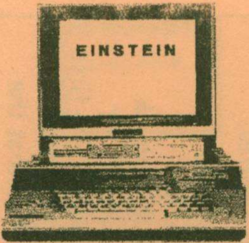


Einstein Magazine

& ALL MICRO NEWS Number 103

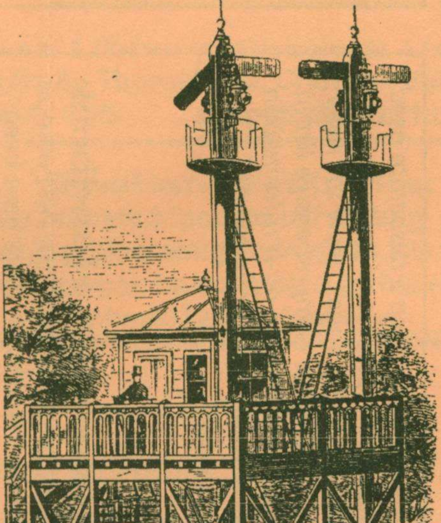
Published for users of Einstein (and other) computers

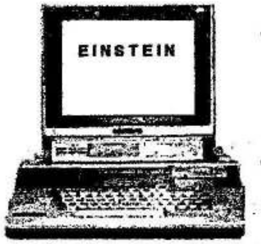
by RPM Society.

Publisher and Secretary:-

A E Adams, Ivy Cottage, Church road, New Romney,

KENT TN28 8TY





Einstein Magazine

& ALL MICRO NEWS Number 103

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: Bob Deeley
33 Britain St., Dunstable, Beds. LU5 4JA
****

Software Library, Shows & Useful Bits:
Steve Potts 85 Thorold Ave, Cranwell Village, Lincs. NG34 8DS s.p.potts@talk21.co.uk
****

Einstien Webb Site:
http://members.tripod.co.uk/~tatung_einstien
ftp://tatung_einstein:gwcdrw@ftp.tripod.co.uk/
Ken Ross can be contacted by his email...
cbm8032@bigfoot.com

Contents

No Time like the Future

There are occasions when we all would like to be able to predict the future with some degree of accuracy, none more so than the science fiction writer or film set designer. Her or his skill is to extrapolate from the past, towards the present technological advances, to a state at which they might have progressed to in the particular period they wish to depict. However it's often revealing when either the time period is reached or some new technology is encountered, and these are clearly out of step with the popular imagination, as we relate events to familiar books, films and television series even from our recent past. Take for instance the hover vehicles in 2001 a Space Odyssey - who would have thought that cars by then would actually be the older models! Retro styled Jags and Rovers. Conversely there's the rounded corners of the monitors and wall screens found in Buck Rogers, Thunderbirds, Lost in Space to name a few - a style dictated only by the limitation of the cathode ray tube of the day. At any given time, it would appear that we fail to calculate for what is really desirable - dilemma of all marketing people who will bore the socks of you with questionnaires given half a chance.

Emulating TC01 Possible

A decade ago, who would have thought eight-bit four meg-hertz computing would still be on the go in 2001, along side thirty two bit, one point seven giga-hertz machines, keeping good company with Bush large dial radios, Arga solid fuel cookers, Volks Wagon Beetles and Morris Minors. Neither has the TC01 been emulated on another platform, to date. I think we have been through the pros and cons of this before so I won't dwell... But to keep you informed our web page has been contacted by Kevin Thacker <kevt@arnoldemu.freeserve.co.uk> he is interested in writing a TC01 emulator, he's in M.E.S.S. - no, he's not IN a mess! That's Multi-Emulator Super System. But he might well get into one if he doesn't get some help with the task - anyone...? Kev writes... The emulation is quite advanced performing a simulation of each system hardware component - video, keyboard, CPU etc. MESS is a non-profit project and I make no money from it. The source and binary to MESS are freely available at http://mess.emuucrsc.com/ . To use each system you will often inquire a system ROM or boot ROM and software in a emulator based file format; a cartridge dump, a ROM dump, a disk image, a tape image or sampled tape sound in a format like WAV. The aim of MESS is to document old computer systems by allowing them to be emulated. Since the source is also free, people can discover about the hardware by reading the source; which often includes comments. I have a digital photograph of the '256' motherboard. On this, I can see a Z80 CTC timer chip, AY-3-8910, 8251 serial IC, wd1778 disc controller, ram and ROM. The graphics controller is likely to be a

