACKTRACK OR CROSS  THE CONTROL WIRE"
10140 PRINT"THE  REACTOR RODS MUST BE COOLED IN ORDER.ON  COOLING  A
      HALF LIFE CODE NEEDS TO BEENTERED INTO THE COMPUTER"
10150 PRINT"IT MUST MATCH THE RODS CODE."
10160 PRINT"TIME  IS CRITICAL YOU HAVE  1000  HALF-LIFEPERIODS--GOOD
      LUCK!"
10165 PRINT"DONT DELAY PRESS A KEY"
10170 A=INCH
10180 RETURN
11000 REM PRETTY TUNE
11010 VOICE 0,31,10,3,90,50
11020 A$="C5C5-A5C8-A5-F8C5D7C6C5C5-A5-bB7-bB7-G5-E8-bB5C7-bB6-A5-A5-
      bB5C7-A5-F6C5D7C6C5C5-A5C7C7-bB7-G7-F9"
11030 TEMPO 5
11040 MUSIC "V0"+A$,"R","R"
11050 PSG7,&7F
11060 RETURN
```

DOS 80 V1.0

The new DOS should be available soon and included on the disc are an 80 column Basic and a new BACKUP to allow formatting of double sided 80 track drives. You can also configure the DOS to suit the drives you are using. It comes set up as single side 40 track for the internal drives and double side 80 track for the external drives, but this can be changed.
Expected to sell at around £20 it will be well worth the investment.

WALLPAPER

by Peter Moon

There is an extremely simple programme which produces amazingly complex patterns resembling wallpaper, Persian carpets, and tessellated pavements. You can find the details in July's issue of "Scientific American", but essentially, what the programme does is to take the x and y co-ordinates of all the points in a gridded square, calculate for each point x^2 + y^2, truncate the result to integer, and plot the point if the result is even, blank if odd. So that you can rush off and make some wallpaper, here is the simplest version of the programme, without any frills or refinements. Try putting SIDE = 20 for a start.

```
  5 REM WALL1
 10 REM By Peter Moon for UKEUG 1.12.86
 20 INPUT SIDE
 30 CLS
 note:- put in lines here to PRINT @
        so as to get a note at the
        R.H.S. of the screen of the
        parameters used. i.e. SIDE
 40 FOR I=0 TO 191
 50 FOR J=0 TO 191
 60 X=SIDE*I/191
 70 Y=SIDE*J/191
 80 Z=X*X+Y*Y
 90 IF Z MOD 2=0 THEN 100 ELSE 110
100 PLOT I,J
110 NEXT J
120 NEXT I
```

If you want to print your patterns, and you have Chris Giles's screen dump routine (Issue 1, p. 24) on your system tracks, add

```
 15 CLEAR &E26F
130 CALL &E270
```

and remember to have the printer switched on ready.
(note ED for the graphic routines later in this issue:-

```
 15 CLEAR &9FFF
125 LOAD"GDMP.OBJ":REM GDMP1, GDMP2, GDMP3 or GDMP4 available
130 CALL &A000 )
```

191 is because you have 192 pixels in the height of the screen. The variable SIDE is of course the length of the side of the square in the xy plane which you specify for operations. The smaller the size of SIDE the larger the magnification of the pattern. Increase SIDE and you are, in effect, standing further away from the pattern. All sorts of elements of larger scale patterns, superimposed on the basic one, appear in shadowy outline as you back away.

In the programme above, your gridded square has its lower left corner at the origin (x = y = 0) but by inputting CX and CY for the co-ordinates of the lower left corner you can move this window where you like. Line 60 then becomes:

60 X = CX + SIDE*I/191/ and 70 similar.

You can try other functions like x^2 - y^2, and MOD 3 instead of MOD 2. All seem determined to produce some regularity of pattern.

Compared with the creeping progress of the Mandelbrot programme, which iterates up to 100 times before deciding whether to plot a point, this one runs like a train.

Here is a more flexible version of the programme incorporating some refinements. I've used BBC BASIC because its TIME function is more useful, and its PRINT @ feature enables you to annotate your printouts.

```
 10 REM WALL2
 20 REM BBC BASIC
 30 REM By Peter Moon for UKEUG 1.12.86.
 40 HIMEM=&E26F
 50 INPUT "SIDE";SIDE
 60 INPUT "CORNER X, CORNER Y";CX,CY
 70 INPUT "MODULO";F
 80 CLS
 85 TIME=0
 90 FOR I=0 TO 767 STEP 4
100 FOR J=0 TO 767 STEP 4
110 X=CX+SIDE*I/767
120 Y=CY+SIDE*J/767
130 Z=X*X+Y*Y
140 IF Z MOD F =0 THEN 150 ELSE 160
150 PLOT 69,I,J
160 NEXT J
170 NEXT I
180 T=INT(TIME/6000)
190 PRINT TAB(76,1);"WALL2"
195 PRINT TAB(76,3);"FUNCTION"
200 PRINT TAB(76,6);"CX";TAB(76,7);CX
210 PRINT TAB(76,9);"CY";TAB(76,10);CY
220 PRINT TAB(76,12);"SIDE";TAB(76,13);SIDE
225 PRINT TAB(76,15);"MOD";TAB(76,16);F
230 PRINT TAB(76,18);"TIME";TAB(76,19);T
240 PRINT TAB(76,20);"MINS"
250 CALL &E270
260 *OPT2
270 FOR L=1 TO 6: PRINT: NEXT L
280 *OPT0
```

You can modify the programme, if you're adventurous, to execute a whole series of runs with different SIDEs and Fs. You can even incorporate EVAL so that it tries a succession of functions at line 160, reading them from a DATA statement. These features will make it print a succession of patterns while you're out at the pub.
I'll settle for half of any royalties you get on carpet designs (no cheques please).

GRAPHIC DUMPS

Dave Salvage has taken the straightforward screen dump program and modified it very effectivly to produce four different routines. The first is the original screen dump. The second is a double size screen dump. The third is double size, but sideways and the fourth is sideways but adjusted to produce circles on paper from circles on screen, hence the 3/4 ratio of expansion.
Dave wrote them on ASSEM the Glentop assembler, but they are produced here for ZEN. The line numbers are left out but as both ASSEM and ZEN insert their own line nos. this should cause no problems.

