



# Einstein Magazine

& ALL MICRO NEWS

Number 87

Published for users of Einstein (and other) computers  
by RPM Society.

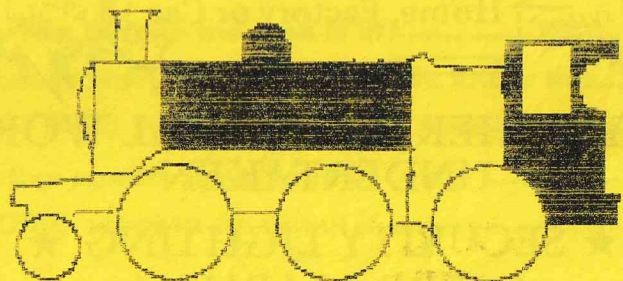
Publisher and Secretary:-

A E Adams, Ivy Cottage, Church road, New Romney,  
KENT TN28 8TY

\*\*\*\*\*

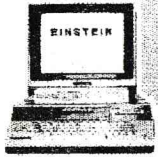
Editor: Ted Cawkwell, 9 King Street, Winterion  
N.Lincs. DN15 9RN

\*\*\*\*\*



"PUFFER"

Done with the latest ver. of SKETCH,  
will add valve gear later!



# Einstein Magazine

**& ALL MICRO NEWS**

**Number 87**

**Published for users of Einstein (and other) computers  
by RPM Society.**

**Publisher and Secretary:-**

**A E Adams, Ivy Cottage, Church road, New Romney,  
KENT TN28 8TY**

\*\*\*\*\*

**Editor: Ted Cawkwell, 9 King Street, Winterton  
N.Lincs. DN15 9RN**

\*\*\*\*\*

## **CONTENTS**

<b>The Einstein Bomb - the fuse is lit!.....</b>	<b>2</b>
<b>Undercover Story- not many people know this.....</b>	<b>4</b>
<b>Part 3 of Dave Salvage's machine code graphics.....</b>	<b>5</b>
<b>A selective screen Save/Load in colour.....</b>	<b>11</b>
<b>That BOMB- what to do about it.....</b>	<b>16</b>

**Thanks to Les Foscett for scanning the Einstein II photo  
for the main heading on this page. Three 3.5" drives  
are fitted, one in the 1/B position and two in the Epson  
drive casing under the monitor.**

**Produced using PRESSWORKS 2 on a 486/25MHz PC,  
assembled from assorted bits!**



## THE EINSTEIN BOMB

No, this is not the famous 'time bomb' that afflicts PCs and Macs but the 3 inch bomb! You will be aware that the EN gang have been looking at the problem and it is time Members were advised how best to update their systems whilst there is a reasonable chance of obtaining drives.



Some retailers are already charging a premium for 720k PC drives because they are in short supply and the demand is still there. It could cost £10 for a 720k and £8 for a 1.44M drive secondhand, and yet NEW 1.44M drives are available for £12-13.

DD disks are still available cheaply but 3" disks are hard to find and are expensive. A cynic might conclude that now is the time to be selling 3" and buying 3.5"!

The day is not far away when data on 3" disk will be almost useless in the absence of drives to read it. 3" drives are impossible to buy and many of the ones in working Einsteins are giving cause for concern.

It would be good if it were possible to recommend 1.44M drives but they are uncertain in use, to say the least, whereas 720k drives are known to work.

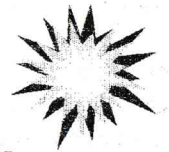
Chris Coxall recently purchased a new Sony MPF920-E 1.44M drive and found that it plugged straight in to his Einstein and worked perfectly. It is a modern version with no drive select jumpers or switches, designed to work directly on a PC. Chris's is on drive select 1/B and I assume that a twisted lead would make it drive 0, but this is yet to be checked. It needs 5 volts and cost him £12-13 at a rally.

Against this is the experience of Andrew McRobbie who bought a SONY MFD110-01 1.44M one that did not work on the TC01.

It is interesting to note that John Ibberson used new 1.44M PC drives for his Sharp MZ80B. They cost him £13 +VAT each new, and were sans jumpers which caused him to use the twisted lead mentioned in the last Newsletter. The drives are by ALPS

Electric Co. type 345 H 911B. They have NOT been tested on the Einstein.