TMS9928 type as used by the MSX genre. (http://www.8bit-museum.de/pcb/)TheAY-3-8910, Z80 CPU, Z80 CTC, TMS9928, wd1770 and 8251 are already emulated in MESS to a high degree, all I would need to do is put these together in the correct way to make a emulation of the Einstein computer. I don't own an Einstein computer, so to make this emulation I would require the help of your user group. The information I would require to make this emulation would be: 1. Dump of the system ROM of the Einstein. 2. List of the I/O ports (or memory addresses) to access each element of the hardware. 5. Dump of a disc I can run to test it is working (CP/M + some other utilities). The dump of the ROM could be made using a Z80 program. The dump of the disc, in the form of a disk image (a single file containing all the data from the disc) could be created with a utility like CPDRead (http://www.classicgaming.com/caprice/ - website of Caprice32: a Amstrad CPC emulator for PC). I feel the emulator could help to promote the Einstein, and as the real computers start to fail to work or become less available, then the emulator could be used in place to still enjoy the experience of using an Einstein. To make the emulation using the existing MESS framework would not take long if I have the information above readily available. I will give credit for any help received, and this will be listed in the source code and in the system information docs that come with MESS. Ultimately it is your call, please let me know via e-mail or via letter, what you decide and thoughts about this are. kevt@arnoldemu.freeserve.co.uk

Kevin Thacker, 57 Oakland Road, Hillsborough, Sheffield S6 4LT.

Ed. A list of useful web sites there, Kev seems to be involved in some good things and would himself enjoy TC01 computing.

----@@@----

All downhill with a hacking cough

John Marriott July 2001

Not quite what it seems, but digging into other people's programs, protected disks, what have you - have one good side effect, the need to create your own little routines, techniques (a polite term for tricks?) in order 'find' what you're looking for.

Idly flipping through some old computer magazines I came across the "Alternative Micro News Vol1/5" which had a competition in it, part of trying to 'solve' it (so I thought) seemed to point for the need to convert Binary into Hex, then 'assemble' that and try to work out where to load it into memory, do the 'Call' and 'Bingo!' the answer...wrong, yet again - but at least I'd cobbled up the following "Binary string input to Hex, poke into RAM". Program...

```
10 CLEAR &7FFF:P=&8000:REM binary input/hex
   conversion, poke to RAM - John Marriott 2001
20 CLS:N=0:X=1
30 INPUT A$
40 C$=LEFT$("00000000",8-LEN(A$))+A$
50 FOR L=8 TO 1 STEP-1
60 B=VAL(MID$(C$,L,1))
70 N-N+(B*X)
80 X=X+X
90 NEXT
100 PRINT C$;"   ";HEX$(N,2)
110 PRINT "CORRECT?"
120 K$=INCH$:IF K$="" THEN 120
130 IF K$="Y" THEN POKE P,N:P=P+1:GOTO 20
140 IF K$<>"N" THEN 120
150 CLS:PRINT "CORRECTION TO BE MADE":FOR
     T=1 TO 500:NEXT:GOTO 20
```

Perhaps I've gone senile, but surely about the time the TC01 was being 'designed' there was still a lot of programming around in 'Binary Form' - even the Spectrum 16/48K machines had a 'Binary Input' command, so where is it on the TC01? Not much of a program, but it does the job.

Then, and I won't mention Stan Gibbs' name, a 'working disk' of Roy Primes "Tubes" landed up in my TC01's drive - and I must admit that the opening menu colour choice wasn't mine either! To those who've not come across this program it's a 1990 era 'disk utility' of 'select icon by moving pointer' written in BASIC then put through the SYSTEM5 'compiler' to finish up with an .XBI file which needs XR.COM to run it. There is some rudimentary 'copy protection' system (you can make 'backup' copies but not file-to-file copy) which 0:MOS<e> just walks by - and this isn't an article of how to rip off Roy Primes' program, but the need to create my own routines.

Whilst I've a legitimate copy of SYSTEM5 with XC.COM (the pseudo compiler program!) I've never 'worked' on .XBI files, so another learning curve. Via MOS I looked at the START.XBI file which had the 'start up menu' section, and then realised that I didn't have a 'Table of Reserved Words', more concisely their TOKEN VALUES. Okay, if you type in the command PRINT you can see that word on the screen, look in 'working area' of RAM and you'll see that word, but in the 'stored program area' it'll have the token value of &A2 (you'd think that PRINT# would have something similar but it hasn't, it's token is made up with &A2 &23).