```
   ;GRAPHICS DUMP ROUTINE
   ;BY DAVE SALVAGE
   ;FOR THE EPSON LX-80 PRINTER
   ;SHOULD WORK ON MOST EPSON PRINTERS
ORG 0A000H          ORG 0A000H          ORG 0A000H              ORG 0A000H
LOAD 0A000H         LOAD 0A000H         LOAD 0A000H             LOAD 0A000H
ORGX: EQU 0FB9AH    ORGX: EQU 0FB9AH    ORGX: EQU 0FB9AH        ORGX: EQU 0FB9AH   ;X CO-ORG IN SCRATCH PAD
ORGY: EQU 0FB9CH    ORGY: EQU 0FB9CH    ORGY: EQU 0FB9CH        ORGY: EQU 0FB9CH   ;Y CO-ORD IN SCRATCH PAD
AUXREG: EQU 20H     AUXREG: EQU 20H     AUXREG: EQU 20H         AUXREG: EQU 20H    ;PRINTER STATUS PORT
IN A,(AUXREG)       IN A,(AUXREG)       IN A,(AUXREG)           IN A,(AUXREG)      ;READ PRINTER STATUS
AND 1CH             AND 1CH             AND 1CH                 AND 1CH            ;MASK NON PRINTER BITS
CP 10H              CP 10H              CP 10H                  CP 10H             ;SEE IF PRINTER READY
RET     NZ                    RET   NZ                   RET    NZ
```

```
    RET NZ                 ;ABORT IF NOT READY
;                      ;                        ;                              ;
LD HL,(ORGX)           LD HL,(ORGX)             LD HL,(ORGX)                   LD HL,(ORGX)           ;STORE CURRENT ORIGIN ON STACK
PUSH HL                PUSH HL                  PUSH HL                        PUSH HL
LD HL,(ORGY)           LD HL,(ORGY)             LD HL,(ORGY)                   LD HL,(ORGY)
PUSH HL                PUSH HL                  PUSH HL                        PUSH HL
;                      ;                        ;                              ;
LD HL,191              LD HL,191            ;SET TEMP ORIGIN TO TOP LEFT OF SCREEN
LD (ORGY),HL           LD (ORGY),HL             ;                              ;
LD HL,0                LD HL,0                  ;                              ;
LD (ORGX),HL           LD (ORGX),HL             ;                              ;
;                      ;                        ;                              ;
PUSH HL                PUSH HL              ;SET X & Y POINTERS IN INDEX REGISTERS
POP IX                 POP IX                   ;                              ;
PUSH HL                PUSH HL                  ;                              ;
POP IY                 POP IY                   ;                              ;
;                      ;                        ;                              ;
GDUMP0: PUSH IX        GDUMP0: PUSH IX       ;SAVE XPOINTER ON STACK
;                      ;                        ;                              ;
;                      ;                        LD BC,0                        LD BC,0                ;HORIZ OFFSET FOR VRAM ADDRESS
;                      ;                        GDUMP0:PUSH BC--               GDUMP0:PUSH BC         ;STORE HORIZ OFFSET ON STACK
;                      ;                        ;                              ;
;                      ;                        ;                              LD B,4                 ;SET COUNTER FOR X-AXIS ENLARGEMENT
;                      ;                        ;                              GDUMP4:PUSH BC         ;STORE ON STACK
;                      ;                        ;                              LD (0AF00H),BC         ;STORE IN MEMORY
        THIS IS LINE    111
LD B,9                 LD B,9                   LD B,9                         LD B,9                 ;SET PARAMETER COUNT
LD HL,PARAMS           LD HL,PARAMS             LD HL,PARAMS                   LD HL,PARAMS           ;POINT TO PARAMETER TABLE
GDUMP1: LD A,(HL)      GDUMP1: LD A,(HL)        GDUMP1: LD A,(HL)              GDUMP1: LD A,(HL)      ;GET PARAMETER
RST 8                  RST 8                    RST 8                          RST 8                  ;MCAL ZPOUT
DEFB 9FH               DEFB 9FH                 DEFB 9FH                       DEFB 9FH               ;SEND TO PRINTER
INC HL                 INC HL                   INC HL                         INC HL                 ;INCREMENT POINTER
DJNZ GDUMP1            DJNZ GDUMP1              DJNZ GDUMP1                    DJNZ GDUMP1            ;UNTIL ALL PARAMETERS SENT
;                      ;                        ;                              ;
;                      ;                        ;                              POP DE                 ;RESTORE X-AXIS ENLARGEMENT
;                      ;                        ;                              ;COUNTER
;                      ;                        POP BC                         POP BC                 ;RESTORE HORIZ OFFSET
;                      ;                        PUSH BC                        PUSH BC                ;COPY HORIZ TO STACK
;                      ;                        ;                              PUSH DE                ;COPY X-AXIS ENLARGMENT CTR TO ST
;                      ;                        LD DE,1800H                    LD DE,1800H            ;POINT TO TOP OF VRAM PAT.
                                                                                                      ;GEN. TABLE (BOTTOM OF SCREEN)
;                      ;                        GDUMP2:LD HL,0FF00H            GDUMP2:LD HL,0FF00H    ;0FF00 = -256 IN DECIMAL
;                      ;                        ADD HL,DE                      ADD HL,DE              ;SET VRAM ADDRESS TO SCREEN LINE A
;                      ;                        PUSH HL                        PUSH HL                ;COPY TO DE REGISTER
;                      ;                        POP DE                         POP DE
;                      ;                        PUSH DE                        PUSH DE                ;COPY TO STACK
;                      ;                        LD HL,8                        LD HL,8                ;SET BYTE OFFSET FOR VRAM ADDRESS
;                      ;                        GDUMP3:DEC HL                  GDUMP3:DEC HL          ;DECREMENT BYTE OFFSET
;                      ;                        PUSH HL                        PUSH HL                ;STORE TO STACK
;                      ;                        ADD HL,DE                      ADD HL,DE              ;ADD VRAM LINE ADDRESS TO BYTE OFFSET
;                      ;                        ADD HL,BC                      ADD HL,BC              ;ADD HORIZ OFFSET TO ABOVE ADDRESS
;                      ;                        PUSH BC                        PUSH BC                ;STORE HORIZ OFFSET TO STACK
;                      ;                        PUSH HL                        PUSH HL                ;COPY VRAM ADDRESS TO BC
;                      ;                        POP BC                         POP BC
;                      ;                        RST 8                          RST 8                  ;MCAL ZVRIN
;                      ;                        DEFB 0C2H                      DEFB 0C2H              ;GET VALUE AT VRAM ADDRESS
GDUMP2: PUSH IY        GDUMP2: PUSH IY       ;SAVE YPOINTER ON STACK           ;
LD C,10                LD C,1                ;SET BIT 0                        ;
;                      GDUMP5: LD B,2        ;SET COUNTER FOR SIZEx2           ;
GDUMP3: PUSH BC        GDUMP3: PUSH BC       ;STORE (BC) ON STACK              ;
RST 8                  RST 8                 ;MCAL ZPNTXY                      ;
DEFB 0C7H              DEFB 0C7H             ;TEST IX,IY PIXEL                 ;
POP BC                 POP BC                ;RESTORE (BC)                     ;
```

Due to a miscalculation this is an extra, unlisted page. I had thought of printing the usual computer manual get out of this line ie.

THIS PAGE IS INTENTIONALLY BLANK

But I thought better of it, so here are a few extra BITS for you.

MAILBOX

This is a new section which will appear when ever it has been updated . It is a directory of our members who are on TELECOM GOLD or PRESTEL with their MAILBOX numbers .If YOU are on either of these systems and your name is not here then let us know so that it will be included next time.

NAME	GOLD	PRESTEL
KEITH STOKES	MAG90044	
DAVE WEST	MAG95189	219994851
JEAN SKILLERN		903725931
JOHN RAPER		219994920
LESLIE WARNER		919999523
KANDY BATES		919999879
PAUL HARRISON		914137847
TONY REYNOLDS		919994227
CHARLES GREGORY		219995080

This is a very short basic program to show what happens when you FILL an area which has already been FILLED.

