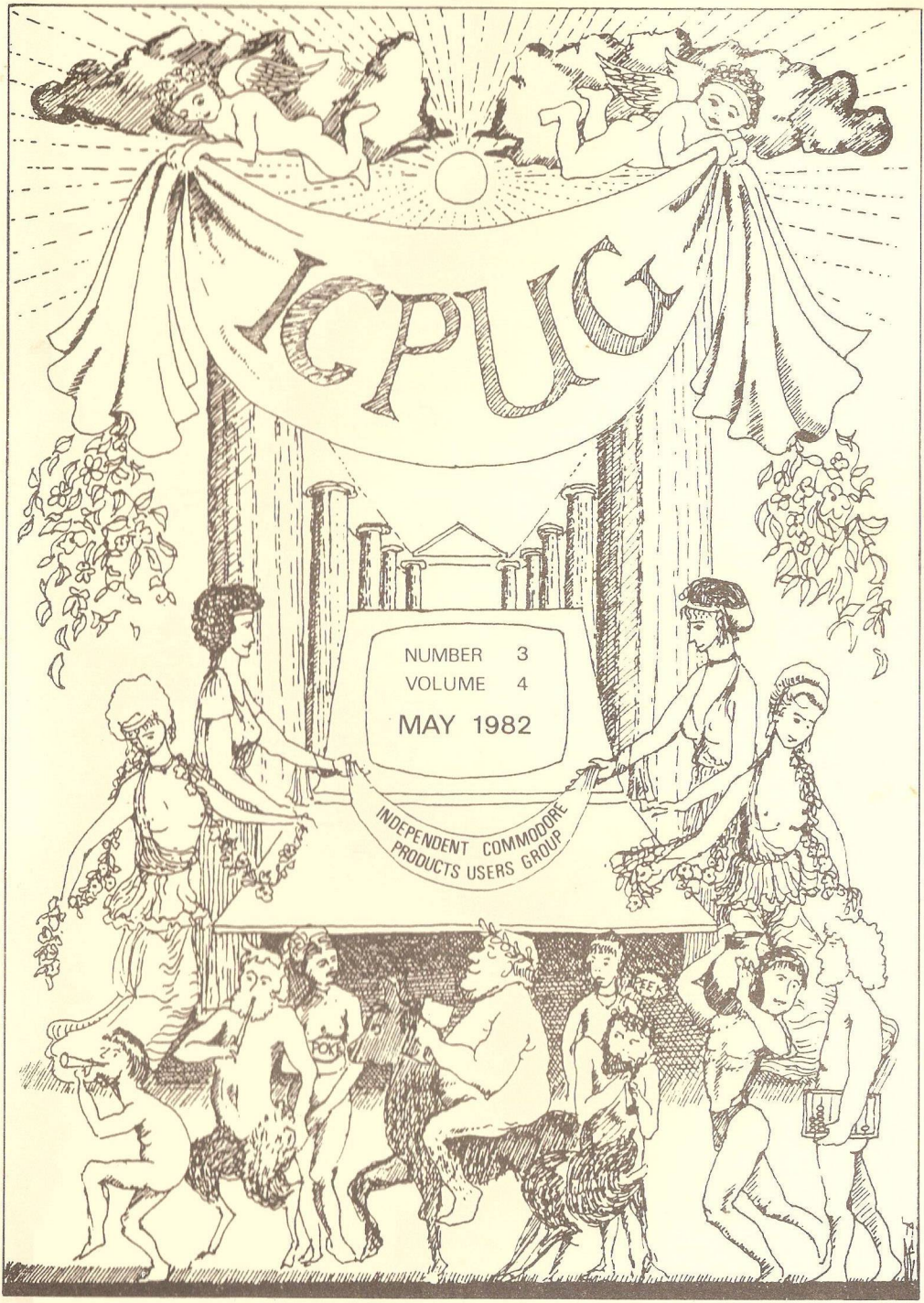

ICPUG
NUMBER 3
VOLUME 4
MAY 1982
INDEPENDENT COMMODORE PRODUCTS USERS GROUP
POKE
PEEK

HONORARY NATIONAL OFFICIALS

Chairman:	Wing Cdr. Mick Ryan 164 Chesterfield Drive, Riverhead, Sevenoaks, Kent TN13 2EH Telephone: Sevenoaks (0732) 453530
General Secretary:	Jim Tierney 11 Collison Place, Tenterden, Kent TN30 7BU Telephone: 058-06 2711
Regional Co-ordinator:	Eli Pamphlet 7 Lower Green, Tewin, Welwyn Garden City, Herts AL6 0JX Telephone: Welwyn (043 871) 7325
Treasurer:	Joseph Gabbott
Software Librarian:	Bob Wood 13 Bowland Crescent, Ward Green, Barnsley, South Yorks S70 5JP Telephone: (0246) 811585 (work) (0226) 85084 (home)
Membership Secretary:	Jack Cohen 30 Brancaster Road, Newbury Park, Ilford, Essex IG2 7EP Telephone: 01-597 1229
Editor:	Ron Geere 109 York Road, Farnborough, Hants GU14 6NQ
Sub-Editor:	Mike Todd 27 Nursery Gardens, Lodgefield, Welwyn Garden City, Herts AL7 1SF
Publicity Officer:	Alan Birks Flat 135, Queen Alexandra Mansions, Judd Street, London WC1H 9DQ

INDEPENDENT COMMODORE PRODUCTS USERS GROUP

Vol. 4. No. 3 **Newsletter** May 1982

Europe's first independent magazine for PET users

Page	Contents

EDITOR'S NOTEBOOK

Yes, I know the cover said March and the contents page said April. Cryptologists among you immediately realised the significance of the '104' (p96) and that Rolf Pialo was an anagram and not a Commodore employee and was also the leading character in the last 9 paragraphs (although somehow two became merged). However the joke nearly back-fired since the article was written before we got news of the range of new products, and it was too near to the truth and had to be hastily up-rated. Never-the-less, it caused a flurry of activity among Commodore management until the penny dropped, and even our worthy Chairman spent many a happy hour cataloguing the article for 'The PET Index'...

ICPUG put in an appearance at two exhibitions in a short space of time, & thanks are due for the hard work and time devoted to supporting our exhibition stand. Again we shall be represented at the Commodore Computer Show and request that volunteers to staff the stand offer their services to our exhibition co-ordinator Stephen Rabagliati on 0442-44743 at home, or (09276) 7141, xtn 39 at work. If you have yet to visit the Commodore Show, it is not like other exhibitions where perhaps only 10-20% of the stands are of interest, I find the figure nearer 95%.

As the Group expands and the services and facilities for members increase, so does the workload of the officers of the Group. If you feel that you can contribute in some way, please do not hesitate to contact any member of the Committee. For example, the Shop Window column has covered many add-ons, add-ins and useful bits and pieces for the user, perhaps someone would like to undertake to compile an up-to-date list of products and suppliers based on this on-going series. I can arrange for publication of the resulting text if required. No doubt other ideas will come to mind, for as a user group there is always much to be done to help other users. How about some more help on the VIC side of interests ?

R.D.G.

--oOo--

UNIVERSAL DISK RENAME

By Brian Grainger

Back in the dim and distant past, IPUG Vol.2 No.3, a Butterfield routine to retitle disks was published. Following a request from a correspondent, I have updated the routine for use on any 2040, 3040, 4040, 8050 disk unit.

This routine, like Butterfields, allows you to modify disk name and identity. I have been informed from other sources that one should not change the identity. I have been doing this for some time with no problems but you have been warned ! Here is the routine:

```
010 REM ***************************
020 REM UNIVERSAL DISK RENAME
030 REM B.D. GRAINGER FEBRUARY 1982
040 REM ***************************
100 PRINT"<clr>PLEASE INDICATE WHICH DOS YOU USE":PRINT
110 PRINT"TYPE <rvs>A<off> FOR DOS 1 OR DOS 2.1"
120 PRINT"TYPE <rvs>B<off> FOR DOS 2.5":PRINT
130 GETA$:IFA$=""GOTO130
140 IFA$="A"THENTR=18:NS=144:IS=162:D$="1 OR DOS 2.1"
150 IFA$="B"THENTR=39:NS=6:IS=24:D$="2.5"
160 IFA$<>"A"ANDA$<>"B"GOTO130
170 PRINT"<clr>YOU USE DOS ";D$:PRINT:PRINT"PUT DISK
    REQUIRING RENAMING IN DRIVE 1":PRINT
180 PRINT"PRESS 'RETURN' TO CONTINUE":PRINT
190 GETA$:IFA$<>CHR$(13)GOTO190
200 OPEN15,8,15:PRINT#15,"I1":OPEN2,8,2,"#"
210 PRINT#15,"U1:"2,1,TR,0
220 D$="NAME":B=NS:R=16:GOSUB500
230 D$="ID":B=IS:R=2:GOSUB500
240 IFV$=""THENV$="NOTHING":GOTO270 260
250 PRINT#15,"U2:"2,1,TR,0:PRINT#15,"I1"
260 CLOSE2:PRINT"* ";V$;" CHANGED *"
261 REM ****************************
262 REM THIS END ROUTINE IS WRITTEN
263 REM TO ENABLE THE PROGRAM TO BE
264 REM RUN MORE THAN ONCE. NORMALLY A
265 REM 'NO CHANNELS' ERROR OCCURS(?)
266 REM THE DISK IS RESET (U:) TO
```

```
267 REM SOLVE THIS BUT THE 3040 THEN
268 REM NEEDS REINITIALISING (I)
269 REM THE ERROR CHANNEL IS READ
270 REM TO REMOVE THE ERROR WHEN ONLY
271 REM 1 DISK IS IN THE DRIVES WHEN
272 REM THE DISK IS REINITIALISED.
273 REM ****************************
280 PRINT#15,"U:":PRINT#15,"I":INPUT#15,EN,ED$,T,S:
    CLOSE15
290 END
500 REM ************************
501 REM MODIFY CHANNEL 2 ROUTINE
502 REM B= 1ST BYTE TO ALTER
503 REM R= NO. OF BYTES TO ALTER
520 REM ************************
530 PRINT#15,"B-P:"2,B
540 PRINT"OLD ";D$;" IS ";CHR$(34);
550 FORJ=1TOR:GET#2,A$:PRINTA$;:NEXT:PRINTCHR$(34):
    PRINT
560 PRINT"ENTER NEW ";D$:PRINT"PRESS 'RETURN' FOR NO CHANGE"
570 INPUT"<2rt>*<3lft>";X$:PRINT:IFLEN(X$)>RTHENPRINT"TOO
    LONG !":PRINT:GOTO560
580 IFX$="*"THENRETURN
590 IFLEN(X$)<RTHENX$=X$+CHR$(160):GOTO590
600 V$=V$+W$+D$:W$=", "
610 PRINT#15,"B-P:"2,B:PRINT#2,X$;
620 RETURN
```