The resulting programs were my way of getting those TOKENS

REM - TOKENS.XBS

```
10 REM a program listing TOKEN VALUES of X80
   BASIC command/reserved words SYSTEM5
   DOS - John Marriott 2001
20 CLS:T=&6F:L=0:J=2
30 FOR M=&4037 TO &4278
40 IF M=&41EE THEN NEXT
50 IF M=&41EF THEN NEXT
60 IF M=&41F0 THEN T=T+&FE96:J=4
70 LET C=PEEK(M):IF C>&80 THEN C=C-&80:
   GOTO 100
80 PRINT CHR$(C);
90 NEXT
100 PRINT:PRINT HEX$((T),J) TAB(8);: T=T+1
110 L=L+1:IF L>22 THEN 120 ELSE 80
120 K$=INCH$;IF K$="" THEN 120
130 L=0:GOTO 80
```

REM - TOKEN2.XBS

```
10 REM program to display individual TOKEN VALUES
```

```
        of X80 BASIC command/reserved words SYSTEM5
        DOS - John Marriott 2001
20 CLS:PRINT@1,10,"PRESS ANY KEY TO SHOW
        NEXT TOKEN VALUE"
30 T=&6F:J=2
40 FOR M=&4037 TO &4278
50 IF M=&41EE THEN NEXT
60 IF M=&41EF THEN NEXT
70 IF M=&41F0 THEN T=T+&FE96:J=4
80 LET C=PEEK(M):IF C>&80 THEN C=C-&80:
        GOTO 110
90 PRINT CHR$(C);
100 NEXT
110 K$=INCH$;IF K$="" THEN 110
120 CLS
130 PRINT:PRINT@16,10,HEX$((T),J);"    ";:
         T=T+1:GOTO 90
```

They both do the same thing, but slightly differently (how English is broken?) and both end up with an 'Error message' - but the job gets done. The line "IF M=&41F0 etc..." is to allow for the 'patching' CRYSTAL did to tag on extra commands as there are two 'stray' bytes at &41EE/EF - but once you've got a TOKEN VALUES list out, check those values against dummy BASIC programs...

...you never know - I could be wrong, yet again?

Sorry I've not done anything similar for earlier DOS/XBAS, so far I've had no need - which of course sums me up - learn on a 'need to know' not 'might need to know' basis.

Boot up with your SYSTEM5 disk, load X80.COM, key in a few lines of program, switch to MOS and >T44A044C0<e> and compare your 'token values' against those few program lines - can you see how easy it is for 'hackers' to change the bytes on a disk of (say) a protected program. Taking Line 1330 of FRUIT.XBS from 'TCOL 5,6: etc...' which sits in 'stored program area' as 'FF 81 20 35 2C 36 3A etc...' to 'AC 3A A4' which changes the start of that Line to 'STOP:REM etc..' for listing after a partial 'run' to see what bit of 'decoding' of machine code has gone on!

Incidentally, I did run FRUIT.XBS through the pseudo compiler of XC.COM to get a FRUIT.XBI program - and it's a nicer program for that! I don't know of XC.COM's limitations, but there's a music program which is slower than slow which I'd like to give it a try on - on the depressive side, probably one which XC.COM can't handle?

Not that YOU should need reminding - never 'play' with original disks or disks you don't want to MESS up! If you're doing a MOS disk read ALWAYS write down the values you're using e.g. >R10004000060B1<e> which means that you're reading into RAM &1000 to &4000 from Sector &06 Track &0B of Drive 1: so if you 'forget' it'll be just as easy to do a >W10004000<e> and any unprotected disk in your Drive 0: has been well and truly ruined (stronger words have been spoken - believe me!). Whilst I tend to work in 80 column mode (the screen 'works' quicker), it's safer to be in 40 column mode where you can 'eyeball' the 'write back' characters easier - trust me!

Hopefully none of you will think "...what an ideal way to copy some copy protected disks..." when my 'CopyAny' won't - tut, tut. Yes, that working disk has gone back from where it came, all my bits and pieces relating to its coding erased/scrapped - aren't I a good boy...or is it that I find MOS cheaper, quicker, easier to use?