## GREMLINS AGAIN!



I have lost the use of my Einstein Mark 2 since changing the MOS chip to try out a spare one I obtained for my grand-daughter's machine. It is very strange, the opening screen comes up as usual and some of the MOS commands (I have tried A, D and H) work as expected, but when it comes to booting a disk I only get 'disc not ready'. I have tried everything I can think of except fitting a 3" drive (I have not got a spare one), but nothing does any good. Refitting the original ROM chip makes no difference, so I can only think that something I did during the changeover caused the fault. Does a dry joint seem likely? There is nothing obvious to the eye. NB one MOS command that does not work is R - the same 'disc not ready' error is the only result.

My experiments are therefore over until I can fix this, or get another machine. I am particularly annoyed because Dick Keynes has very kindly lent me his Silicon Disk to try out and also a Flexidos ROM chip. I tried the Flexidos chip in the defunct Ein, more in hope than reason, but there was no difference in performance. I have also, at long last - I've been trying for years, got a spare 80 column card on the way and was intending to fit that to the Ein 2 as well.

Since writing the above, quite a lot of things have happened. After no communication with Tony since the end of January I today received EM 86 plus a short letter. It seems that Tony is too busy making a living to continue as Editor of EM and has me pencilled in as the next candidate. I am not confident that I could do a very good job at it, but rather than let the magazine die I am prepared to have a go, at least for a while. There are many other things which need sorting out as well and I expect that Tony will be letting members know the new arrangements in due course.

Also in EM 86 I noticed that member Mr H Yee was looking for a good home for his TC01 and I have arranged to have this machine sent to me, so I should be back in the experimental



mode before too long.

I appeal to anyone with the time and inclination to join me in editing future editions of EM with a view to taking over as Editor at some future time. The job is honorary (i.e. unpaid!) but has benefits which make it worthwhile to anyone who is into Einsteining as a refreshing change to surfing the Internet or running ready written and memory hungry Windows applications. Steve Potts, in his article in the current issue seems confident that there are people still interested in old Albert and is even suggesting a Membership drive!

#### **SOMETHING NEW UNDER THE COVER?**

Having owned and used an Einstein for over 12 years it takes something really unusual to surprise me, but I was astounded recently when I discovered a pair of Manuals with a TCO1 kindly donated by Sid Dunn.

I can hear you say it! "He's finally flipped - quite a few machines come with manuals. Nothing very strange in that!"

Ah, but.....how many have you seen with INDEXES? (or should that be indices?). Yes, an alphabetical list of contents at the end of the book. At first I thought that Sid had done it himself as a private venture, but further perusal established the fact that these were **SECOND EDITION** manuals and the Reference Manual also had extra pages.

**ELLIPSE** has 2 extra pages and **POLY** has 5, **KEY** has one and **RST** is now present. The latter, for some reason, is not mentioned in the new index!

The extra information with **ELLIPSE** and **POLY** is not exactly new, I remember seeing it in an early issue of Einstein User, but it is good to have it all to hand in the book, as the detail is rather hard to get your head round. It is all about doing open ellipses and boxes rather than the boring old complete ones and involves the dreaded PI and radians and things. So you can do circles with a vee shaped mouth like the dot-gobbler of the old Munchman games, among other shapes.

There is an error in the Ellipse syntax line though, the sixth parameter is given as 0 when it should be a; a and b are the start and end angles from the centre of the figure.

The extra page in the **KEY** command is devoted to amplification of the way carriage return is inserted into a function key definition, by pressing **GRAPH** and **ENTER** together.

In the file-handling section the correct syntax for **INPUT#** is given instead of the incorrect **PRINT#** of the 1st. Edition.

Finally, the list of Reserved words is given in index form instead of a simple list.

The Introduction Manual has a reasonable index but I was unable to find any other changes.

It is a pity that the wire spiral binding makes it so difficult to take pages out for copying. I would have been prepared to duplicate the extra pages for any members who wanted them, but as it is I am reluctant to do so - sorry! My own original manuals are much annotated and generally scribbled in with information I would be reluctant to be without, but my first idea to add the new pages from the 2nd. edition came to naught when I tried mastering the springs. I shall have to soldier on with two sets of manuals.