```
 0 CLS40
20 ELLIPSE 70,80,50
30 FILL 70,80
40 ELLIPSE 180,80,50
50 GCOL (RND(15))
60 FILL 180,80
70 GOTO 20
```

FORTHCOMING

In future issues we have articles on the following subjects

A SUBMIT FOR CRYSTAL DOS

MACHINE CALLS (MCAL) WHAT THEY ARE AND HOW TO USE THEM

REVIEWS ON AIRWOLF AND ELITE ,YES IT IS AVAILABLE FOR THE EINSTEIN.

If you have any subject you would like to see articles about or any articles about any subject then let Chris know and we will be only to pleased to include them.

```
  JR Z,GDUMP4         JR Z,GDUMP4       ;IF PIXEL ON                    ;
SCF                  SCF              ;SET CARRY FLAG                 ;
GDUMP4: RL C         GDUMP4: RL C     ;ROTATE CARRY INTO LSB POSITION
;                    DJNZ GDUMP3      ;DECREMENT SIZE COUNTER         ;
DEC IY               DEC IY           ;DECREMENT TEMP Y POINTER       ;
JR NC,GDUMP3         JR NC,GDUMP5     ;UNTIL C>255                    ;
;                    ;                    ;                           ;
LD A,C               LD A,C           ;TRANSFER BYTE TO ACCUMULATOR   ;
RST 8                RST 8            ;MCAL ZPOUT                     ;
DEFB 9FH             DEFB 9FH             ;                           ;
;                    RST 8            ;MCAL ZPOUT                     ;
;                    DEFB 9FH             ;                           ;
POP IY               POP IY           ;RESTORE Y COUNTER              ;
INC IX               INC IX           ;INCREMENT X COUNTER            ;
PUSH IX              PUSH IX          ;TRANSFER TO AF;                ;
POP AF               POP AF               ;                           ;
OR A                 OR A             ;CLEAR FLAGS IN F REG           ;
JR Z,GDUMP2          JR Z,GDUMP2      ;IF COUNTER> 255                ;
POP IX               POP IX           ;RESTORE COUNTER TO 0           ;
;                    LD BC,0FFFCH     ;&FFFCH = -4 IN DECIMAL         ;
LD BC,0FFF8H                        ;0FFF8H =-8 IN DECIMAL;           ;
ADD IY,BC; BY 8      ADD IY,BC        ;DECREMENT COUNTER BY 4         ;
LD A,0AH             LD A,0AH             ;                           ;
RST 8                RST 8                RST 8        ;MCAL ZPOUT    ;
DEFB 9FH             DEFB 9FH             DEFB 9FH                    ;
PUSH IY              PUSH IY          ;COPY COUNTER TO HL REG         ;
;                    ;                    ;                           CALL EXPAND
;                    ;                    ;                           ;
POP HL               POP HL               ;                           ;
LD BC,191            LD BC,191            ;                           ;
ADD HL,BC            ADD HL,BC            ;                           ;
JR C,GDUMP0          JR C,GDUMP0      ;IF Y COUNTER <-191             ;
;                    ;                    POP BC                      POP BC            ;RESTORE HORIZ OFFSET
;                    ;                    POP HL                      POP HL            ;RESTORE BYTE OFFSET
;                    ;                    LD A,0                      LD A,0
;                    ;                    OR L                        OR L
;                    ;                    JR NZ,GDUMP3                JR NZ,GDUMP3      ;IF BYTE OFFSET =0
;                    ;                    POP DE                      POP DE            ;RESTORE VRAM LINE ADDRESS
;                    ;                    LD A,0                      LD A,0
;                    ;                    OR D                        OR D
;                    ;                    JR NZ,GDUMP2                JR NZ,GDUMP2      ;IF MSB OF LINE ADDRESS=ZERO
;                    ;                    LD A,0                      LD A,0            ;CHECK LSB OF LINE ADDRESS
;                    ;                    OR E                        OR E
;                    ;                    JR NZ,GDUMP2                JR NZ,GDUMP2      ;IF LINE ADDRESS = ZERO
;                    ;                    LD A,0AH                    LD A,0AH          ;LINE FEED
;                    ;                    RST 8                       RST 8             ;MCAL ZPOUT
;                    ;                    DEFB 9FH                    DEFB 9FH