----@@@----

TC01/PC File Exchange – *Part 2*

SRLRUN & SRLLOAD and the BBCBASIC ASSEMBLY LISTING.

By Chris Coxall

The only assembler I know is in BBCBASIC. What I have learned has been through self-discovery, manual bashing and any book or magazine I could find with a useful hint or tip. The most helpful book for routines being "Mastering Machine Code On Your Spectrum ZX" by Toni Baker. The hardest part was finding information on USART 8251A so to be able establish RTS & CTS handshaking with PC terminal Programs. I have done my best to explain the assembly in the listing but experienced programmers will have to forgive obvious breaking of convention. All I can say is that SRLRUN has worked successfully for me with Windows terminals from a 286-laptop win3.1, a 486 win95 and an AMD K62 475 computer win98. I have used other programs and commands successfully. One, a BBCBASIX program (bbcbasic for the PC), which I have used to transfer CP/M 2.2 files down loaded from the net stored as binary COM files. More on this later...

Loading files from the PC via serial input is not as fast as loading from an Einstein drive so it is not so good for files that are regularly used. For files that might be used or rarely used, having them all together on a PC hard drive is ideal. Trying out a new program without having to find room on an Einstein floppy is a great advantage. Other advantages can be; a library of Albert's software could be put onto a CDROM, connecting Albert to the Web via a PC, Emailing files to others. And for those with scanners and optical character reading software getting hex listings from magazines (i.e. E/M) into a file is possible.

ASCII text has the ability to cross all platforms, and be viewed by text editors and word processors. Its disadvantage is that two ASCII characters are needed to represent one real byte - at least twice as big than needs to be on an Einstein disc - but for PC's 1.4mb floppies and hard drives these are small.

The basic features of SRLRUN

Only a hexadecimal (in capital letter) representation after a back-slash \ (this shows as a «, on the Einstein) and before the equals sign =, will be loaded as code. Only characters "0 1 2 3 4 5 6 7 8 9 : < > ? @ A B C D E F", will be loaded as code after the \. All lower case letters carriage returns and tabs will be ignored.

The second utility SRLLOAD.COM can be made from the assembly listing by changing line 40 to 40 *SAVE SRLLOAD.COM E900 EA34 and 1520 to 1520 .LEP JP &0000 then running.

The start address for a down loaded file can be changed by putting a ~ (this shows as a ö, on the Einstein) then immediately four hexadecimal digits for the Start Address. All other ASCII characters before the backstroke will be ignored. A text description for the hex dump and number of 256 byte blocks can be used before the "\" without fear of it being placed as machine code in the TC01.

When down loading to SRLRUN or SRLLOAD each byte installed in Albert's memory will be echoed back to the terminal screen.