#### **INTRODUCTION TO AN INTRODUCTION TO MACHINE CODE GRAPHICS (3) by Dave Salvage.**

Welcome back to the third and final introductory article.

Now to expand on some commands already used in previous introductory articles.

You may remember using the **DEFB n** command to **DE**fine a Byte of data for use with the **RST 8** command to access the **MCAL** routines in Albert's MOS, n representing a byte of data. It is not surprising, therefore, that it is possible to define larger sections of data.

**DEFW nn** **DE**fines a Word (ie 2 bytes) of data, and **DEFM "s"** **DE**fines a Message held as the ASCII representation of the



string "s". To make things easier when programming in Assembler, these words and messages can be given labels to make them easy to refer to.

An example, to print a message to the screen:

```
LD B,10
LD HL,MESSG:
LOOP:LD A,(HL)
RST 8
DEFB &9E
INC HL
DJNZ LOOP:
RET
MESSG:DEFM "A MESSAGE!"
```

So what is happening in this routine? The B register is being used as the special counter register, and is loaded with the number of characters in the message to be printed. The HL register pair is loaded with the address of the message given the label MESSG: A loop LOOP: is then started. The accumulator (register A) is loaded with the contents of the memory address held in the HL register pair (extended indirect addressing).

The contents of the accumulator is then printed to the screen at the current cursor position using RST 8 and MCAL &9E (subroutine in Albert's MOS). The contents of the HL register is then increased by 1 to become equal to the address of the next character in the message.

The contents of the B register (the counter) is decreased by 1, and if NOT ZERO, the machine code routine returns to the beginning of the loop, LOOP:, and the next character of the message is printed to the screen. This continues until the contents of the B register are zero, when the loop finishes and the program moves on to the next instruction. In this example, this is RET, and the routine returns to the position in the program immediately after that from which the routine was called.

I have just realised that there is a new Assembler instruction in that little routine. INC simply increases by 1 the contents of whichever register or register pair is specified. This can be applied to registers B, C, D, E, H, L and A, but not F or PC. Not surprisingly, there is also a DEC instruction to decrease the contents of the specified register by 1. If, after this instruction, the contents of the register are zero, the zero flag is set to 1. This can be tested to find out when this condition has been achieved, usually as part of a conditional instruction, eg JP Z,address or label.

Individual bits of most registers can be tested as well, using the BIT b,r instruction where b is bit number from 0 to 7, and r is the register. If the bit tested is zero, then the Zero Flag is set to 1. The BIT instruction can be used with registers A, B, C, D, E, H and L, and also indirect addressing with (HL), (IX+d) and (IY+d).

As well as testing individual bits of most registers, individual bits can be set to 1 or reset to zero, using SET b,r and RES b,r instructions, respectively. Only the same registers and indirect addressing as for BIT b,r can be used. Two special instructions exist to set and complement the Carry Flag, SCF and CCF. This is the only flag bit which can have such instructions applied to it. In order to test other bits of the Flag register, other than as part of conditional instructions (should it ever be necessary), the AF register pair would have to be PUSHed to the stack and then POPped from the stack into another register pair, such as BC, and then the relevant bit of the C register tested, the C register now holding the contents of the Flag register.

Sorry! Got sidetracked there.

Back to DEFINing areas for data in an Assembler program. One further DEFINE instruction exists, DEFS n, which DEFINes n bytes of memory for Storing data, for example that generated during intermediate stages of certain calculations. Again, it is advisable to give such instructions labels so they can be referred to easily, eg STORE1:DEFS 4.

Now for something completely different! Well almost.

There may be occasions (usually in complex programs I



presume, since I have not yet had to use these instructions) when blocks of memory may be required to be moved from one location to another. I believe this can be done as a form of program protection, with a series of block moves reorganising the machine code program when it starts running so that when jumps are executed they occur to the correct locations in the program, but prior to running the program the jumps will occur to incorrect locations in the program, making it very difficult to decode. Any other suggestions as to how these instructions can be used?

Anyway, for straightforward transfer of a block of data from one memory location to another, LDIR and LDDR are used. BC is loaded with the number of bytes to be transferred, HL is loaded with the address from which the data is to come, and DE is loaded with the address to which the data is to go. LDIR then LoadS data from address HL into address DE, Increments the address pointers and Repeats having decremented BC. When BC becomes zero, the Load instruction is not repeated, and the required block of data will have been moved. LDDR is very similar, only the address pointers are Decrementd rather than incremented.