--o0o--

REVIEW

Asteroids — Supersoft £ 8

After Space Invaders, Asteroids was probably the most popular arcade game. I have always moaned about the lack of high resolution graphics on PET, so making this game impossible to put on this machine. That was without the ingenuity of today's software writers. I would never have thought it possible but SUPERSOFT are now selling an excellent version of Asteroids for the PET. With judicious use of the graphic characters available a very good representation of the original is obtained without the need of high resolution graphics.

For those who do not go out and have not seen Asteroids, the aim is to break up and obliterate chunks of rock while avoiding them crashing into you. You have a gun which you can rotate in either direction or thrust through space and of course a fire button. Should things get desperate there is a hyperspace button which when pressed will cause your gun to disappear and reappear somewhere else in space. After clearing one screenfull of moving rocks you will get more coming at you. You have three lives with extra lives thrown in if you score enough. As with all SUPERSOFT arcade games there are sound effects and the game works on all PETs except old ROM machines. There are nine levels of play, which will allow beginners to the game to get used to all the controls, not easy after being used to Invader type games. For those who are used to the game you should really play at the top level (9) to make it interesting.

As I said earlier this is an excellent version of the original, only two points to make. In the original when one moved the gun through space it would not stop until you had reversed the direction of thrust and jetted again. Just like real space. The SUPERSOFT version stops the gun as soon as the thrust button is released. Also on the original, when one left the side of the screen one reappeared on the other. On the SUPERSOFT version one just stops on the edge of the screen. Despite these differences from the original SUPERSOFT Asteroids is an excellent game and well worth the cost of £ 8.

I must finish by giving credit to SUPERSOFT at the level of protection against copying on this program ! Every trick in the book has been used and only the more knowledgeable enthusiast will break it. The only trouble is that those with disk drives cannot transfer it to disk! For those who are thinking of writing to ask me how to remove the protection - DONT - pay your £ 8. It is well worth it!

By Brian Grainger

--o0o--

COMMODORE COLUMN

As competition in the microcomputer market hots up Commodore have announced price cuts to educational establishments during March and April. The 4016 dropped from £ 550 to £ 399, the 4032 from £ 695 to £ 499 and the 2031 single disk drive from £ 395 to £ 299. These price adjustments strengthen Commodore's position against the BBC microcomputer.

Commodore has abandoned a project to produce a rival to SSE's CP/M Stunt Box. This arises from a change in management and a new policy to make everything itself. As a result more new machine models than ever before are to be introduced in 1982. No doubt these will be unveiled in the UK at the Third International Commodore Computer Show (see details elsewhere).

Down south viewers of Southern TV franchisee TVS series 'The Real World' were treated to a program in which twelve families were loaned a VIC-20 for two months and then invited to discuss how the computer had affected their lives and what they achieved by having a computer in the home. Not only were reactions overwhelmingly favourable, but Commodore received over 1,000 viewer enquiries as a result of the programme.

A £ 1,500 program writing competition open to all (except Mike Todd) has been launched. Submissions are invited for any type of program to run on VIC-20 or 4000 series machines utilising up to 32K RAM. Preferred entries will be both inventive and instructive and designed for educational or home application. Multiple submissions on cassette or disk are permissable. Judges include Commodore's Technical Manager, a leading educational computer consultant and our ubiquitous Mike Todd. First prize comprises VIC single disk drive, printer and Programmers Aid cartridge. Second prize is the disk drive and Programmers Aid, third is the printer and Programmers Aid. Runners-up prizes bring the total prize fund to over £ 1,500. The closing date is June 30th, 1982 and entries

must be accompanied by the appropriate magazine coupon and sent to Commodore Software Competition, 35, Garway Road, London, W2 4QD.

Commodore have made a bold decision to stay with 8-bit microprocessors for their current developments, by upgrading the 6502 to the new 6509 and 6510 processors. These are faster versions of the 6502 with extended addressing capability. The justification is that a large number of programmers can write 8-bit software and a much 6502-based software already exists. 16-bit competitors reckon the Commodore argument is misplaced. we shall have to wait and see.

The communication network Petnet looks like being followed by a VIC-20 version called Vicnet. The networks are mainly a software development and testing environment for users wanting to exchange software and are due to be provided later this year.

Although in some ways in competition with Commodore, ACT of Birmingham plans to sell an emulator package for about £ 900 to PET users which will allow them to link into an ICL mainframe based bureau service by either dial-up telephone service or leased line. This will enable users to access much greater processing power for commercial accounting for example. On the subject of ACT, it appears that they have been taken to court regarding a contractual dispute over royalties for David Levy's chess program.

--o0o--

NEW RAMS FOR OLD

By Malcolm Pritchard

The author's 2001 8K PET (the one with the calculator-style keyboard) has reached its third birthday and is beginning to show its age. On two occasions the power-up 'BYTES FREE' message has indicated failure of one of the 6550 memory i.c's. This particular static RAM is not easy to find and the current replacement cost is over £ 17 each !

Optimized Data Services of Placentia, California offers a '2114 RAM Adapter' which fits into two of the 22-pin 6550 sockets and allows up to eight 6550's to be replaced by cheaper, more reliable, 2114L dynamic RAMs. The adapter is available as a bare printed circuit board ($11.95), as a kit minus 2114's ($17.95) or as an assembled, tested unit minus 2114's for $24.95. The fully assembled version is also available from Supersoft in the UK for £ 40 complete with 2114's.

A decision was made to order the kit by air mail from California, paying by credit card and three weeks later a small parcel arrived. This contained a printed circuit board, two 22-pin wire-wrap i.c. sockets, one 16-pin i.c. socket, a 74LS139, eight 18-pin i.c. sockets and two 0.1 µF capacitors. The instructions specified low-power 2114's (2114L) with access time of 450nS or faster. 300nS types were obtained from Watford Electronics at 87p each.

Construction was very straightforward - if you can solder i.c. sockets there should be no problems. The mounting position of the decoupling capacitors was not marked clearly but was obvious from the circuit diagram. The unit was built and installed in the PET in less than one hour. Initial testing is carried out with the 6550's taken from column 8 of the PET board and placed in the 22-pin sockets of the RAM adapter. Subsequently, up to eight 6550's can be replaced by 2114's, one by one if required. There are 16 6550's in the 2001-8K PET's main memory and replacement of all these requires a second RAM adapter. Total cost of two adapter kits plus air mail is $43.40 and sixteen 2114L's cost £ 14. Compare this with £ 270 for sixteen 6550's !

All prices quoted are without VAT. Addresses: Optimized Data Systems, PO Box 595, Placentia, CA 92670, USA. Tel: (714) 9963201. Supersoft, First Floor, 10-14, Canning Road, Wealdstone, Harrow, HA3 7SJ. Tel: 01-861 1166. Watford Electronics, 33-35, Cardiff Road, Watford, Herts, WD1 8ED.

--o0o--

THE SOFTWARE LIBRARY

In recent months the software library has grown enormously and in consequence the library adminstrators have been hard pressed to keep up with things. It follows therefore that the recently reprinted list from my files has been hopelessly out of date. The following items should be noted:

1) Additions to the library as follows:

Disk 4 Music 1	Disk 5 Music 2
Disk 6 Games	Disk 7 Utilities
Disk 9 Graphics	Disk 10 Utilities
Disk 11 Education 2	Disk 12 Education1
Disk 13 Business	

[What happened to #8 ? - Ed]

2) Will members requesting software by cassette please limit your choice of programs to a maximum of four per request. This is purely a matter of the time it takes to save and verify programs on cassette as opposed to disk.