BBCBASIC ASSEMBLY LISTING FOR SRLRUN

```
10 GOSUB 90

20 PRINT "START"

30 GOSUB 100

40 *SAVE SRLRUN.COM E900 EA34

70 STOP

90 BEGIN=&E900

100 CODE=&E900

110 P%=CODE

120 [OPT 1

130 .BEGIN LD BC,&0132; load BC the number of bytes to move.

140 LD DE,&E912; DE the address to move to.

150 LD HL,&0112; HL the address to move from

160 LDIR

170 JP LAD_ADDS; jump and run srlrun.com from its new location
&E916.

210 RET
```

```
220 .RN DEFW &0100; store address to download starting at.

230 .SAFE DEFW &E8FE; fail safe to stop downloading over running
SRLRUN.com.

240 .LAD_ADDS LD HL,(SAFE);

250 LD DE,(RN) ;

260 CALL CHK_IN ; a routine to clear serial port buffer.

270 .CHK_START CALL RTS_ON ; handshaking allow PC to download.

280 CALL RECEIVE ; a routine to wait for next byte from serial input.

290 IN A,(&10) ; put serial input into A reg.

300 LD DE,(RN) ; load DE with address to load at.

310 CALL RTS_OFF ; set RTS on to hold flow of serial input from PC.

320 CP 126 ; compare A to  ~" ascii 126

330 CALL Z,LAD_START ; routine to change DN store according to
next 4 hex ascii bytes from serial input.

340 CP 92 ; compare A reg with  /

350 LD DE,(RN) ; load DE new start address if changed.

360 CALL Z,LAD_PROG ; routine that converts and loads hex list into
Einstein.

370 JR CHK_START ; loop back

380 RET

390 .CHK_IN PUSH AF ; routine to clear bytes not wanted for
loading.

400 CALL RTS_ON

410 .CHKO IN A,(&10);clear buffer

420 IN A,(&11) ; read 8521A USART register.

430 BIT 1,A ; see if bit 1 = 0.

440 JR NZ, CHKO ; if not jump back and clear next byte.

450 POP AF
```

```
460 RET
470 .LAD_START PUSH AF ;routine to change address where code will be loaded.
480 CALL  GT_BYTE ; download next two ascii hex bytes convert to real byte
490 LD D,A ; real byte returned in A reg. then loaded into D reg.
500 CALL H_PRINT ; routine convert real byte back to ascii and echo to PC terminal.
510 CALL GT_BYTE ; get next real byte.
520 LD E,A ; put second byte into E reg.
530 CALL H_PRINT ; echo second byte to PC.
540 LD (RN),DE ; store DE for new address to down load to.
550 POP AF
560 RET
570 .LAD_PROG ADD A,00 ; routine to load code into Einstein.
580 DEC A
590 DEC A
600 PUSH HL
610 PUSH DE
620 SBC HL,DE ; check to see if loading over runs.
630 JR Z NO_START ; if so jump to abort
640 POP DE
650 POP HL
660 CALL GT_BYTE; GET TWO ASCI HEX BYTES FOR REAL BYTE
670   LD (DE),A ; returned byte in A reg. loaded at address given by DE
680   CALL ECO ; echo loaded byte back to PC terminal.
690   INC DE ; get next address to load next byte.
```

```
700  JR LAD_PROG ; loop back
710  RET
720  .NO_START RET ; fail safe finish.
730  .GT_BYTE PUSH HL ; the routine to download two hex ascii
bytes for real byte.
740  PUSH DE
750  CALL RTS_ON ; set handshaking on.
760  .BACK CALL RECEIVE ; routine to check if new byte has been
received in 8251A buffer.
770  IN A,(&10) ; returned new byte in A reg.
780  CP 71 ; compare A to G
790  JR NC,BACK ; if A greater than 71-ascii G or higher jump back
get next byte.
800  CP 48 ; compare A to 0
810  JR C,BACK ; if A smaller than 48 jump back and get next byte.
820  CALL RTS_OFF ; hardware handshaking to hold down load from
PC.
830  CP 61 ; compare A reg.(next byte) to  =
840  JR Z,INEND ; if so program down load complete jump to end.
850  CP &40 ; compare A to  @  ascii character before  A
860  JR C,WR1 ; jump if A value less than &40-for digits 1 to 9
870  AND &DF
880  SUB 07 ; standardize character code.
890  .WR1 ADD A,A ; shift A one hex digit left
900  ADD A,A ; so &30 becomes 00
910  ADD A,A
920  ADD A,A
930  PUSH HL ; store HL
```

```
940  LD L,00
950  LD H,00
960  LD H,A ; H=first hex digit * 16
970  CALL RTS_ON ; handshaking allows serial input
980  .BACK2 CALL RECEIVE ; checks for new byte received.
990  IN A,(&10) ; serial input returned in A reg.
1000  CP 71 ; compare A to G.
1010  JR NC,BACK2 ; if G or Higher jump back and get next byte.
1020  CP 48 ; compare A to 0.
1030  JR C,BACK2 ; if not 0 and ascii characters less than 48.
1040  CALL RTS_OFF ; handshaking hold serial input.
1050  CP 61 ; compare A reg. to  = .
1060  JR Z,INEND ; if so program download complete jump to end.
1070  CP &40 ;compare A to  @  ascii character before  A
1080  JR C,WR2
1090  AND &DF
1100  SUB 07
1110  .WR2 AND &0F ; consider second hex digit only.
1120  OR H ; combine with first hex digit-A reg now holds value of
two ascii hex bytes.
1130  POP HL
1140  POP DE
1150  POP HL
1160  RET
1170  .ECO PUSH DE ;routine to echo byte loaded back to PC
terminal.
1180  PUSH HL ; store HL
1190  LD H,D:LD L,E ; put address held in DE into HL.
```

```
1200  PUSH AF
1210  LD A,(HL) ; load last transferred byte into A reg.
1220  CALL H_PRINT ; routine to serial output A reg. value.
1230  POP AF
1240  POP HL ; restore HL value.
1250  POP DE
1260  RET
1270  .RECEIVE PUSH AF ; routine to check for new byte in 8251
USART.
1280  .CHK IN A,(&11)
1290  BIT 1,A ; bit 1=1 if fresh byte received.
1300  JR Z, CHK ; if bit 1=0 go back for new byte
1310  POP AF
1320  RET
1330  .RTS_ON PUSH AF ; hardware handshaking routine.
1340  LD A,&27 ; bits 0,1,4 and 5 set. bit 5 is request to send
enabled.
1350  OUT (&11),A ; out A to 8521A reg.
1360  POP AF
1370  RET
1380  .RTS_OFF PUSH AF ; hardware handshaking routine to hold
serial out put from PC.
1390  LD A,&07 ; bits 0,1,2 set and bit 5 reset to 0
1400  OUT (&11),A ;out A to 8521A reg.
1410  POP AF
1420  RET
1430  .END2 POP HL
1440  .INEND CALL CHK_IN
```

```
1450  POP HL ; restore as jumped out of loop.
1460  POP DE; restore as jumped out of loop.
1470  POP AF; restore as jumped out of loop.
1480  LD A,64; load A  ascii  @
1490  CALL SEND; send to pc terminal to denote download
complete.
1500  LD HL,&0100 ;
1510  LD (RN),HL ; reset load start address
1520  .LEP JP &0100 ; jump to run com prog.-change to hex 0000
to reboot and not run.
1530  RET
1540  .H_PRINT PUSH AF ; routine to turn real byte into two hex
ascii bytes.
1550  AND &F0 ; isolates first digit.
1560  RRA ; move this
1570  RRA ; digit to
1580  RRA ; its proper position
1590  RRA ; in A reg.
1600  ADD A,&30 ; change to ascii character.
1610  CP &3A ; is digit between A and FOR
1620   JR C,HP_H ;
1630   ADD A,07 ; change to correct symbol if so.
1640   .HP_H CALL SEND ; serial output byte.
1650   POP AF ; retrieve original value AND
1660   AND &0F ; isolate second digit.
1670   ADD A,&30 ; change to ascii value for character.
1680   CP &3A ;
1690   JR C,HP_L
```

```
1700    ADD A,07 ; change to correct hex symbol.
1710    .HP_L CALL SEND ; serial output AND
1720    RET
1730    RET
1740    .SEND PUSH AF ; store A value
1750    .REP IN A,(&11)) ; read 8251A usart reg.
1760    BIT 0,A ; bit 0=1 if new byte ready to send.
1770    JR Z  REP ; jump back if byte not ready.
1780    POP AF ; restore A
1790    OUT (&10),A ; serial output A
1800    RET
1810    ]
1820    RETURN
```