If other instructions are required during the transfer of blocks of data, such as printing the data to the screen as it is transferred, similar instructions are used without the Repeat: LDI and LDD. These have to be incorporated into a loop. BC is loaded with the number of bytes to be transferred PLUS 1, and the Parity bit is used to test when the loop is complete. The Parity bit is reset to zero when BC is decremented to 1, and this is tested with the PO

condition. All other values of BC set the Parity bit to 1. This is why BC needs to be loaded with one more than the number of bytes to be transferred.

Comparison of the accumulator to blocks of memory can also be done, searching for matches. The instructions are CPI, CPIR, CPD, CPDR.

HL is loaded with the address at which the search is to start. For CPI and CPD, BC is loaded with the number of bytes to be

examined PLUS 1, as for LDI and LDD. Again the Parity bit is used to determine when the required memory has been compared, being reset to zero when BC = 1. It is presumed that something will be done on each occasion of a match of the accumulator with a memory location, such as printing the matching character to the screen or incrementing the contents of a memory location or another register.

With CPI and CPD, both the Parity flag and the Zero flag need to be tested for every comparison. With CPIR and CPDR, comparisons continue automatically until a match is made, making execution quicker, but otherwise the program is unchanged.

Such a program might be:

LD A,&28 Put "(" in A

LD HL,0 Start at beginning of memory

LD BC,0 Search 64K (BC=bytes+1, 0-1=FFFF)

LOOP:CPI

JP PO,FINISH: Finish if Parity flag is set to 1

CALL Z,FOUND: Call FOUND if zero is set (ie a match)

JP LOOP: Loop again

FINISH:CALL Z,FOUND: Check zero flag before finishing

RET Back to main program

FOUND:CALL PRINT: Print "(" on screen

RET Return from subroutine FOUND

PRINT:RST 8

DEFB &9E MCAL to print contents of A to screen

RET Return from subroutine PRINT

This illustrates several techniques from the previous articles: LoadIng registers and register pairs, looping, a conditional absolute Jump, an unconditional absolute Jump, CALLing subroutines, labels, and using an MCAL.



For faster execution, substitute CPIR for CPI, and to save a byte of machine code, make the unconditional absolute jump an unconditional relative jump, ie. JR LOOP: instead of JP LOOP:. Relative jumps require 2 bytes of machine code whereas absolute jumps require 3. The restrictions on relative jumps are given in the previous article of this series.

In the last article, I explained how to load and run a machine code subroutine from a BASIC program. What if you want to load and run more than one, particularly if you are developing a machine code program in sections and want to load each section one after the other?

You may remember that to load one machine program, we set the upper limit of memory available to BASIC using the CLEAR instruction. This also sets the memory location at which the start of the machine code subroutine is loaded. For more than one machine code subroutine, that which will reside highest in memory is loaded first, with subsequent routines loaded at lower and lower memory locations, each being sufficiently lower than the previous routine to avoid overwriting it. This serves to gradually lower the upper limit of memory available to BASIC, and prevents the possibility of BASIC overwriting any of the machine code routines.

If the routine lowest in memory was loaded first, though there would not be a problem from the point of view of the other machine code routines, the upper limit of memory available to BASIC would become higher, and the machine code routines lower in memory would not be protected from overwriting by BASIC.

Just a brief example:

```
CLEAR &E000:LOAD"ROUTINE1.OBJ"
CLEAR &D800:LOAD"ROUTINE2.OBJ"
CLEAR &B000:LOAD"ROUTINE3.OBJ"
CALL &B536
```

where &B536 is the start address for the machine code program which uses all three machine code routines.

One final comment in these introductory articles.

It is very useful to have a list of the Einstein "Scratch-Pad" locations. These are the memory locations where Albert stores gems of information such the current position of the cursor, the current cursor character, text colour, graphics parameters, disc and drive related information, etc. There is a complete list in Alternative Micro News Vol 1,2 pp 11-14, and "Albert Revealed".