3) VIC-20 software should be available by the time you read this.

Finally please note my correct telephone numbers which were incorrect inside past covers: Work (0246) 811585; Home (0226) 85084.

Bob Wood

--o0o--

ROUND THE REGIONS

Brian Grainger tells me that the North Herts regional group recently had a demonstration of the Supersoft high resolution graphics board. Program problems are tackled with Paul Mortiboy, the duo currently developing a program to take a disk and create a back-up copy of all program files on cassette. Ultimately it is proposed to get a debugged version running under RABBIT as a very fast disk back-up facility.

--o0o--

ANOTHER MERGE

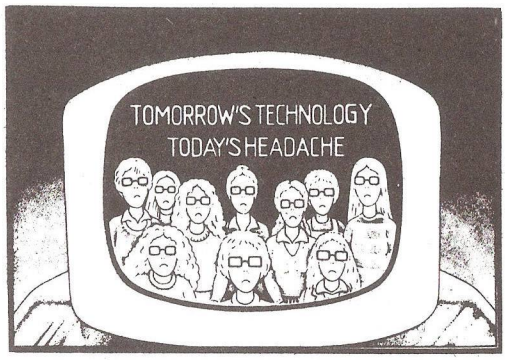
From time to time a number of MERGE routines have been published in the Newsletter, the last one of which (p70 May '81) merged two files on disk leaving the resultant file on disk. When there is a requirement to merge several modules with perhaps intermediate stages of renumbering, the disk soon becomes crowded with superfluous files. The following program merges a file on disk to the program in memory, leaving the result in memory.

The program gets a program line from disk and 'inserts' it in the correct place in memory, continuing until the end of file. Before calling, the file must be open so that the disk is set as a talker for channel #2, hence to merge, do OPENx,8,2,"dr:Filename,P,R":SYS28672 and hope for the best.

Lines 20-24 set the disk as talker and set up channel #2 for load. (This presumes the foregoing OPEN). Lines 26-31 get the first two bytes (LOAD start address) and if it is $0400, it reads a third byte. Lines 39-40 read the link bytes of the line and ignore them. If EOI is detected (lines 51-52) then the terminal sequence at 63-65 is entered. If end of line ($00) is detected at 59-61 then go to the insert routine (73 on), else get next character.

Lines 73-118 checks for duplicate line and if so deletes old line; lines 127-153 insert new line into text. Pointers and links are adjusted at 154-156 where the routine loops for another line. All operating system routines are for BASIC2. For BASIC4 the following equates should be used:

```
READY  = $B3FF
FNDLIN = $B5A3
RUNC   = $B5E9
LNKPRG = $B4B6
BLTU   = $B350
TALK   = $FOD2
UNTALK = $F1AE
IEEEIN = $F1CO
SACLOS = $F733
COMOPB = $F143
```



```
MERGEX.SRC......PAGE 0001
LINE# LOC   CODE          LINE
0001  0000              ;*******************************
0002  0000              ;*
0003  0000              ;* MERGE PROGRAM FROM DISK
0004  0000              ;*
0005  0000              ;*******************************

0007  0000              READY  = $C389     ;RTN TO "READY" LOOP
0008  0000              ISCNTC = $FFE1     ;CHECK STOP KEY
0009  0000              FNDLIN = $C52C     ;FIND LINE IN TEXT
0010  0000              RUNC   = $C572     ;RESET & CLR
0011  0000              LNKPRG = $C442     ;REGENERATE LINKS
0012  0000              BLTU   = $C2D8     ;OPEN UP SPACE
0013  0000              TALK   = $FOB6     ;SEND TALK ADDRESS
0014  0000              UNTALK = $F17F     ;SEND UNTALK
0015  0000              IEEEIN = $F18C     ;GET BYTE FROM BUS
0016  0000              SACLOS = $F6F4     ;SEND "CLOSE" SA
0017  0000              COMOPB = $F128     ;SEND SEC AD/CLR ATN

0019  0000                     *=$7000
0020  7000  A9 08       MERGE  LDA #$08
0021  7002  85 D4              STA $D4     ;SET DEVICE NUMBER
0022  7004  20 B6 FO           JSR TALK    ;SET DISK TO TALK
0023  7007  A9 62              LDA #$62
0024  7009  20 28 F1           JSR COMOPB ;SEND CH 2 SEC ADD

0026  700C  20 8C F1           JSR IEEEIN
0027  700F  08                 PHP         ;SAVE STATUS BYTE
0028  7010  20 8C F1           JSR IEEEIN ;GET START ADDRESS
                                          ;& IGNORE
0029  7013  28                 PLP         ;RECOVER STATUS
0030  7014  DO 03              BNE LINEIN ;IF LOAD FROM $0401
0031  7016  20 8C F1           JSR IEEEIN ;SKIP EXTRA BYTE

0033  7019  A9 04       LINEIN LDA #$04
0034  701B  85 05              STA $05     ;CHAR COUNT
                                          ;(INC HEADER BYTES)
0035  701D  A9 00              LDA #$00
0036  701F  85 FB              STA $FB     ;SET PTR TO BUFFER
0037  7021  A9 02              LDA #$02
0038  7023  85 FC              STA $FC
0039  7025  20 8C F1           JSR IEEEIN
0040  7028  20 8C F1           JSR IEEEIN ;GET LINK BYTES
                                          ;& IGNORE
```

```
0042  702B  20 8C F1          JSR IEEEIN
0043  702E  85 11             STA $11
0044  7030  20 8C F1          JSR IEEEIN
0045  7033  85 12             STA $12       ;GET LINE # TO 11/12

0047  7035  20 E1 FF    CHARIN JSR ISCNTC   ;CHECK STOP KEY
0048  7038  20 8C F1          JSR IEEEIN    ;GET LINE CHAR
0049  703B  A0 00             LDY #$00
0050  703D  91 FB             STA ($FB),Y ;SAVE CHAR IN BUFFER
0051  703F  24 96             BIT $96
0052  7041  70 0D             BVS ENDX      ;IF EOI RECEIVED

0054  7043  E6 05             INC $05       ;COUNT CHARACTERS
0055  7045  E6 FB             INC $FB
0056  7047  D0 02             BNE NOOVF
0057  7049  E6 FC             INC $FC       ;BUMP POINTER

0059  704B  A8          NOOVF  TAY
0060  704C  D0 E7             BNE CHARIN    ;IF NOT 00 THEN
                                            ;GET NEXT
0061  704E  F0 09             BEQ LINEND    ;IF END OF LINE

0063  7050  20 7F F1    ENDX   JSR UNTALK   ;UNTALK IEEE BUS
0064  7053  20 F4 F6          JSR SACLOS    ;CLOSE FILE
0065  7056  4C 89 C3          JMP READY     ;READY

0067  7059                    ;*******************************
0068  7059                    ;*
0069  7059                    ;* LINE ASSEMBLED - NOW INSERT
0070  7059                    ;*
0071  7059                    ;*******************************

0073  7059  20 2C C5    LINEND JSR FNDLIN   ;FIND LINE IN TEXT
0074  705C  90 44             BCC NODEL     ;IF NOT FOUND

0076  705E  A0 01             LDY #$01
0077  7060  B1 5C             LDA ($5C),Y ;GET LINK HI BYTE
0078  7062  85 20             STA $20
0079  7064                    ;
0080  7064  A5 2A             LDA $2A
0081  7066  85 1F             STA $1F       ;START OF VAR (LO)
0082  7068  A5 5D             LDA $5D
```

```
0083  706A  85 22              STA $22     ;PTR (HI) TO
                                          ;CURRENT LINE
0084  706C  A5 5C              LDA $5C     ;LINE PTR (LO)
0085  706E  88                 DEY
0086  706F  F1 5C              SBC ($5C),Y ;SUBTRACT LINK LO
0087  7071  18                 CLC
0088  7072  65 2A              ADC $2A
0089  7074  85 2A              STA $2A     ;NEW START OF
                                          ;VARIABLES LO
0090  7076  85 21              STA $21
0091  7078                 ;
0092  7078  A5 2B              LDA $2B
0093  707A  69 FF              ADC #$FF
0094  707C  85 2B              STA $2B     ;NEW START OF
                                          ;VARIABLES (HI)
0095  707E                 ;
0096  707E  E5 5D              SBC $5D     ;SUBTR LINE PTR HI
0097  7080  AA                 TAX
0098  7081  38                 SEC
0099  7082  A5 5C              LDA $5C
0100  7084  E5 2A              SBC $2A
0101  7086  A8                 TAY
0102  7087  B0 03              BCS QDECT1
0103  7089  E8                 INX
0104  708A  C6 22              DEC $22
0105  708C  18           QDECT1 CLC
0106  708D  65 1F              ADC $1F
0107  708F  90 03              BCC MLOOP
0108  7091  C6 20              DEC $20     ;SET UP $1F/20
0109  7093  18                 CLC
0110  7094                 ;
0111  7094  B1 1F        MLOOP  LDA ($1F),Y
0112  7096  91 21              STA ($21),Y ;MOVE BYTE DOWN.
0113  7098  C8                 INY
0114  7099  D0 F9              BNE MLOOP
0115  709B  E6 20              INC $20
0116  709D  E6 22              INC $22     ;BUMP POINTERS (HI)
0117  709F  CA                 DEX
0118  70A0  D0 F2              BNE MLOOP   ;MOVE REST OF
                                          ;TEXT DOWN
0119  70A2                 ;
0121  70A2  20 72 C5     NODEL  JSR RUNC     ;RESET & CLR
0122  70A5  20 42 C4            JSR LNKPRG  ;REBUILD CHAINING
```

```
0123  70A8  AD 00 02          LDA $0200   ;CHECK 1ST BYTE
0124  70AB  D0 03            BNE LL101   ;OF LINE
0125  70AD  4C 19 70          JMP LINEIN  ;IF NO CHARS IN LINE
0127  70B0  18          LL101 CLC         ;CALC TEXT MOVE PTRS
0128  70B1  A5 2A              LDA $2A
0129  70B3  85 57              STA $57
0130  70B5  65 05              ADC $05
0131  70B7  85 55              STA $55
0132  70B9  A4 2B              LDY $2B
0133  70BB  84 58              STY $58
0134  70BD  90 01              BCC NODELC
0135  70BF  C8                 INY
0136  70C0  84 56        NODELC STY $56
0137  70C2                ;OPEN UP SPACE FOR NEW LINE
0138  70C2  20 D8 C2          JSR BLTU
0139  70C5  A5 11              LDA $11
0140  70C7  A4 12              LDY $12
0141  70C9  8D FE 01           STA $01FE
0142  70CC  8C FF 01           STY $01FF   ;LINE # IN PLACE
0143  70CF  A5 2E              LDA $2E
0144  70D1  A4 2F              LDY $2F
0145  70D3  85 2A              STA $2A
0146  70D5  84 2B              STY $2B     ;ALIGN END OF BASIC
0147  70D7                ;
0148  70D7  A4 05              LDY $05     ;CHAR COUNT
0149  70D9  88                 DEY
0150  70DA  B9 FC 01     STOLOP LDA $01FC,Y ;GET NEW LINE BYTE
0151  70DD  91 5C              STA ($5C),Y ;PLACE IN SPACE
0152  70DF  88                 DEY
0153  70E0  10 F8              BPL STOLOP  ;DO WHOLE LINE
0154  70E2  20 72 C5     FIN1   JSR RUNC    ;RESET & CLR
0155  70E5  20 42 C4            JSR LNKPRG  ;RELINK
0156  70E8  4C 19 70           JMP LINEIN  ;GO AND GET NEXT LINE
0157  70EB                     .END
ERRORS = 0000
SYMBOL TABLE
BLTU      C2D8     CHARIN    7035     COMOPB    F128     ENDX      7050
FIN1      70E2     FNDLIN    C52C     IEEEIN    F18C     ISCNTC    FFE1
LINEIN    7019     LINEND    7059     LL101     70B0     LNKPRG    C442
MERGE     7000     MLOOP     7094     NODEL     70A2     NODELC    70C0
NOOVF     704B     QDECT1    708C     READY     C389     RUNC      C572
SACLOS    F6F4     STOLOP    70DA     TALK      F0B6     UNTALK    F17F
END OF ASSEMBLY
```