;                    ;                    ;                           POP BC            ;RESTORE X-AXIS ENLARGEMENT COUNTE
;                    ;                    ;                           DJNZ GDUMP4       ;IF ENLARGEMENT COUNTER=ZERO
;                    ;                    POP BC                      POP BC            ;RESTORE HORIZ OFFSET
;                    ;                    LD HL,8                     LD HL,8
;                    ;                    ADD HL,BC                   ADD HL,BC         ;INCREMENT HORIZ OFFSET
;                    ;                    PUSH HL                     PUSH HL           ;COPY TO BC REG
;                    ;                    POP BC                      POP BC
;                    ;                    LD A,0                      LD A,0
;                    ;                    OR B                        OR B
;                    ;                    JR Z,GDUMP0                 JR Z,GDUMP0       ;IF HORIZ OFFSET > 255
;                    ;                    LD A,1BH                    LD A,1BH          ;RESET PRINTER LINE SPACING
LD A,1BH             LD A,1BH             RST 8                       RST 8
RST 8                RST 8                DEFB 9FH                    DEFB 9FH          ;MCAL ZPOUT
DEFB 9FH             DEFB 9FH             LD A,40H                    LD A,40H          ;RESET PRINTER TO POWER ON STATUS
LD A,40H             LD A,40H             RST 8                       RST 8
RST 8                RST 8
DEFB      9FH             DEFB      9FH             DEFB      9FH
```

```
   DEFB 9FH           ;MCAL ZPOUT
;                     ;                     ;
POP HL                POP HL                POP HL
LD (ORGY),HL          LD (ORGY),HL          LD (ORGY),HL
POP HL                POP HL                POP HL
LD (ORGX),HL          LD (ORGX),HL          LD (ORGX),HL
RET                   RET                   RET
;                     ;                     ;
PARAMS:DEFB 0DH       PARAMS:DEFB 0DH       PARAMS:DEFB 0DH
DEFB 1BH              DEFB 1BH;ESC          DEFB 1BH;ESC
DEFB 41H              DEFB 41H              DEFB 41H
DEFB 08H              DEFB 08H              DEFB 08H
DEFB 1BH              DEFB 1BH       -      DEFB 1BH
     THIS IS LINE          222
DEFB 2AH              DEFB 2AH              DEFB 2AH
DEFB 05H              DEFB 05H              DEFB 05H
DEFB 00H              DEFB 00H              DEFB 0C0H
DEFB 01H              DEFB 02H              DEFB 00H
END                   END                   END
```

I apologise straight away for the reduced print size, which is against our policy, but it really was the only way to get all the listings in in the one issue. And I feel that it was only right to put them in the one issue.

A few notes. The leftmost column will give a graphic dump of the screen pixel for pixel. The next column will give a double size display, i.e.2 pixels wide by 2 pixels high for each pixel on the screen. The next column will give a sideways, pixel by pixel dump and the fourth column will give a dump that plots sideways with a magnification factor of 3 and 4. Hope this is clear. They can all be saved as NAME.OBJ files to be used from BASIC and are designed to reside at A000 Hex. So from BASIC a CLEAR &A000 is used early in the program so as to allow space and each routine can be called at will, thus:-

```
1000 I=INCH:IF I<49 OR I>52 THEN 1000
1010 IF I=49 THEN LOAD"GDMP1.OBJ":REM I="1"
1020 IF I=50 THEN LOAD"GDMP2.OBJ":REM I="2"
1030 IF I=51 THEN LOAD"GDMPS.OBJ":REM I="3"
1040 IF I=52 THEN LOAD"GDMP34.OBJ":REM I="4"
1050 CALL &A000:REM CALL M/C ROUTINE JUST LOADED
1060 RETURN:REM GO BACK TO ORIGINAL PROGRAM
```

NOTES ON CONVERSION

The value loaded into the B register in line 111 is the number of parameters needed to be passed to the printer. The value in line 222 needs to be changed to 4BH for the Tatung TP 80 (or 100) and the next line should be left out for the Tatung Printers.

If anyone knows the relevant values for other printers, do let us know so that we can pass the information on.

Elsewhere in this issue you should find an article on WALLPAPER, which requires a graphics dump routine, these could help produce some very interesting results.