Well, I hope I have encouraged some of you that machine code programming can be fun. To follow this introductory series, will be a series of articles on machine code graphics programming, and the use of interrupts. You too can make the screen scroll sideways, play and print music and compress many lines of BASIC programming into several bytes of machine code which will run many times faster than the BASIC program. I bet you can't wait for your next issue of Einstein Magazine!



## A SELECTIVE COLOUR SCREEN SAVE

by Ted Cawkwell

Since using Peter Hambidge's colour Save/Load routine in arecent article I became curious about how the Colour Table had been put into it. I used Zen to make a listing of the source code and, whilst studying this, realised that with a few changes it would be possible to save single or multiple lines from the screen, rather than the whole screen, and then load them back in another position.

The machine code takes the start of the screen in VRAM which is 0000hex and then loads one line at a time into RAM until the whole screen is done, by loading the value 18hex or 24 decimal (24 lines on screen) into the B register. I reasoned that if I changed the start address and number of lines I could select one or more screen lines and save them and then change the position I wanted them Loaded back to.

It turned out to be very easy. Line 0 at the top of the screen runs



from 0000hex to 00FFhex, so the next line starts at 0100hex and the next at 0200hex, etc. These are two byte numbers but only the first byte needs changing, so the start address of any line is the same as its decimal number but written in hex notation. I.e., the bottom line 23 is 17hex which is the same number. The numbers of lines to save or load is 1 to 22 in this program to save any trouble with scrolling if the bottom line is used to the end. In practice, this means that any screen image should only use lines 0 to 21 because line 22 is used to input messages during execution of the program.

The colour screen is done in just the same way except that the screen starts at 2000hex, so no problem there. The first byte 20hex is 32 so we just add 32 to the line number.

LLDEMO (large letters demo) was written to show both what could be done with double size letters and how they (or any other display) can be repositioned or repeated on the screen. To keep the program fairly simple only one filename (SCRNCOPY.OBJ) is used, so files will need to be renamed if they need to be kept. It would not be too difficult to write a routine to choose a filename but it might be difficult to remember just what was on a dozen or so files!

I think the selective system might be useful for building up Menu and opening screens without having to go through the tedious business of writing everything out in Basic every time. Stan Gibbs has been telling me he finds a need for something like this. You see, he's got me going again!

```
10 REM ***LLDEMO.XBS JAN 1998 UKEUG
20 REM ***PD ROUTINES PUT TOGETHER
30 REM ***BY TED CAWKWELL DOS 2.05
40 REM ***XBAS 5 or 1.31/XBAS 4
50 CLS: CLEAR &A000
60 WD$="LARGE LETTERS"
70 X%=14:Y%=2
80 GOSUB 450
```

```
90 WD$="DEMONSTRATION"
100 X%=14:Y%=6:TCOL8
110 GOSUB 450
120 WD$="WITH COLOUR LOAD/SAVE"
130 X%=10:Y%=10:TCOL11
140 GOSUB 450
150 WD$="FOR U.K.E.U.G"
160 X%=14:Y%=14:TCOL3
170 GOSUB 450:TCOL15
180 REM SAVE SCREEN
190 SP$=MUL$(" ",38)
200 LOAD "SCRNSL.OBJ"
210 PRINT@0,22;:INPUT"Start of SAVE - Line No.(0-21)";RN
220 PRINT@30,22;" ";
230 PRINT@0,22;:INPUT"Number of Lines to SAVE (1-22)";NL
240 POKE &A00B,RN:POKE &A011,NL
250 POKE &A02F,RN+32:POKE &A035,NL
260 PRINT@0,22;SP$
270 CALL &A000
280 SAVE "SCRNCOPY.OBJ",&B000,&E000
290 PRINT@0,18;"SCREEN SAVED, ANY KEY TO LOAD IT BACK."
300 Y$=INCH$
310 PRINT@0,18;SP$
320 REM LOAD IT BACK
330 CLEAR &B000
340 LOAD "SCRNCOPY.OBJ"
350 PRINT@0,22;:INPUT"Start of LOAD - Line No.(0-21)";RN
360 PRINT@30,22;" ";
```



```

370 PRINT@0,22;:INPUT"Number of Lines to LOAD (1-22)";NL
380 PRINT@0,22;SP$
390 POKE &A04E,RN:POKE &A050,NL
400 POKE &A06F,RN+32:POKE &A071,NL
410 CALL &A049
420 GOTO 420
430 REM LARGE LETTERS SUBROUTINE
450 PRINT@X%,Y%,"";
460 FORJ%=1TOLEN(WD$)
470 I%=ASC(MID$(WD$,J%,1))
480 A%=I%*8+6144
490 B$=HEX$(VPEEK(A%),2):C$=HEX$(VPEEK(A%+1),2):
D$=HEX$(VPEEK(A%+2),2):E$=HEX$(VPEEK(A%+3),2)
500 SHAPE128,B$+B$+C$+C$+D$+D$+E$+E$
510 PRINT CHR$(128);CHR$(8);CHR$(10);
520 B$=HEX$(VPEEK(A%+4),2):C$=HEX$(VPEEK(A%+5),2):
D$=HEX$(VPEEK(A%+6),2):E$=HEX$(VPEEK(A%+7),2)
530 SHAPE128,B$+B$+C$+C$+D$+D$+E$+E$
540 PRINT CHR$(128);CHR$(11);
550 NEXT:RETURN