--ooo--

M.R.T.

REVIEW

ZIF Sockets — Supersoft
£5.75 + VAT

This review is directed at those who have thought of upgrading to BASIC4 but did not want to lose the facility of BASIC2. There are a number of devices which enable both ROM sets to be in PET at once and allow you to switch between the two. These products are all relatively expensive however (£ 80 upwards). Now SUPERSOFT have cut the cost of this facility.

SUPERSOFT now sell a Zero Insertion Force (ZIF) socket for ICs which will fit into the standard 24-pin ROM socket of a new ROM PET. What you do is to remove your existing ROMs, put the ZIF sockets into the sockets on the PCB and place your ROMs in the ZIF sockets. ZIF sockets allow you to change ROMs easily without bending pins, so if you wish to use BASIC4 take out BASIC2 and insert BASIC4. You can of course put BASIC2 back in when you wish. Alternatively you could plug in your own EPROM set ! Another use is for swapping between utility chips which share the same socket (power down the PET before swapping !).

I have fitted these sockets to all ROM sockets and the character generator socket. Apart from some initial contact problems, which I am told happens with normal sockets from time to time, I have had no problems.

The cost of these sockets is £ 5.75 plus VAT each. Thus to fit all ROM sockets plus character generator will mean eight ZIF sockets at a total cost of £ 52.90. Remarkable value - well done SUPERSOFT !

One further hint for all those 3032 users with BASIC4 upgrade. Get hold of a FAT40 'E' ROM, stick it in your ZIF socket and you have most of the useful bits of a FAT40. Repeat on all cursor control and space keys, erase begin, erase end, chime (if userport sound fitted).

By Brian Grainger

--oOo--

STRICTLY FOR BEGINNERS (4)

By Ray Davies

Some more programming hints this time, and then I think I will call it a day, unless you contact me with any particular problems you may have. When you have a statement such as INPUT"Number";N in a program, and you don't wish to type anything, what can you do about it ? Well, you can type 0 followed by <return>. But if you just type <return> the program will BREAK. One solution is as follows:

```
100 INPUT"NUMBER<3rt>0<3lft>";N
```

This will enable you to press <return> at this point in the program without BREAKing.

How about simple addition of numbers ? The easiest way of achieving this is by using an array. Remember arrays ? Let me show you how I do it.

```
150 INPUT "<dn>MAXIMUM NUMBER OF NUMBERS TO BE ADDED";N:
    DIM A(N)
160 FOR I=1 TO N: INPUT A(I):A=A + A(I):IF A(I)=0THEN 180
170 NEXT
180 PRINT "<dn>TOTAL = " A
```

Here is a sample RUN of the program (well, a simulation)

```
READY.
RUN

MAXIMUM NUMBER OF NUMBERS TO BE ADDED ? 4
? 4
? 3
? 2
? 0

TOTAL = 9
READY.
```

You will notice that I terminated the program with a 0 at the fourth position. Of course you will also have noticed that the program would have terminated there anyway, as I set N at 4. But if N had been 24, I could still have terminated the program by entering a 0 at the point I wished to finish.

I wonder if you know how to search for items in lists of data ? This is a most useful thing to be able to do. Here is a sample program to do just that. (If you have entered the first program and wish to keep it, SAVE it now. Then type NEW to clear the PET memory). I am also now going to dispense with the spaces in programs, which really aren't necessary. I put them in only to make the program more intelligible. We will also try making the program start with a nice title effect.

```
10?"<clr><rvs>**************************************":REM
  39 ASTERISKS
20?"<rvs>* BEGINNERS SEARCH PROGRAM *": REM 39 CHARACTERS
30?"<rvs>**************************************":
   DIMA$(25),B(25),C(25),D(25)
40 GOTO80
50 PRINT"<dn><rvs>PRESS A KEY TO CONTINUE
60 GETQ$:IFQ$=""THEN60
70 RETURN
80 PRINT"<dn>TYPE '*' FOR NAME TO END INPUTS
90 PRINT"<dn>IS THIS A NEW PROGRAM?"::GOSUB60
100 IFQ$="Y"THENPRINT" <rvs>YES<off>":GOTO120
110 IFQ$="N"THENPRINT" <rvs>NO<off>":GOTO230
115 GOTO90
120 FORI=1TO25
130 INPUT"NAME";A$(I):IFA$(I)="*"THEN170
140 INPUT"QUANTITY";B(I):INPUT"PRICE";C(I)
150 D(I)=B(I)*C(I)
160 NEXT
170 K=I-1
180 OPEN1,8,2,"@0:TEST1,S,W"
190 PRINT#1,K:FORI=1TOK:PRINT#1,A$(I):PRINT#1,B(I):
    PRINT#1,C(I):PRINT#1,D(I)
200 NEXT:CLOSE1:PRINT"<dn>ALL DATA NOW RECORDED
210 PRINT"<dn>PRESS 'S' TO SEARCH, 'E' TO END":GOSUB60
215 IFQ$="E"THENEND
220 IFQ$<>"S"THEN210
230 OPEN1,8,2,"0:TEST1,S,R"
240 INPUT#1,K:FORI=1TOK:INPUT#1,A$(I),B(I),C(I),D(I):
    NEXT:CLOSE1
```

```
250 PRINT"<dn>SEARCH FOR <rvs>N<off>AME OR <rvs>P<off>RICE ?
260 GOSUB60
270 IFQ$="P"THEN320
275 IFQ$<>"N"THEN250
280 INPUT"<dn>NAME REQUIRED";N$:L=LEN(N$)
290 FORI=1TOK:IFLEFT$(A$(I),L)=LEFT$(N$,L)THEN350
300 NEXT
310 PRINT"<dn><rvs>NOT ON FILE":GOTO210
320 INPUT"<dn>PRICE REQUIRED";P
330 FORI=1TOK:IFC(I)=PTHEN350
340 NEXT
345 PRINT"<dn><rvs>NOT ON FILE":GOTO210
350 PRINT"<dn>NAME"TAB(12)"QUANTITY"TAB(25)"PRICE"
    TAB(33)"VALUE
360 PRINT"<dn>A$(I)TAB(14)B(I)TAB(25)C(I)TAB(32)D(I)
370 GOTO210
```

That is the complete program. I have thoroughly tested and de-bugged it, so provided it has been reproduced correctly, it definitely works. It forms a base program for you to alter to suit your own particular requirements. There are many improvements to be made to it, but they are all personal preference, and so best left to individual taste. I will help anyone who does not understand how to do this.