```
;
POP HL              ;RESTORE ORIGIN AT START OF ROUTINE
LD (ORGY),HL
POP HL
LD (ORGX),HL
RET
;
PARAMS:DEFB 0DH    ;CARRIAGE RETURN
DEFB 1BH           ;ESC
DEFB 41H           ;n/72 INCH LINE SPACING
DEFB 08H           ;n=8
DEFB 1BH           ;ESC

DEFB 2AH           ;BIT IMAGE SELECT
DEFB 05H           ;MODE 5 - HORIZ=VERT SPACE
DEFB 040           ;NO. OF DOTS MOD 256
DEFB 02H           ;INT(NO. DOTS/256)
;
RET
;SHOULD NEVER GET HERE
;
;
;
EXPAND:PUSH BC      ;STORE VRAM ADDRESS ON
                    ;STACK
PUSH AF             ;STORE DATA AND FLAGS ON STACK
LD BC,(0AF00H)      ;LOAD X-AXIS ENLARGEMENT
                    ;COUNTER FROM MEMORY
DJNZ GDUMP5         ;IF COUNTER = ZERO
CALL FOUR           ;EXECUTE EXPANSION FOR FOURTH
                    ;PASSAGE
RLCA                ;ROTATE MSB TO LSB
CALL ONE
JR PRINT
GDUMP5:DJNZ GDUMP6  ;IF COUNTER = ZERO
CALL THREE          ;EXECUTE EXPANSION FOR THIRD
                    ;PASSAGE
RLCA                ;ROTATE MSB TO LSB
CALL ONE
JR PRINT
GDUMP6:DJNZ GDUMP7  ;IF COUNTER = ZERO
CALL TWO            ;EXECUTE EXPANSION FOR SECOND
                    ;PASSAGE
RLCA                ;ROTATE MSB TO LSB
CALL ONE
JR PRINT
GDUMP7:CALL ONE     ;EXECUTE EXPANSION FOR
                    ;FIRST PASSAGE
RLCA                ;ROTATE MSB TO LSB
CALL ONE
;
PRINT:LD A,C        ;TRANSFER BYTE TO A REG
RST 8               ;MCAL ZPOUT 3 TIMES
DEFB 9FH
RST 8
DEFB 9FH
RST 8
DEFB 9FH
POP A               ;RESTORE DATA AND FLAGS
POP BC              ;RESTORE VRAM ADDRESS
RET
;
```

Before I forget, the last column has all the "REM" statements and these apply to all the columns to the left that have an instruction in them.

```
 FOUR:RLCA
RLCA
THREE:RLCA
RLCA
TWO:RLCA
RLCA
ONE:RLCA                  ;ROTATE MSB TO LSB AND CRY FLAG
RL C                      ;ROTATE CARRY FLAG TO LSB C REG
RRCA                      ;ROTATE LSB TO MSB AND CARRY FLAG
                          ;(THE SAME BIT AS BEFORE)
RL C                      ;ROTATE CARRY FLAG TO LSB C REG
RLCA                      ;ROTATE MSB TO LSB AND CARRY FLAG
                          ;(THE SAME BIT AS BEFORE)
RL C                      ;ROTATE CARRY FLAG TO LSB C REG
RRCA                      ;ROTATE LSB TO MSB AND CARRY FLAG
                          ;(THE SAME BIT AS BEFORE)
RL C                      ;ROTATE CARRY FLAG TO LSB C REG
RET                                  FLAG TO LSB C REG
RET
END
```

ELECTRONIC BOOK

The first thing I did when I got the book sort of running was to use it to control a game. The only one I could think of that was readily to hand at the time was GALRAD, published in Volume 1 Number 2 of the EINSTEIN USER. The changes I made are as follows:-

```
860 A$="PRESS 11 TO CONTINUE.":PRINT@4,22;:GOSUB1020
870 GOSUB 2000
990 GOSUB 2000
2000 A=INT(ADC(0)/40)
2050 IF A>0 THEN GOTO2090
2060 B=INT(ADC(1)/40)
2070 IF B>0 THEN GOTO 2200
2080 A$="":RETURN
2090 IF A=1 THEN A$="Z":RETURN
2100 IF A=2 THEN A$="/":RETURN
2110 IF A=3 THEN A$="C":RETURN
2150 RETURN
2200 IF B=4 THEN A$=" ":RETURN
2210 RETURN
```

And surprise surprise it worked. I was well chuffed. Spurred on to greater things I thought to myself, 'what about something useful??', PICPEN! So, that's what I tackled next. Here follows the changes:-

```
1510 IF INT(ADC(1)/40) <> 2 THEN 1510
1660 GOSUB 6000
1670 IF C<1 OR C>5 THEN 1660
1920 IF INT(ADC(1)/40) <> 2 THEN 1920
2180 IF INT(ADC(1)/40) <> 2 THEN 2180
2440 IF INT(ADC(1)/40) <> 2 THEN 2440
2610 IF INT(ADC(1)/40) <> 2 THEN 2610
2660 GOSUB 3620:DIS=1:GOTO2820
2720 GOSUB 6000:IF C=0 THEN GOTO 2720
2730 IF C=4 THEN X=X-DIS:GOTO 2800
2740 IF C=6 THEN X=X+DIS:GOTO 2800
2750 IF C=8 THEN Y=Y-DIS:GOTO 2800
2760 IF C=2 THEN Y=Y+DIS:GOTO 2800
2790 IF C<>0 THEN GOSUB 4910:C=0
2920 GOSUB 6000
3020 IF C=11 THEN FOR W=0 TO 50:NEXT W
3030 GOSUB6000
3040 IF C=12 THEN GCOL INKCOL,0:PEN=PENS:SPRITE OFF:GOTO3070
```