```

Enter the listing, being very careful with lines 490 to 540, and SAVE"LLDEMO" before trying it out. You will need to know what is on which line to work the program and this can be seen above i.e. LARGE LETTERS starts at Line 2 and takes up 2 lines on the screen. In line 70 Y% is given as 2. Y% is the number of lines down the screen, X% is the number of columns. Note how the 'message', X% and Y% and text colour is set before the GOSUB to the large letters routine at 450.

Note also that at the end of the program the routine is stuck at line 420 just going round and round. This is to leave a clear screen with no cursor. Exit using Control/Break.

When you run the program you will see the large letter messages and the request at the bottom of the screen "Start of Save - Line No.(0-22)". For a start try entering 2, and another 2 for no. of lines. The Save then occurs and you press any key to load the screen back.

What you have saved, if all is well, is the 2 lines holding the words 'LARGE LETTERS'. Now to load them back in place of 'FOR U.K.E.U.G' enter 14 followed by 2. Don't blink or you will miss it! LARGE LETTERS will now be at the bottom of the screen as well as the top. Now experiment. It will help if you realise that the phrases are 2 lines high and separated by 2 lines. Try taking a 6 line chunk and putting it somewhere else, to see the effect.

By the way, it should be obvious that SCRNSL.OBJ must be on the same disk as the program! My SKETCH article tells you how to get it. As a matter of interest, if you happen to run the program from Drive 0 when the disk is in Drive 1 by entering RUN"1:LLDEMO" it will fail to find SCRNSL.OBJ because it will be looking for it on Drive 0! Unless it happens to be there as well! The remedy is to either put the disk in Drive 0, or change to Drive 1 before attempting to RUN the program. All very obvious perhaps, but it has caught me out from time to time! When a drive is not specified the current drive is the one searched.

To use the Load/Save with your own programs you only need Lines 50 and 180 to 420. You can slot your own program between 50 and 180 (renumbering if necessary) and then contrive a better end than line 420, but be careful not to use the bottom line right to its end else you will scroll off the top of your screen.

If you make it part of a drawing program you might want to return to that, and if the Save/Load was in a GOSUB say, at the end of your drawing routine, it would still be available later to save the whole screen if required.





## 💣💣💣 THE BOMB AND WHAT TO DO ABOUT IT 💣💣💣

*Why is there a 3 inch bomb to worry about?*

The original 3" drives with which the Einstein is equipped are no longer made. They are difficult to obtain secondhand and many of the ones in use NOW are giving cause for concern. In addition, the 3" disks themselves are no longer made and good ones are almost impossible to obtain; they are also reputed to be less durable than the 3.5" types. The magnetic coating is less robust than newer disks and the white plastic centre of the disk can break up. They always were, and still are, expensive to buy. For these reasons, 3.5" DD disks make more sense.

Most, if not all, owners will have all of their programs on 3" disk so the breakdown of the 3" drive will result in a computer that is no use to man nor beast! Some lucky owners will have a pair of drives so can put off the dread day, but many will not.