There is another way of printing data to the disk file, but it is a little more complicated, so I'll leave it as it is for now. For SAVEing to tape, you need only alter the OPEN statement in lines 70 and 130 to OPEN 1,1,2,"TEST1" and OPEN 1,1,0,"TEST1" respectively.

One more little routine to incorporate in your own financial programs, which will convert any amounts into two places of decimals. This is best used as a subroutine. I put mine near the beginning of the program so that the program runs a bit faster than if the routine was at the end, so a GOTO is needed (to the proper starting point of the program proper) in the program listing before the start of this subroutine.

```
10 ?"<clr>PROGRAM TITLE":?"####### #####": REM SHIFTED
   # FOR UNDERLINE
20 REM Your program, e.g. DIM etc
30 REM Your program
40 REM Your program
50 GOTO170
60 Z=FNA(Y)
70 Z$=STR$(Z)
80 L=LEN(Z$)-2
90 IFL=0THEN130
100 IFMID$(Z$,L,1)="."THEN140 110 L=L+1
120 IFMID$(Z$,L,1)="."THEN150
130 Z$=Z$+".00"
140 GOTO160
150 Z$=Z$+"0"
160 RETURN
170 INPUT"CASH TODAY";S:INPUT"CASH YESTERDAY";T:
    Y=S+T:Z=Y:GOSUB60
180 ?"<dn>TOTAL"TAB(39-LEN(Z$))Z$
190 etc.
```

Explanation: if your total (Y) amounts to 3, for instance, the subroutine will cause the printout on screen to appear as 3.00, and if you use the appropriate numbers in the TAB statement, you will end up with a nice neat column of figures, all with their decimal points aligned. If you want more than one column of figures, simply change the number in the TAB statement, which represents where the FINAL digit of your number will appear on the screen. But you must leave enough space between columns for the other digits in your numbers, e.g.

```
Total Today:          123.45
Total to Date:                  567.89
```

The first TAB was TAB(29-LEN(Z.. The second was TAB(39-LEN....

If you type TAB(40-LEN.etc. you will have your number printed right to the right hand edge of the screen, which will make PET produce a blank line for the next screen line. Hence the figure 39.

--o0o--

SHOP WINDOW

In the 'where can I get it' department some items crop up with regular monotony. 6550 RAM chips are advertised by PET/VIC dealers Super-Vision at £ 12.00 from 13, St. James Road, Shirley, Southampton. Tel: (0703) 774023. User port and cassette port connectors are available from Small Systems Engineering and Stack Computer Services Ltd.

Also from SSE comes a Winchester disk box, a descendant of the Stunt Box (provides CP/M for the PET), which can switch operating systems and give 12Mbyte storage, provide 64K RAM and act as a Z80 stand-alone computer. Price, about £ 3,360 with 26- and 50-Mbyte versions to follow from Small Systems Engineering, 2-4, Canfield Place, London, NW6. Tel: 01-328 7145.

In case you haven't seen their advertisements, Autograph II program prints high-resolution graphs on the Commodore 3022 or 4022 printers. The graphs are produced complete with scaled axes, a variety of backgrounds and in one run. In addition a number of numerical analysis programs are also available from Computace Ltd., P.O.Box 50D, New Malden, Surrey, KT3 8D3. Tel: (0224) 876622.

Upgrades can be a do-it-yourself job, or for the more prudent, a two hour while-you-wait job for a quite modest sum. Prices are £44 for 8K to 16K, from £ 56 for 16K to 32K, and £ 69 for 8K to 32K. There is no extra charge if the expansion area is drilled with holes. Contact Mick Bignell, 01-953 8385.

Professional users may find software development more cost-effective using a cross-assembler. A range of cross-assemblers for various processors, including 6502 and 6809, to run under RT-11 and RSX-11M on PDP-11s are available for £ 450 + VAT for the first processor supported plus £ 200 + VAT for each additional one on the same purchase. Further details from Eric Goodyer, Sira Institute Ltd., South Hill, Chislehurst, Kent, BR7 5EH. Tel: 01-468 7941.

If you are using the PET/CBM in a dedicated application, you could have your program put in ROM and leap into action on power up. Two options are offered. For £ 380 + VAT you have a do-it-yourself job comprising Commodore-approved EPROM programmer with special software and ROM pager. Alternatively, a custom service is offered which will put up to 28K of BASIC or machine code in EPROMs on a small p.c.b. that fits inside a 4032 or 8032. Contact JCL Software, 47, London Road, Southborough, Tunbridge Wells, Kent. Tel: (0892) 27454.

Less than £ 20 will get you a 'dongle' - now coined as safeware with its interrogation routine to protect your software. Contact Mektronic Consultants, 116, Rectory Lane, Prestwich, Manchester. Tel: 061-798 0803.

A computer-aided design package known as Path-Trace has been written for the PET by a company known as Ergon Design. The package was specifically written for use with a milling machine but can be adapted for use with any numerical controlled machine. Path-Trace costs £ 495 and requires a 32K PET and dual floppy disk drive unit. No other details available.

A software package to handle digital signal processing accomodates correlation, FFT's and amplitude statistics. Hardware includes 8-bit A-D & D-A, programmable frequency division and 4K RAM for each of two I/O channels. Price is £ 399 + VAT, details from Micro Enterprises, 68, Park Hill, London, SW4 9PB. Tel: 01-622 6816.

Full IEEE-488 bus compatibility ROMs were mentioned last issue (p60), but another source comes from PPM. The ROM set for the 4032 (12" Fat-40) cost £ 200. A modified and repackaged PET is offered as a system controller with power-on autoload and EPROM storage. The PPM 4032 sells for £ 900 and £ 1200 for the 8032 from PPM Ltd., Hermitage Road, St. Johns, Woking, Surrey. Tel: Brookwood (04867) 80111.

Also from PPM comes Compucal 5 a system for computer calibration of oscilloscopes using the Ballantine 6125C calibrator driven by a 32K CBM/PET. The 80K program uses

virtual memory techniques so that the 0.5Mbyte floppy disk becomes an extension of main memory. For full details of this comprehensive program and Compucal 5, contact PPM.

Briefly mentioned on p86 last issue was Simplicalc. This, in case you haven't guessed is a sort of poor-man's VisiCalc, and comes in three versions, 9" models, 12" 40-column (Fat-40) and 8032 on cassette or CBM disk. Prices £ 29.90 on cassette, £ 36.80 on disk, security copies extra. Further details from Mark Turner, Simplicalc, The Cronite Group Limited, Montgomery Street, Birmingham, B11 1DT. Tel: 021-773 8281.

The financial modelling package Finplan which was originally developed for the ACT 800 has been adapted to run on the 8032. The cost, including the inevitable 'dongle', is £ 475 from McDowell Knaggs and Associates, Worcester. Tel: 0905 28466.

KeyChip is a 4K set of utilities to aid BASIC program development. It gives an enhanced LIST capability and several screen manipulation facilities, REM & space deletion, user defined subroutines and screen save variants. Full details from WIRT Microsystems, 12, Alleyn Crescent, London, SE21 8BN.

Calco Software do a range of ROM-based enhancements, but coming shortly are SuperKRAM, REQUEST (a multi-key relational database), Turbo-ROM to speed up 8050 disk drive, and Supergraphics (high-quality high-resolution graphics for the 8032). Details of their products can be obtained from Alan Emery, Calco Software, Lakeside House, Kingston Hill, Surrey, KT2 7QT. Tel: 01-546 7256.

People would often like to use an ordinary cassette deck to LOAD/SAVE programs, in particular for the VIC-20. An interface is now available at £ 19.55 (VAT incl., but postage £1.50 extra) from Customised Electronics Ltd., 155, Marton Road, Middlesbrough, Cleveland, TS4 2EN. Tel: Middlesbrough 247727.

--o0o--

REVIEW

SUPERSCRIPT - THE ICPUG WORD PROCESSOR

By Brian Grainger

If I had to give the benefits of being a member of ICPUG, SUPERSCRIPT would be very high on the list. Written by Simon Tranmer it is a wordprocessor program, which given the right encouragement should make the PET standards Wordcraft and Wordpro obsolete !

SUPERSCRIPT is very similar to Wordpro in operation but borrows the screen window idea of Wordcraft, so combining the best of both worlds. The program is written in machine code and allows about 20K of space for text on a 32K PET. It can also be used on a 16K PET. In preparing text you just type as normal, the program manages the screen window for you, and you need not worry about the end of lines as on a typewriter. When displaying on the screen, wraparound occurs. When text is printed the lines will be formatted so that whole words are not split between lines. To help in the preparation of tables you can set tabs, both horizontally and vertically. Numeric tabs can also be set to enable easy alignment of numerical tables. In-text formatting allows right justification, centering of text, enhanced text, headers and footers, page numbering, definition of page layout, etc.