```
3050 IF C=11 THEN INKCOL=INKCOL+1:IF INKCOL>15 THEN INKCOL=1
3080 X=XS:Y=YS:C=0:RETURN
3180 C=0:RETURN
4780 GOSUB2670:GOSUB2890:IF C<>12 THEN GOTO 4780
4910 IF C=5THEN BEEP DIS=DIS*2:IF DIS>8THEN DIS=1:FOR C=0TO30:NEXT:
     RETURN
4920
4930
4940
4950 IF C=3 THEN BEEP:GOSUB 3090:RETURN
4960 IF C=10 THEN BEEP:GOSUB2940:RETURN
4970 IF C=11 THEN BEEP:GOTO5100
4980 IF C=7 THEN BEEP:GOSUB4730:RETURN
4990
5000 IF C=9 THEN BEEP:DRAW X,Y TO X,Y:DR=-1:SPACE=-1:GOSUB3550:RETURN
5010 IF KBD=27 THEN BEEP:POP:RETURN
5020 IF C=1 THEN BEEP:GOSUB3190:RETURN
5030
5040 IF C=12 THEN BEEP:DR=2:DIS=2:MES$=UMESS$:GOSUB 3550:RETURN
5100 C=0:IF DR=0 THEN DR=-1:DRAW X,Y TO X,Y:SPACE=-1:GOTO3550
5110 DR=0:SPACE=-1:GOTO3550
6000 C=0
6010 A=INT(ADC(0)/40)
6020 IF A>0 THEN GOTO 6090
6030 B=INT(ADC(1)/40)
6040 IF B>0 THEN GOTO 6160
6050 C=0:RETURN
6090 IF A=6 THEN C=1:RETURN
6100 IF A=5 THEN C=2:RETURN
6110 IF A=4 THEN C=3:RETURN
6120 IF A=3 THEN C=6:RETURN
6130 IF A=2 THEN C=5:RETURN
6140 IF A=1 THEN C=4:RETURN
6150 RETURN
6160 IF B=6 THEN C=7:RETURN
6170 IF B=5 THEN C=8:RETURN
6180 IF B=4 THEN C=9:RETURN
6190 IF B=3 THEN C=12:RETURN
6200 IF B=2 THEN C=11:RETURN
6210 IF B=1 THEN C=10:RETURN
6220 RETURN
```

If the above lines are typed in after loading PICPEN then the ELECTRONIC BOOK will replace the keyboard and joystick for most functions.

Pad No.	Pad No.	Pad No.
1 = Fill	2 = Up	3 = Change Background
4 = Left	5 = Change Speed	6 = Right
7 = Circle	8 = Down	9 = Draw To
10 = Ink Colour	11 = Space and Ink Start/Stop	
12 = Enter and Rub Out Toggle		

There is a bit of 'Key Bounce' and it doesn't like two pads being pressed at the same time, but I think it definately has potential. Well there's your first ADD ON. Now do something with it!!!!

BACK ISSUES

We are assembling all the back issues on disk and hope to have these available soon. As soon as they are we will let you know. They will be available as a text file to print out on your printer at will or display on the screen. It should be possible to extract the programs from the files to save the bother of typing them in.

ALARMING

Use your Einstein as an alarm. Incorporate this into other routines to remind you that you have an appointment with dinner this evening. Written in XBAS by Dennis C. Hain

```
  50 REM * REMINDER ALARM * BY DENNIS C HAIN *
  80 TCOL3,1
  90 CLS:REM  SET CLOCK
 100 PRINT"Enter time: HHMMSS (even No. of secs)."
 110 INPUT"";T$
 120 TI$=T$
 150 CLS:GOSUB1000
 200 GOTO2000
1000 N=1:TST=1
1010 INPUT "Reminder Time (HHMM) ";RT$(N)
1020 INPUT "Message: ";M$(N)?:CLS
1030 INPUT "Any More (Y/N)? "+Y$
1040 IFY$<>"Y" THEN RETURN
1050 N=N+1:GOTO1010
1999 REM TEST FOR TIME CHANGE
2000 MIN$=MID$(TI$,4,1):CLS:PRINT@0,0;LEFT$(TI$,2);".";MID$(TI$,3,2)
2010 IF MIN$<>MID$(TI$,4,1)THEN GOTO 3000:ELSE 2010
2999 REM ....TEST FOR REMINDER TIME
3000 IF RT$(TST)<>LEFT$(TI$,4)THEN2000
3005 PRINT@0,0;LEFT$(TI$,2);".";MID$(TI$,3,2)
3010 PRINT@3,10;LEFT$(TI$,2)+".";MID$(TI$,3,2)
3020 PRINT@10,10;M$(TST):BEEP30
3025 PRIJNT@10,20;"Press any key to continue"
3030 Z$=INCH$:CLS:IF TST=N THEN 4000
3050 TST=TST+1:GOTO2000
3999 REM EXIT OR RESET ALARM
4000 PRINT "   That was the last entry,":PRINT:PRINT"Do you wish  to
     set another time? (Y/N)";
4010 INPUT"";Y$:IFY$="Y" THEN RUN 150
4020 CLS:PRINT"END OF PROGRAM":END
```

HINTS AND TIPS

Whenever I pick up a program to modify I start off with good intentions to make notes as I go along!! After the first few alterations it goes by the board and 'I'll make notes when I've finished this bit' sets in. The result is that I end up with a program that has all the modifications that I want in it but I can't for the life of me remember where they are.

The easiest way to compare two such programs, i.e. the original and the modified version, is to clear the machine with a NEW.

Then LOAD"ORIGINAL"
Hold ,LAST LINE NO
LOAD"MODIFIED"
MERGE

This will then give a merged listing with TWO line numbers the same for each of the original ones. The first will be the original version and the next will be the modified version thus:-

```
10 REM This is my test program
10 REM This is my test program
20 PRINT"ORIGINAL VERSION":A$="O"
20 PRINT"MODIFIED VERSION":A$="M"
30 IF A$="O" THEN 50
30 IF A$="M" THEN 50
40 STOP
40 STOP
50 PRINT "THIS IS THE ";A$;"VERSION"
50 PRINT "THIS IS THE ";A$;"VERSION"
```

ALLSORTS NO 2

In the first ALLSORTS we looked at the Bubble sort. Good for general purpose but slow when there are large numbers of items to sort. This article will concentrate on speeding up the Bubble sort.
First some facts and figures. The number of comparasons done in a bubble sort is N*(N-1)/2, where N is the number of items to be sorted, which explains why a large number of items to be sorted takes a long time and it will still take this time even if only one item is out of order. Type in this new listing with the DATA statements from the previous listing.