Members with single-drive machines should realise that, though it is not difficult to install a 3.5" drive, all of their programs on 3" disk will need to be transferred to the larger size. To do this obviously needs both drives to be working on the same machine. Those who write many of their own programs will be particularly interested in making sure it will be available on media to suit a future drive configuration.

It is in this area that most of the snags arise. Anyone in possession of Crystal System 5 or Tatung DOS 80 will have little trouble as both of these systems support mixed drive configurations, but any owner with only the standard XDOS 1.31 (or the earlier XDOS 1.11) will find that although a 3.5" drive can be used, it can only be formatted for 188k per side just as if it were a 3" disk. As it cannot be turned over in the drive this is only half of the 3" storage space!

I know quite a few members who have been using this system for several years and are quite happy to have 188k on a disk capable of holding 786k. They reason that the disk costs about 20p rather than over a pound, and they are easier to obtain.

There is also the fact to bear in mind that the more you use a 3.5" drive the less use your 3" drive gets and the longer it will last. Even this very basic level of use means that at least one 3.5" drive makes sense.

Having transferred your files to the DD disks means that when the 3" drive finally expires you can change the 3.5" to drive 0 and carry on with it as your boot drive. A second 3.5" drive could be added to the system later, if needed.

There is at least one DOS in the Software Library that is designed for mixed drives, ZDOS, but it is only marginally compatible with Crystal Basic. It has a fixed setup of 40 track single-sided for drives 0 and 1, 80 track double-sided for drive 2 and a Silicon disk for drive 3.

In house we have CP/M Plus by Duncan Elvis but it is not yet on general release; it has the capability of setting the drive configuration but is not compatible with Crystal Basic. It is planned to issue a suitable Basic similar to Xbas.

It is possible to patch Xdos 1.31 to use 80 track DD drives instead of the standard drives but not to support mixed drives. In this case Xbas 4 is fully supported.

*So what is the sudden hurry?*

It is understandable that Members who have been chugging along happily for years with one or two 3" drives and an apparently indestructible machine should ask such a question. It is true, during the Einstein's lifetime many good machines have bitten the dust and the rise of the PC has been phenomenal. Albert is the last survivor of the 8 bit computers to have a reasonable User Group support, but even that support is beginning to flag. I believe that, with new drives and cheaper disks the micro has a good bit of life in it yet.



The sudden hurry is because 720k PC drives of the type that are suitable for fitting to the Einstein have, in the last 12 months, become hard to find, the price has gone up and rumour has it that DD disks are no longer being produced.

Since 1.44M PC drives are easily available and appear so similar to the 720k version that they could be expected to work, the Einstein Newsletter was started a few months ago to gather together a group of Einstein enthusiasts to investigate the whole question of drives and the results of these investigations is now to published in the pages of this magazine.

Using a 1.44M PC drive does not imply that it will ever be possible to put 1,440k of data on to a disk in such a drive as, unfortunately, the floppy disk drive chip in the Einstein is not designed to do it. There is likewise no advantage in using HD disks, in fact, experience shows that many HD disks will not even format properly for 786k.

The attempts to use 1.44M drives has met with mixed results, I know at this moment of only two members who have succeeded. Chris Coxall has two drives working as drive 1/B, one is a new SONY MPF920-E, the other is a used MITSUBISHI MF355C-18UC. The Sony has no drive setting jumpers and simply plugged in and worked straight away. The other needed a jumper set to 1, and 3 others in the correct positions.

Dick Keynes has a MITSUMI D35976 working as Drive 2/C, this too has no jumpers. Dick has used a twisted lead.

I have tried my 1.44M TEAC FD235HF and got it working twice but each time it stopped working the next time it was tried. The reason remains a mystery. On the other hand the 3.5" drive I have been using for many years as drive 2/C is a TEAC FD235F, 786k, which installed with no trouble and has always worked perfectly.

Installing the 720k version therefore seems to be the better bet for a quiet life!

### *What is the best way to proceed now?*

It seems to me that the best, safest way is to obtain a 720k PC drive and a good supply of 3.5" DD disks as soon as possible. The choice of 720k drive is because most seem to be easily fitted to the Einstein and none have displayed the problems we have experienced with the 1.44M variety.