The editing facilities are very impressive. You can insert or delete characters as normal with PET. You can delete lines or paragraphs, insert lines or enter an insert mode where any characters entered push the existing text below the inserted text. You can repeat text, transfer text. You can search for text occurrence. You can search and replace the text. Both these functions operate either on text in memory or on a complete set of linked files.

The printing facilities are tailored to just about any kind of printer. It supports Commodore, MX80, Spinwriter and Qume printers. It allows pausing between pages to allow you to change pages or realign paper if you have friction feed. You can print to the screen to see how your text looks before committing to paper and, the ultimate magic, you can spool your printing to disk so that you can do your printing while preparing or editing some other text at the

same time. By judicious use of pause controls you can print portions of text on paper and other portions to the screen. You can print form letters by creating a blank which is filled with text from a fill file during the printing operation.

As with most word processors you will need a disk to use SUPERSCRIPT. You can break into your text preparation at any time (except during spooling) and perform disk management commands. You can save and load text files in SUPERSCRIPT format OR ASCII FORMAT. This latter facility is extremely useful for interfacing your text or loading text from other programs. You can load Wordpro files with SUPERSCRIPT. A Wordcraft to SUPERSCRIPT file converter is also supplied. You could load a LISTed BASIC program and use SUPERSCRIPT to edit it !

Having said all the above, are there any bad points ? Not really, although I have some reservations. I wish SUPERSCRIPT would NOT do screen wraparound. I find it much easier to check my text when words are not split between lines. Wordpro didn't do this either and that sells well so it must be me I suppose ! During operation one can change from one file width to another. Unfortunately a file in one width is not compatible with a different width. Nevertheless it is fairly easy to convert from one to the other and it is well explained in the manual. Concerning the manual, it is generally well written although I would have preferred more explanation of page layout format controls. A diagram would definitely help to explain pp:pg:ft:hd:of:vp. The difference between appending and transferring text was initially unclear to me. I did not find it at all clear how to set numeric tabs. I eventually worked it out however.

In terms of operation SUPERSCRIPT works extremely well. I did find some inconsistency in the fact that one could move from line to line with cursor right, but not cursor left. I could not decide which was better. Both have plus and minus points. Perhaps a switch control would be useful so one could pass from one line to the next in either direction or not pass at all. Another minor point is

that when searching for strings I would like a single key press to move from one to the next. I think this facility should operate on search and replace as well. At present SUPERSCRIPT replaces all occurrences of a given string. I am sure there will be times when only some occurrences need to be changed.

After a fairly extensive afternoon of testing I found only one bug in the program. By the time you read this (it is two months to publication !) that will no doubt be sorted out.

SUPERSCRIPT comes on a protected disk from Tom Cranstoun, Flat 7, 10 Lancaster Rd., South Norwood, London. SE25 4AQ.

Having said all the above you may be asking why I said that SUPERSCRIPT should make Wordcraft and Wordpro obsolete and why it is such a benefit to ICPUG members. Bearing in mind that SUPERSCRIPT has all the useful facilities of Wordpro and Wordcraft note the following:

Wordpro3 plus ..... £ 275 + VAT
Wordcraft80..... £ 375 + VAT

SUPERSCRIPT (ICPUG members) .... £ 30 *
Backup copies to registered users £ 5

There is nothing more to say really....
* [£ 15 of this goes to SE Region funds - Ed.]

--o0o--

QUOTE OF THE MONTH

"These people pose as Apple dealers and then once the customer expresses an interest in buying a machine they sell him a PET instead." - Tim Keen, Apple distributor.

--o0o--

VIC MATTERS

By Mike Todd

April has seen two computer fairs in London. On 15-17th April was the London Computer Fair (organised by the Association of London Computer Clubs at the Polytechnic of North London) where, as well as setting up a stand over the three days ICPUG held a user seminar on the Saturday which included a VIC/PET programmers clinic. We also showed off some of the games currently available on the VIC. Earls Court was the venue for the second fair (organised by IPC, publishers of Practical Computing, and confusingly called "The Computer Fair") and again ICPUG had a stand.

Both fairs have swelled ICPUG's numbers, and I would like to extend a warm welcome to all the new VIC members and point out that ICPUG is here to help you get the most out of your VIC. We will keep you in touch through the Newsletter, let you know of discounts from dealers and offer you a range of software for the cost of copying it !

At the present moment, the VIC software library is still awaiting contributions from members. Our PET software library has been built up over four years out of donations from members and other clubs and is now fairly extensive. In time, this is how the VIC library will be, but we rely on material to be submitted - so if you have a program that you think worth offering to other members, then send it to me and I will see that it gets into the library. In the next Newsletter I hope to be able to announce what the library has to offer.

We are a group which depends a great deal on its members - after all, there is an enormous corporate talent amongst our members and offers of advice, articles for the Newsletter or assistance in running the VIC side of the Group are always welcome. Even if you think your contribution is trivial, send it to me. You may be able to save someone less experienced much head-scratching and frustration.

Ever since I started to look after the VIC side of ICPUG, my mail has been steadily increasing. Although much of what I write in the column is based on queries received, I'm afraid that the workload is getting such that individual replies are nearly impossible. If you have written to me and I haven't replied, I can only apologise and say that I will try to cover all I can in the column.

CONNECTIONS

One of the commonest queries that I get relates to the video lead supplied with the VIC modulator. It is only a few inches long and many have complained of this fact. Clearly, it would be possible to make up a new lead with a phono plug at one end for the modulator and an aerial plug at the other for the TV. Unfortunately the type of cable supplied is only suitable for short extensions (up to a couple of feet) since it is likely to reduce the signal significantly by the time it reaches the TV set. Proper aerial cable, although more expensive, provides a better solution. Even so, both require plugs to be connected - sometimes a fiddly task.

The best solution I can offer is to go to your local TV shop and ask for an aerial extension cable with a socket at one end and a plug at the other. This can be almost any length you like since it will normally use good quality cable and the signal loss will be small. Many extensions of this kind have a plug at each end and so you may have to get a small adapter which has a socket at each end.

The need to swap over VIC and aerial leads can be avoided by means of a combining unit. This takes two aerial plugs and combines them into one plug which goes into the TV set. This will mean that using the VIC is simply a matter of selecting the correct channel. These combining units vary in price from 1.50 up to three or four pounds and are usually sold as "aerial splitters" since they are originally designed for feeding two sets from one aerial.

These splitters are more than just plugs and sockets connected together. If you did connect them together, the

signal coming from the aerial would have a path into the VIC as well as the TV, and the VIC signal would also disappear up the aerial lead! This is partially overcome by attenuating the signals in a special way within the combiner which unfortunately results in a reduction in signal, and therefore picture quality. This is likely to be most noticable if you already have a fairly poor TV signal.

My own personal answer to the whole problem is to have a short aerial extension lead from the TV set which is long enough to reach just in front of the TV set. My aerial lead and VIC extension cable plug in to it one at a time, and swapping the plugs over is an easy matter - no longer requiring contortions around the back of the TV.

PROBLEMS

Because of the shortness of the video connection leads, many people are operating the VIC close to their TV set. This is normally perfectly OK. Unfortunately, if the cassette unit is also operated close to the TV set it is possible for the cassette unit to pick up interference from the TV. If you do seem to be getting inexplicable errors when using the cassette unit this may be the cause. To cure the problem, move the cassette away from the TV until the problem clears. With even the worst TV sets around this should not need to be more than a couple of feet.

In the last column I mentioned problems with the INPUT statement picking up the prompt as well as the data on long lines (that is, lines over 22 characters in length which have wrapped around). There are a variety of bugs in the VIC which are tied up with the same problem.

Sometimes, after a cursor home and a cursor down, the cursor doesn't drop one screen line but may actually drop a couple. There may be other problems, but they all seem associated with the wrap-around problems and most have no easy cure other than avoiding wrap-around lines !

It may be worth pointing out that a line is automatically extended onto the next one as soon as a character is printed onto the last position on the line, character position 22. Therefore, although you may have only sent 22 characters to a line you will have created a 44-character line immediately the 22nd character was printed.

If you are expanding the VIC yourself, be careful not to overload the power supply in the VIC. It is designed to supply the VIC, the VIC cassette and the normal VIC memory expansion. Any attempts to drive more than this could cause damage. Note that units which are self-powered (VIC printer, disk drive and some expansion boards) cause no problems.

Some of you may have had a problem with the POKE/SYS that I gave last time for positioning the cursor on the screen. The first problem arises if you use the command in direct mode (ie straight from the keyboard and not within a program). If you execute the SYS command by typing the line and then hitting return, then doing a PRINT expecting this to occur at the position that you thought you had set up - it won't. As soon as you hit RETURN, the cursor position is reset to the beginning of the next line. If you want to play with it in direct mode, always put the PRINT statement on the same line (after a colon of course) and then it will work.

There is a possibility, however, that the SYS command won't actually work. This is because a third location has to be preset. Use POKE 783,0 as well as the other two POKEs. This will mean that the SYS command will be reliable. After the SYS command, locations 781 and 782 always hold the cursor co-ordinates which can be useful if you want to know where it is. If you just want to find the position then execute the same SYS command after POKE 783,1.