When the ascii text sent by READCOUT.BBC files are viewed with a PC Editor i.e. Notepad or Wordpad only the hex numbering after the \ (shown as « on the Einstein) and before the = will be loaded as file. If a ~ (this shows as a ö on the Einstein) is used before the \ the following four hex digits will be the start address where the binary code will be loaded in the TC01.

For Example: BBCBASIC files are binary files which load in at Hex 4000 after BBCBASIC.COM has been loaded. From bbcbasic use *LOAD SRLRUN.COM E900 to load SRLRUN as machine code then CALL &E916 to run. The ~4000 viewed in the text file before the \ will load the code starting at hex 4000. When the BBCBASIC heading appears again type the commands OLD ENTER then LIST ENTER.

All other text before the \ besides ~**** will be ignored by SRLRUN.COM and SRLLOAD.COM.

Other ways to send SRLRUN or SRLLOAD hex files from the PC.

From the PC dos prompt I have used the command MODE COM1:9600,N,8,2 to set compatible baud rates with the TC01 defaults then COPY <filename.ext> COM:1 to send files from the 286 & 486 PCs. Ironically using the K6 2 475 PC the files became corrupted at 9600 baud. Slowing the baud rate to 4800 worked ok. Using 9600 baud then ms dos s EDIT.COM to load files and using settings to set the printer to COM1 worked with all PCs. The hardware handshaking then seems to take affect.