### *How do I know I have the right drive?*

The following are known to work:

TEAC FD235F 100U

NEC FD1036A

CITIZEN OSDC-29C

SONY MP-F53W-30D

SHARP ER01FD

The Sony drive is unusual in being 1/8th of an inch thicker than the usual 1 inch, also the drive light does not work on two samples checked.

The Sharp drive works off 1.8 volts which means a special power supply is required. If you find one complete with PSU it should plug in and work OK.

The majority of the drives for sale seem to be the TEACs or NECs. Computer rallies are a good source.

The right sort of drive to buy is four inches wide by one inch thick and about six inches from front to back and is known as a 'half-height drive'. Anything larger than this is likely to be an old model and something of an unknown quantity, and should be avoided. There should be the familiar front panel, usually grey, with the slot and flap for the disk and an LED and push button.

The details of the drive are found on the back panel near the input connectors, which should consist of a 34 pin and a small 4 pin socket. Avoid drives with 26 pin sockets or without a four pin power socket.



It may be difficult to decide whether a drive is 720k or 1.44M. Some drives have a small switch which can be seen inside the disk aperture on the righthand side. the presence of this switch means that the drive is a 1.44M type, but unfortunately it may be absent in some drives intended for IBM or Amstrad computers. If you are not sure it is probably best to buy from a dealer you can trust. One I can recommend is Willis Enterprises, Foley Bank, Worcester Road, Great Malvern, Worcs. WR14 4QW Tel: 01684 575772. They give a 3 month guarantee and will deliver by GPO parcel post rather than one of those private concerns that double the price of everything to get it to you by the next day. Willis's current price for a 720k drive is £10.

In addition to the 'bare' drive there are occasionally to be found at computer rallies the original add-on 3.5" drives which come in a casing complete with PSU and leads and can be plugged directly into the socket on the back of the computer. These will usually have markings such as Tatung- for Einstein. Note that there is also a 3" version which looks very similar! As far as I know these are no longer on sale new.

In the next issue will be details of connecting your drive and using the Einstein power supply to run it. This will obviously involve lifting the lid and delving inside, but no soldering will be required and you will not need a degree in electronics.



## THAT OTHER 'BOMB'

Albert is of course immune to the Millenium Bomb but it may be that some programs, Bank programs come to mind, may not be written to use a four digit date in which case they will be troublesome. Any member with such a program written in Xbas is welcome to send me a copy and I will see if I can sort it out. I am afraid .COM programs are not within my capabilities, sorry. Also, it is probably unrealistic to expect large data banks to be converted.



## SHARP USER CLUB MAGAZINES -- BACK NUMBERS

\*\*\*\*\*

If you are a member of the Sharp User Club too, you will know that most Sharp computers are 8-bit Z-80 machines like the Einstein, that Xtal made CP/M available for them (as a result of which it developed the more user-friendly clone that we know as XtalDos), & that it adapted Xtal BASIC (originally written for the Nascom computer) to Sharp machines before doing the same for Einey.

You'll also know that KUMA Software and a lot of other software houses supplied software for both Sharp and also for Einstein. The result is that an amazing amount of the content of the Sharp User Club magazine is of direct or of definite general interest to keen Einstein users too.

What's the point? Simply that Einstein User Group is now providing a reprint service for those issues of the SUC magazine that have gone out of print, and you too can take advantage of this to get your hands on copies, easily and cheaply.

SUC magazine are great meaty volumes, packed full of items to interest keen Einstein people too, and the following are now available from stock:-  
Volume 1 Nos 1, 2 and 3; Vol 3 Nos 1 and 4; Vol 4 Nos 1, 2, 3 and 4; Vol 6 No 2; and Vol 9 No 3.

Prices are 50p per order, plus 50p each for 1/1, 1/2; £1.00 for 1/3; & £1.50 per copy for the rest.

Back numbers not listed, plus a comprehensive index to all back numbers from Volume 1 to Volume 16, can be obtained for you (or can be reprinted if they have gone out of print) at £2.00 each.

\*\*\*\*\*

## EINSTEIN USER GROUP MAGAZINE BACK NUMBERS

Are available from Einstein Monthly 1/4 onwards at 50p per copy plus 50p per order. For the full benefit of this we badly need a volunteer (or more than one?) to co-ordinate all the bits of Einstein index that members have compiled, (and are still?) to produce a master index. Volunteers please?