For those who understand machine code, the location 783 holds the status register which is passed to the machine code routine when you execute the SYS command. This routine checks the carry flag (bit 0 of the status register) and if clear it positions the cursor as requested. If the carry flag is set then it merely returns the current cursor position. POKE

783,0 clears the carry flag and POKE 783,1 sets it. Because the original POKE/SYS I gave didn't set location 783 the carry flag was undefined and could have been clear or set and therefore sometimes the SYS command would only return the co-ordinates and not set the position.

Finally amongst the problems, several VIC users have written to say that their VIC doesn't always come on when first switched on. I cannot offer any help as yet, although I suspect that there may be a minor problem with the RESET circuitry. When you switch on, a RESET pulse is generated (by an LM555 timer chip). It may be that this pulse is not long enough. I hope to have an answer from Commodore next time.

UTILITY CARTRIDGES

In the last few weeks I have been trying out the three utility cartridges now available from Commodore dealers.

The Machine Code Monitor (VIC-1213) allows similar functions to the machine code monitor on the PET and is essential if you want to do any machine code work. The main restriction is the usual one of the screen width - it is not possible to provide memory dumps of more than four bytes per line and some other functions are similarly limited.

The Programmer's Aid cartridge (VIC-1212) makes utility commands available, similar to those provided on the PET's Toolkit. You can renumber BASIC programs, delete sections, search the program for character strings and change them if required, step through your program displaying the line numbers executed as you go. All useful aids for program development, but none of which is of much use once the program is written.

The Super Expander (VIC-1211A) is the most useful of the three since, as well as including 3K RAM expansion, it provides a host of additional commands for controlling the sound, drawing on the screen and defining the function keys. At last it is possible to make use of the full facilities of the VIC without resorting to PEEKs and POKEs. With the facility to define the 8 function keys so that they will generate a sequence of characters when pressed and an on-board 3K expansion, this is undoubtedly the most useful VIC add-on I have come across.

MEMORY CONFLICTS

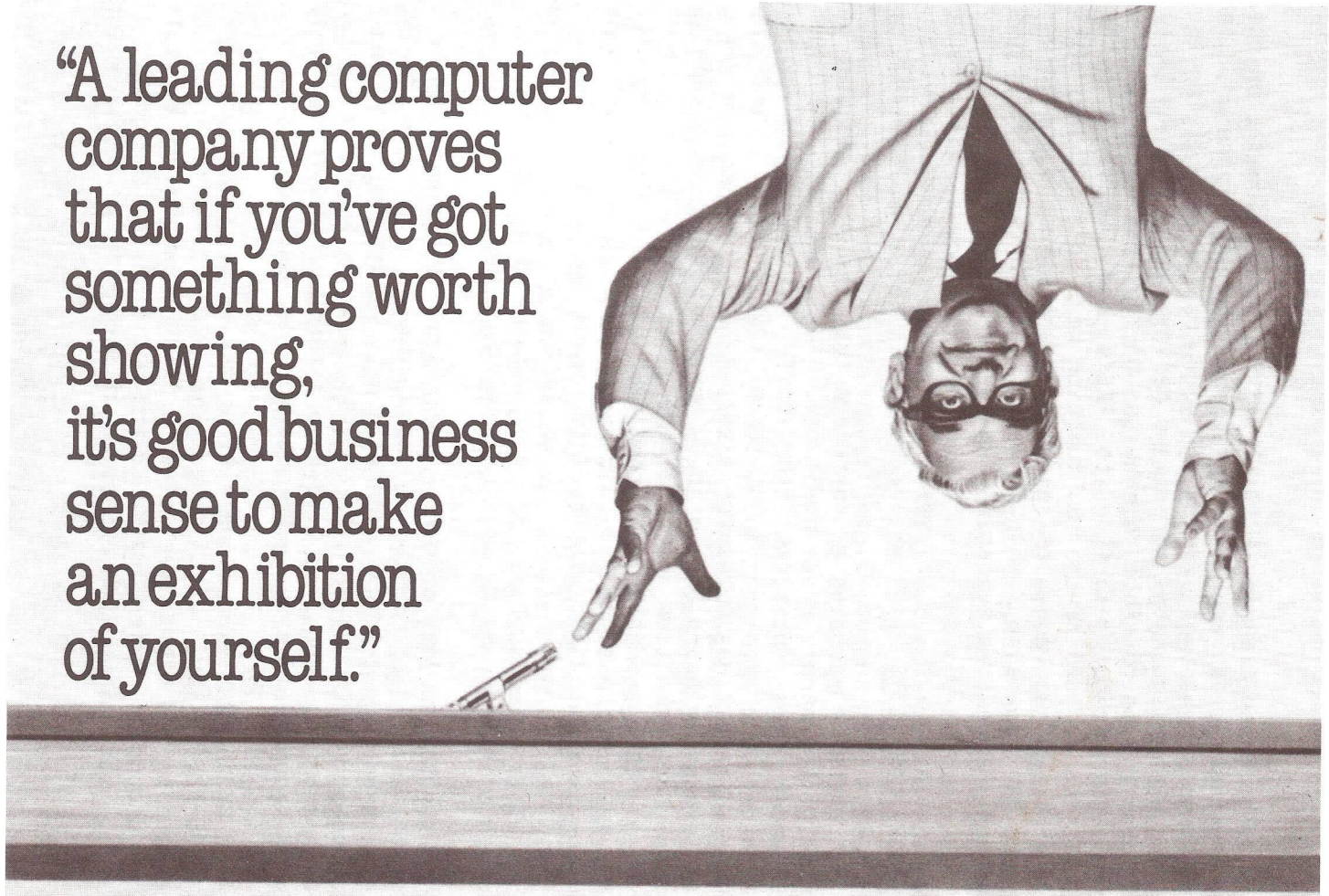
Whenever you plug a cartridge into the VIC, it uses up some of the memory space available. For instance, the Super Expander uses locations $A000-$BFFF (40960-49151), the Machine Code Monitor $6000-$6FFF (24576-28671) and the Programmer's Aid $7000-$7FFF (28672-32767). If you are using an expansion board and already have a cartridge occupying these locations (for instance some games use $A000 onwards or you may have expansion RAM from $6000 onwards) then you will be in trouble since two cartridges can not occupy the same memory space and you will have to remove one or the other.

This also applies if you already have 3K expansion plugged in and want to use the Super Expander.

On the subject of memory expansion, there seems to be a small amount of confusion over the 3K and the 8K/16K expansions. Any 3K RAM expansion will ALWAYS fit into memory at location $0400-$0FFF (1024-4095), and as long as no other expansion is added, you will have a total of 8K RAM on board, with around 6600 bytes available for your BASIC program.

8K/16K RAM expansions fill up the space from $2000-$7FFF (4096-32765) but will force the VIC to ignore any 3K expansion you might have - so a 3K+8K expansion will result in around 8K bytes available and not 11K! As a result of this, the maximum RAM expansion of 3K+8K+16K will give you a

"A leading computer company proves that if you've got something worth showing, it's good business sense to make an exhibition of yourself."



THE THIRD INTERNATIONAL COMMODORE COMPUTER SHOW OPENS JUNE 3RD

The Commodore Show is one of the finest opportunities to see the best of today's microcomputer systems in action. A wide range that covers our home colour computer and our selection of sophisticated business systems as well as the latest in software and related products.

Also we'll demonstrate just how they can improve efficiency, whatever line of business you're in.

There'll be seminars on education, communications and a wide variety of business applications.

Guests include knowledgeable and interesting people like Jim Butterfield, who's the foremost authority on the PET and its capabilities.

All in all, it's the biggest and best Commodore Show yet, and definitely not to be missed.

See us at: The Third International Commodore Computer Show, Cunard Hotel, Hammersmith, London.

Thursday June 3rd	12am - 6pm
Friday June 4th	10am - 6pm
Saturday June 5th	10am - 5pm

For further details ring
SLOUGH (0753) 79292
675 Ajax Ave, Slough, Berks.

commodore
COMPUTER

Quite simply, you benefit from our experience

total of 32K RAM in the VIC, but only 28K available for BASIC text. The 3K is still there, it's just that the BASIC interpreter does not know that it's there, so you can still PEEK and POKE it or put machine code programs there.

At the VIC clinic, it was pointed out at that the POKE that I gave for disabling the STOP key does not work when the Super Expander is installed. The reason is that the POKE given bypasses the STOP key checking routine, but the Super Expander also alters this location for its own use and handles the STOP key check at a different point in the software. Thus the POKE no longer by-passes the check and there is no longer an easy of way of disabling the STOP key.

THE TIME FUNCTION

The TI clock built into the VIC has caused a few problems for some. Unfortunately the VIC manual is not very clear on its use so here is a quick look at what it is, how it works and how to use it.

60 times a second, the variable TI has 1 added to it. This starts as soon as you switch the VIC on and continues at all times. Therefore, if after 1 minute has elapsed you type PRINT TI, you should get something in the region of 3600. There is also a variable TI$ which is updated once a second and which is in the form "HHMMSS" where HH are hours, MM are minutes, SS are seconds after switch on (it reverts to 000000 after 235959) - in the above PRINT, if you typed PRINT TI$ instead, you should get 000100 indicating 1 minute.