BBCBASIX

There are some sites on the web with CP/M utilities that can be down loaded. These are stored as binary files and naturally can't be run on a PC as they are intended for Z80 computers. BBCBASIX is a freeware BBCBASIC program for the PC (can be down loaded from www.bbcbasic.com). I have used the READCOUT.BBC listing in BBCBASIX to send binary files to SRLLOAD in Albert. At 9600 baud this worked ok with the 286 & 486 computers but, again ironically, needed 4800 baud for the K6 2 475. To save changing settings for my faster PC I have adapted the BBCBASIX listing to include hard ware hand shaking. Programming the PC's 16550 UART is complicated to grasp, but the adapted listing below seems to be working ok. The hand shaking is in lines 171 to 173. Also added is a count for sent bytes line 170.

```
 10 REM READCOUT.BBC for PC BBCBASIX

 20 REM to open a Z80 com-file and serial output it in hex.

 30 REM to srlrun.com on the Einstein

 40 PRINT:PRINT

 50 *MODE COM1:9600,N,8,2

 60 *.*.*

 70 PRINT"TYPE FILE NAME TO READ"

 80 INPUT A$:PRINT A$

 90 X=OPENIN(A$)

100 L=EXT#X:PRINT,"LENGTH ";L,"BLOCKS ";L/256

110 PRINT"FILE NUMBER ";X

120 *OPT 1

130 PRINT "zilog com file to output in hex \":REM \ needed to start
loading.

140 FOR I=1 TO L
```

```
150 Y=BGET# X
160 *OPT 0
170 PRINT TAB(VPOS+13,POS+22);" "; I
171 N=GET(&03FE)
172 BT=N AND 16
173 IF BT =0 GOTO 171
180 *OPT 1
190 IF Y<&10 THEN PRINT"0";
200 PRINT;~Y;
210 NEXT I
220 PRINT"="
230 *OPT 0
240 CLOSE# X
250 PRINT:PRINT"FILE ";A$,"FILE NUMBER ";X,"LENGTH ";L
```

The CP/M Main Page "http://www.seaship.demon.co.uk" seems to have all the links for down loading CP/M files. The search engine Altavista "http://www.altavista.com/" finds it easily.

I have transferred BBCBASIC & XBAS to the PC and back again many times without any mishaps. I've also used the PC BBCBASIX READCOUT to transfer many CP/M 2.2 COM files down loaded from the net. Bit of a jungle here! Lots of COM files listed but no instructions. For those that down load and appear to work in Albert knowing how to use them is the problem.

Further development for SRLRUN & SRLLOAD.

There are many features I thought of adding to the program but keeping it small in the Einstein's limited memory seemed important. The program is not relocate able so its final position at E912 to EA34 (called from E916) is as high in memory, I found, I could go without over running the stack. The reason for the long assembly listing, instead of just the hex dump for the MOS editor, is so others can make versions, which locate the code in different areas of memory. One feature I think should be added is a way to change the final jump address from the down loading file. This would be ideal for chain loading files. Example down loading BBCBASIC.COM then jumping back to E916 SRLRUN again to down load a bbc file at hex 4000.

I haven't tried down loading another operating system yet so if any one has a hex dump of CP/M+ or ZDOS and has some idea where to load it into Albert's memory please Email it to chris@coxa.fsnet.co.uk.

For faster down loading it should be possible to wire the standard PC printer port to Albert's user port. If anyone has already done this, I would be grateful for the wiring details.

Tel: 01322 346102 - email chris@coxa.fsnet.co.uk

Ed: This concludes the article, but it is evident there's much more to this, being worked upon by Chris right now. Give him some feedback - you too, may well benefit.

----@@@----

Hello - was anybody there?

John Marriott © July, 2001

Nothing to do with computers, or is it? Can somebody explain to me where the 'Base of 6' or the 'Base of 12' came from - when obviously the 'Base of 5' or the 'Base of 10' automatically springs to mind?

The reason I'm curious is that back in our dim distant history there must have come a time when some form of 'mathematics', or addition, became a 'must' - you know, when a 'flock of sheep' got past that 'looks about right' to 'I know it's right'. A simple idea would be to have as many pebbles in a pouch as there were sheep, but that ain't 'based' on anything other than a 'gigantic leap' of 'associated comparison'...

...rather like handing somebody a bit of coloured paper in exchange for their lump of gold?

Okay - it's easy enough to see that our word 'dozen' came from the French word 'douze', a 'cloth yard' originated from he long bow arrow - nocked and stretched, but what about a 'fathom', or even 'poles' and 'perches' - all subject to being 'integer' divided by 6 when everything logically points to 5...