```
 10 INPUT"NO OF ENTRIES ";N
 20 DIM ENTRY$(N)
 30 FOR ENT=1 TO N
 40 READ ENTRY$(ENT)
 50 NEXT
 60 CLS
 70 FOR ENT=1 TO N
 80 PRINT ENTRY$(ENT);" ";
 90 NEXT
100 TI$="000000"
110 PRINT:PRINT"BUBBLE SORT";TI$
120 SWPCNT=0:COMP=0:SWPFLG=0
130 BOTT=N
140 TP=1
150 PNTR=BOTT
160 REM:SWPPAS=0
170 IF ENTRY$(PNTR)<ENTRY$(PNTR-1)THEN SWAP ENTRY$(PNTR),
    ENTRY$(PNTR-1):SWPCNT=SWPCNT+1:SWPPAS=SWPPAS+1:REM:SWPFLG=PNTR
180 COMP=COMP+1
190 PNTR=PNTR-1
200 IF PNTR>TP THEN 170
210 REM:IF SWPPAS=0 THEN 300
220 TP=TP+1
230 REM:IF ENTRY$(PNTR)>ENTRY$(PNTR+1)THEN SWAP ENTRY$(PNTR),
    ENTRY$(PNTR+1):SWPCNT=SWPCNT+1:SWPPAS=SWPPAS+1:REM:SWPFLG=PNTR
240 REM:COMP=COMP+1
250 REM:PNTR=PNTR+1
260 REM:IF PNTR<BOTT THEN 230
270 REM:IF SWPPAS=0 THEN 300
280 REM:BOTT=BOTT-1
290 IF BOTT>TP THEN 150
300 T$=TI$
310 PRINTCHR$(18):PRINT"SORTED ";N;" ITEMS IN ";TI$
320 PRINT"COMPARASONS = ";COMP
330 PRINT"SWAPS = ";SWPCNT:PRINTCHR$(19)
340 FOR ENT=1 TO N
350 PRINTENTRY$(ENT);" ";
360 NEXT
```

Try running the program WITH the REM statements with different numbers of entries. Then GOTO 100 so that you can compare a random list with a sorted list. (The GOTO 100 will 'sort' the already sorted list). Two things should become apparent, the number of comparasons does not change between a sorted list and an unsorted list, and the time increases dramatically with the number of items to be sorted.

No to sort	Time		No Comparasons		No Swaps	
	sorted	Un	sorted	Un	sorted	Unsorted
10	1	1	45	45	27	27
20	3	5	190	190	0	97
40	15	20	780	780	0	383
80	101	122	3160	3160	0	1666

So if we can find a way to reduce the number of comparasons then we can speed the sort up. On way of doing this is to count up the number of SWAPS done on each pass and if none are done then the list is in order and the sort can be regarded as finished.
Take the listing and remove the REM statements from lines 160 and 210. RUN it again as before and 'GOTO 10' as before.

No to sort	Time		No Comparasons		No Swaps	
	sorted	Un	sorted	Un	sorted	Unsorted
10	0	1	9	45	0	27
20	0	5	19	190	0	97
40	0	20	39	780	0	383
80	1	122	79	3160	0	1666

It can be seen that for a totally unsorted list there is little difference in time but for a sorted list the time taken is dramatically reduced. Try taking the sorted list and swapping two items and run the test again. (Here's how.) When both tests have been done type in:-

```
    SWAP ENTRY$(4),ENTRY$(33) <ENTER>
    GOTO 100 <ENTER>
```

If the only item out of order is at the top and should be at the bottom then it will still take the full number of passes to complete the sort, so if we sort from top to bottom and bottom to top on successive passes then this worst case senario is cleared.
Take the original program again and remove all the REMs.
Now run it and you should get the following results:-

No to sort	Time		No Comparasons		No Swaps	
	sorted	Un	sorted	Un	sorted	Unsorted
10	0	1	9	45	0	27
20	0	5	19	187	0	97
40	0	17	39	655	0	383
80	1	122	79	2655	0	1666

Again it can be seen that there is an improvement, although not that great. Try SWAPPING two items again and see the difference there. So this gives an idea of some improvements that can be made to a simple SORT to make it faster. Try using different DATA, numbers for example, integers or single precision numbers. By testing each SORT you can find out which is best for your application.

BASIC BASIC 4

OK folks we have have now reached the grand old age of 4.

IS THERE ANYBODY THERE?

To follow on from the last lesson where we asked the computer to compare the two strings we had entered, as the next stage we will use the comparison to sort strings into alphabetical order. I will list the program and then explain each step.

```
10 INPUT A$
20 INPUT B$
30 IF A$<=B$ THEN GOTO 70
40 LET C$=A$
50 LET A$=B$
60 LET B$=C$
70 PRINT A$;" comes before or is equal to ";B$
80 GOTO 10
```

Line 10 & 20 Is where we enter the two strings to be compared.

Line 30 The computer checks if the ASCII values of A$ are less than or equal to the value of B$ and if this is true goes to line 70 else it continues to the next line.

Line 40 we want to swap the values of A$ and B$ and so as not to lose either value we need a temporary storage area C$. Move A$ to C$.

Line 50 We move B$ to A$.
Line 60 We move C$ to B$.
Line 70 The two strings are printed in the correct order.
Line 80 we return to the start of the program.

NOTE: To stop the program use <shift><break>.

In normal BASIC the above program would be necessary to perform this simple operation BUT we have a much better computer and so we have a single command that will perform lines 40 to 60 "swap". The program now becomes