...yet not! Old 'tally markings' all show 'the uprights with a diagonal cross through', even as children (and not so children) we tally that way - a 'five barred gate', why not a "six...", so just where or when did this 'anomaly' come into being?

Don't be fooled with today's 'Metric' system - that's only been going a smidging of a time, an artificial system if ever there was. Our old 'Farthing' was more logical - a 'Penny' halved and halved. Cut out a circular disk of paper, fold it in half, than half again - so logical, and have you spotted it - the basis of 'Binary'!

So why 12 segments to a clock face, 60 minutes to the hour, 60 seconds to the minute - 360 degrees to the compass circle, 60 seconds to a degree, when 'our' logic suggests that 'Binary' is an easier 'division' system.

Yes, there are 6-toed cats - even humans with 6 fingers, but these appear to be genetic throwbacks, hiccups, or are they?

Scientists now believe the Moon is a 'captured' asteroid - but what if it was a 'placed' asteroid with the idea of 'terra forming' Earth? If you compare the 'time life span' of a human as to a butterfly, why not a 'being' to a human life span comparison - the Old Testament certainly gives credence to that idea.

Genetically it is well nigh impossible to 'transplant' tissue from one species to another (but don't hold your breath, the way things are going...), but 'genetic manipulation' is proving to be more than possible - and a 'being' would tend to take that route if it wanted to 'expand' into a 'fairly suitable' planet environment - Earth.

Of course, you'd need to do a bit of genocide with the existing 'top of the pyramid' species e.g. the Dinosaurs - and it's not very hard to get a comparison way of how the North American Indians were decimated - slaughter their food/materials supply to near extinction, the buffalo! Of course, if the 'being' didn't want its 'ancestors' to know i.e. autopsies showing bullet holes, lazer charring, things like that - then starvation of the large herbivores is the route, temperature another, with the 'food chain' carnivores quickly following the 'starvation extinction' route...

...which is where the Moon came in! Pre-dawn history the diameter of the Earth was greater (gravity has been compressing it, still is - where do you think we're getting our volcanoes and earth quakes from?) so would have rotated slower than it does today - simple 'bobweight' and 'flywheel' effect - so the Moon could have been 'placed' so that a permanent 'solar eclipse' affected one portion of the Earth - a 'land bridge' corridor that the herbivores needed to cross, feed, on their annual circulatory march - changed to a mini Arctic?

Now, this is where the computer comes in - can somebody create a program to work out the Earth's 'shrinkage', rotational slowing, the 'geo-sync' of the Moon to Earth...

...and just how old my Mother-in-Law really is!

----@@@----

EINSTEIN MAGAZINE and all micro news No.103

Would you like some FREE MEMBERSHIP? The contributors to this issue have been credited with one free magazine issue on their membership subscription for each item published.

Why not join their ranks and get some free membership too?

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

Or enter our multiple-question COMPETITION instead. As well?

COMPETITION.....COMPETITION.....COMPETITION.....COMPETITION..

On the front cover we bring you the latest development in railway safety - mechanical signals to control the passage of trains and supersede the practice of uniformed policemen giving handsignals at intervals along the lineside!

As you will have read in your newspapers recently, after many years of operating with mixed gauge, the Great Western Railway has finally taken the inevitable step of abolishing Mr Brunel's broad gauge track entirely, and it now operates its trains entirely on Mr Stephenson's narrow gauge track.

1. What was the distance between the rails of Mr Brunel's track? 2. What was the distance between the rails of George Stephenson's track originally? 3. What is it now? 4. Which was the last section of track that was broad gauge only? 5. When did the last broad gauge train run over it? 6. What was the name of the guard in charge of the last train over the last broad gauge section of track? 7. How do you know his name? 8. What was he required to do at each station that was unusual, in working this last broad gauge train? 9. How many broad gauge engines have survived? 10. How many broad gauge engines currently exist? 11. What's the difference?

(All entries to Tony at New Romney Einstein World HQ please)

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

BACK NUMBERS: We have lots of these and need to reduce our stock as they take up far too much space. From EM vol 1 no.4 on, send a 1st class stamp for each issue you need, or send a cheque payable to Einstein User Group to New Romney, at only 4 for £1. Don't forget to include your name / address!

H.

AND COMING SOON: A clear-out of surplus Einstein user manuals and hardware, plus other computer hardware / software / books. WATCH THIS SPACE for more details.