```
      10 INPUT A$
      20 INPUT B$
      30 IF A$>B$ THEN SWAP A$,B$
      40 PRINT A$;" comes before or is equal to ";B$
      50 GOTO 10
```

As you see line 30 now does all the work of lines 30 to 60 in the previous program.

The above is fine if all you want to do is compare two strings but I am sure you are more ambitious than that.

Using some new commands it is possible to store and sort as many as you wish within pre-defined limits. These are "DIM".

DIM

Suppose you have a list of numbers, for instance the number of traffic wardens that have been run over for each month in the year. To store them in the computer you could set up a single variable for each month, but you would find this very awkward. You might decide to call the variables wardens1, wardens2, & so on up to wardens12, but the program to print out these twelve numbers would be rather long and boring to type in.

The mechanism by which you can apply this idea is to use arrays. An array is a set of variables or elements, all with the same name, & distinguished only by a number (the subscript) written in brackets after the name. In our example the name would be A (like control variables of FOR-NEXT loops, the name of an array must have the first character as alpha the twelve variables would then be A(0),A(2) and so on to A(11).

The elements of an array are called subscripted variables as opposed to the simple variables that you are already familiar with. Before you can use an array you must reserve some space for it inside the computer & you do this using a DIM (for dimension) statement.

DIM A(11)

sets up an array called A with 12 elements (0 to 11 inc).The subscript can be any numerical expression, so we can now write

```
      10 DIM A(11)
      20 FOR X=0 TO 11
      30 INPUT A(X)
      40 NEXT X
```

This will allow you to input the twelve figures of deceased traffic wardens.

To list these figures out again we use another small loop

```
      50 FOR Y=0 TO 11
      60 PRINT A(Y)
      70 NEXT Y
```

Although we have now got the information into the computer it does not mean a lot as we cannot identify which is which so we need to use a second array for the months.

```
      15 DIM B$(11)
      25 INPUT B$(X)
      60 PRINT B$(Y);" ";A
```

Now we can run the program and it will identify which month is which. Great now we've go the data into the computer lets do something with it. Ahh! but that will be in the next all action installment.

NOTE:Traffic wardens read above as postmen bitten by dogs.

UKEUG Newsletter Jan 1987 Backpage

THANK YOU KEITH

I note from issue Number 12 of the newsletter you have relinquished the editorship and would like to thank you for the work you have put in on the previous issues. I certainly look forward to the newsletters arrival. I would be pleased if you would ask Chris Giles to include my name on the Front Page of membership details. My interests are primarily that of a business user, using Wordstar Professional and the Cracker, although I have been bitten by the computer bug and am taking an interest in all matters relating to it's use.

ENCORE

Don't believe all you see. Although Keith is no longer the Editor he is still very involved in the production of the newsletter and usually sees most things before they get printed. May I also take this opportunity to thank Keith for the work he has put in, not only as editor but also as host for the monthly meetings, photocopier, (300 copies of 24 sides), folder, stapler, addresser and stamper and still finding time to produce some very good articles himself. Once again thank you Keith.

DISREPUTABLE DOS

May I make a plea for contributors to the Newsletter to state the version of DOS that they are using when sending in Hints And Tips. The matter was highlighted vin the November Newsletter in a short article on using a serial printer, by Graham Betteridge. What Graham writes is valid for DOS 1.11 , but the modification which he stated to be incorrect is in fact valid for DOS 1.31. Different versions of DOS are written differently, and unless the version number is quoted , we may all end up bringing the newsletter into dis-repute!!

Thank you for the descriptions you have given in the last two issues on material from the P.D. library. When I got the catalogue disc a couple of months ago I was shattered to see the meaningless names that people give to their programs - at least they meant nothing to me!! So please carry on commenting on the library.

Can I also tell Mr. WALLIS (letters Nov) that here is at least one who would be interested in his SUBMIT program.

Having just completed my RTTY program, I'm starting on an EPROM programmer for the Einstein. Anyone got any suggestions to save me the trouble? I'm determined to fill that second ROM socket you see! T.Crossfield.

ANSWER

I agree that it is becoming important to state the version of DOS being used. I have just spent some time catalogueing the different DOSs available and it runs into 2 figures!! Hopefully next issue will see the return of the P.D. reviews.

COMMS

I have recently written a RTTY receive program for the 80 column Einstein, and if any Radio Amateur is interested I would be happy to send him a copy if he sends me a disk to put it on. It is a 4K COM program, requiring an input from a terminal unit to the User port, and has with it a text file of instructions. T.Crossfield, 8, Nevill Rd, Stocksfield, Northumberland, NE43 7JX

EXCITED

Thanks for all the info you sent me. With regard to the T.I.Video chip information, I am hoping to get more involved in it shortly. As you said there is a great deal to consume but after a brief wiz through I can see it will be quite interesting. I took the circuit diagram straight round a pals house who said it would be possible to use battery backed ram from the second rom socket by running a second connector lead to a separate board, three additional wires would have to be taken from the main board however, and this would require soldering skill as damage could be done. I have ordered a chip to switch in and out the battery, I have the RAM chip and a 3.6 volt lithium battery so I'm almost ready to try it out, I'll let you know if it works.

I'ts nice to see 3" discs going down in price, I got one Amsoft disc for £2.99 in Norwich last week thanks to Mr. Sugar I think. It's nice to find at least one computer manufacturer with some business sense. Tatung seem to go from one extreme to the other, they now sell Einsteins for £69.99, or FREE with a T.V.Video outfit. The last comment I heard from a prospective buyer was "There must be something wrong with them". All I hope is that the low price will shift enough of them to encourage the software people. D.H.

NOTE (Ed)

Tatung are not selling the Einstein for £69.99 nor is it free with a video. COMET sold off their last few machines at this price to get rid of them and Yes I Was dissappointed that I couldn't get one at that price.

MANUAL HARDWARE

Do you know where I can get a hardware manual? J.Mekin

YES You should be able to get one from Screens or Blagborough & Hebblethwaite. If not there try contacting Roy Clark at TATUNG.