The variables TI and TI$ are just like normal variables that you would use in a BASIC program except that you cannot alter TI using something like TI=55. You can, however, alter TI$ by typing something like TI$="132000" which will set TI$ accordingly (it also sets TI as well) as soon as the statement is executed from within a program or as soon as you hit the return key in direct mode.

A simple example of the use of the TI variables is to put a clock in the top left corner of the screen. Type TI$="132058" (or whatever time is approaching) and at that time hit the RETURN key. TI$ will now remain at the correct time. You can then enter the following simple program:

```
10 PRINT TI$ "<home>"
20 GOTO 10
```

<home> indicates that the HOME key should be pressed and this shows as a heart on the screen. When RUN is typed, the correct time is displayed on the screen.

TI can be used to time events by setting it to zero (use TI$="000000") and then the time elapsed (in 1/60 second) can be ascertained at any time by using TI in your program.

There is a problem with TI and TI$ whenever the cassette is being used since TI and TI$ use the same timer mechanism used by the cassette routines. If you do use the cassette you will find that TI and TI$ start counting far faster than they should. Unfortunately, there is not a lot one can do about it.

GARBAGE COLLECTION

Ian Logan, from Lincolnshire, sent the following which newcomers to the world of computing may find interesting (if not a bit worrying). It shows up a problem that exists on several micro computers including earlier PETs and the APPLE.

```
10 A=500 : DIM A$(A)
20 FOR B=0 TO 2500
30 A$ (RND(1)*A) = "A" + "B" + "C" + "D"
40 PRINT B
50 NEXT B
```

When you run it, you are allocating a computed string ("ABCD") to random locations within the array A$. Every time

you do so you will get a count printed on the screen.

If you have no expansion memory, the program will suddenly pause for a few seconds after approximately 220 loops, will then continue and pause more and more frequently for longer and longer. If you have expansion memory, then the pauses will be less frequent but of greater duration. I will explain the why and wherefore next time - it is all tied up with "garbage collection" and is well known to PET users.

THE FUTURE

As you will no doubt have read in the press (especially in the US magazines) there is a whole new range of Commodore personal computers on the horizon. While none should make the VIC-20 obsolete, they should provide a useful upgrade for VIC-20 users in the future.

The lowliest in the range is the ULTIMAX (already nicknamed the VIC-10 or SID). This is a tiny machine with membrane-style keyboard, not unlike the Sinclair, only bigger. It has a more sophisticated video chip and a dedicated sound generator chip called a SID. There is no built in BASIC interpreter, but a "mini-BASIC" can be run with a plug-in cartridge. There should be a large range of games available, also on plug in cartridges. These cartidges are tiny (about the size of a matchbox).

The VIC-30 is the first upgrade from the VIC-20 and comes with 16K RAM and has all the same facilities as the ULTIMAX but a proper keyboard and additional expansion capabilities. [Full details next issue - Ed.]

Software development for this range (and also for the much larger, and more expensive CBMII range) is already under way. With such sophisticated hardware, these machines will offer better facilities for expansion and communication than any in the present Commodore range.

To finish off, users of PETs may know that when using cassette input/output on the PET, the spurious line feed at the end of a PRINT# command was suppressed by the cassette routines. This meant that CHR$(10) could never be sent to a cassette data file. On the VIC, the line feed is not sent to any file number less than 128 and so there is no need to suppress it. Therefore, every ASCII character can now be sent and retrieved from the cassette unit.

You may already have noticed that many of the notes produced by the VIC are not quite in tune. This is because of the method of deriving the tones on a simple division from the main oscillator. To achieve accurate notes requires quite complex division ratios. The following formula will allow you to calculate the actual frequency generated by each of the three tone generators. R is the register (0,1 or 2), C the clock frequency of 1108404.5 Hz and N is the POKEd value. Freq=(C/2↑(8-R))/(255-N) note that if N is 255 then it should be replaced by -1. If N < 128 then there's no sound.

A quick reminder that any material (software, articles and queries) are always welcome. If you want to start a regional group then contact Eli Pamphlett, if you want discount information write to David Annal - anything else, then write to me. I can't guarantee a personal reply, but will do my best to cover the more general queries in the Newsletter.

The workload on the VIC side of ICPUG is mainly on my desk - if there is anyone who would like to assist looking after VIC queries, writing the odd article, or even taking over my job! then let me know.

--o0o--

REVIEW

VIC SUPER EXPANDER

The VIC Super Expander consists of 3K RAM plus a 4K ROM which plugs into the normal expansion port and provides graphics commands to make programming very much easier. In addition the sound registers can be programmed relatively easily and the function keys can be defined.

Like all 3K expansions, the RAM is located at $0400-$0FFF (1024-4095) and automatically provides a total of about 6600 bytes for BASIC program text. The ROM is in block six from $A000-$AFFF (40960-45055) and is automatically initialised as soon as the VIC is switched on. Because of the positioning of the ROM, some expansions (such as other 3K RAM and games cartridges) will have to be removed when using the Super Expander.

With a totally unexpanded VIC, the Super Expander comes up on switch-on with 6519 bytes free and the full range of new commands available. The RAM available is slightly less than might be expected since 136 bytes at the top of RAM are used to hold the strings allocated to the function keys.

Three graphics modes are available, selected using the command GRAPHIC n. If n=0 then the normal text screen is displayed; n=1 selects a medium resolution graphics screen of 80x160 individual pixels; n=2 selects the high resolution mode of 160x160 pixels. The difference between modes 1 and 2 arises from the number of colours available within a single character cell on the screen. In mode 1 there are 4x16 pixels since each dot requires two bits on the screen to specify one of four colours (hence the term 'multi-color' mode) but in mode 2 there are 8x16 and only two colours possible - the background and character colours. Of course, if the background character is selected the character won't be seen.

Although I have said that the definition of the screen is a maximum of 160x160 (which means that in the graphics modes 1 and 2 the screen shrinks slightly) the X and Y co-ordinates are specified in the range 0-1023 in both modes.

This means that any scaling that you have had to include in programs will work in both modes. It does mean however that plotting a point at x=2,y=2 is exactly the same point as x=5,y=5!

Colours are selectable using the COLOR command which sets the screen background colour, border colour, character colour and auxilliary colour. All four colours are available in mode 1, but only the background and character colours are available in mode 2. Points are actually plotted using the command POINT cr,x,y where cr determines which of the colours specified by COLOR (eg cr=1 to use the border colour) and x,y the co-ordinates of the point. Note that in mode 2, cr can be 0 or 2 only.

In mode 1, whenever the background, border or auxiliary colours are changed, all points written in that colour will be changed. However, single points can be put on the screen in different colours by changing the character colour; the only limitation being that these colours cannot be mixed within a single cell. Plotting a point with a different character colour in the same cell will change all points in the cell written using the character colour.

It sounds complicated, but it is much easier than POKEing the VIC chip direct and with a little bit of practice some very complicated graphics can be generated.

CIRCLE cr,x,y,rx,ry allows circles to be plotted where cr is the colour (as for POINT), x,y is the centre of the circle and rx,ry the x and y radii of the circle. The inclusion of x and y radii allows ellipses to be drawn. In fact, because the screen layout is oblong and not square, a circle will normally come out elliptical unless the x and y radii have been doctored slightly to produce a true circle. Additional parameters can be added to specify part of a circle to be drawn.

On the sound side, all four voices can be turned on using the command SOUND s1,s2,s3,s4,v where s1-s4 are the

four note values and v is the volume. If you do not know the numbers required it is possible to set up a string of characters which can then be "played".

For instance, by printing CTRL left-arrow, it is possible to invoke a mode that plays anything printed to the screen. After invoking this mode, notes can be specified by their letter, and octaves, voices, durations and volumes can be selected. This mode is automatically turned off as soon as a carriage return is printed. You can even have the note letters printed on the screen if you wish. It is also possible to use it in direct mode and have the VIC play notes as you type them at the keyboard.

The status of a variety of VIC parameters can be examined using some special numeric functions. For instance, X= RGR(0) will set X to the number of the graphic mode the VIC is in. Other functions return the colour code of specific colour registers (eg RCOLR(1) returns the border colour). RDOT(x,y) returns the colour of the dot at x,y. The position of games paddles can be obtained by X=RPOT(n) and X=RPEN(n) will return the value of the light-pen register with n=0 to return the x co-ordinate and n=1 for the y.

RJOY(0) will return the value of the joystick and RSND(n) the value of a sound register.

Finally, KEY1,"RUN"+CHR$(13) will assign the character string "RUN" plus a carriage return to function key number 1. From now on, pressing key 1 will reproduce the string as if it had been typed. A variety of strings are defined initially ("LIST" on key 8, "GRAPHIC" on key 1) and you can check how the keys are defined just by typing KEY. The assigned strings can be any length as long as no more than a total of 128 characters are assigned to the eight keys.

The Super Expander is what most VIC users have been waiting for, and with the 3K RAM expansion already included there is no doubt that it is a very worthwhile buy at 39.95

--o0o--

ICPUG AT THE ALCC FAIR

15-17th April 1982

Some of those who helped at the ICPUG stand and seminar. From left to right:

ROD EVA - helped Steve set up the stand.
HARRY BROOMHALL - PET/VIC seminar
JACK COHEN - membership secretary
MALCOLM NORTH - our man in Commodore (!)
SIMON TRANMER - author of "SUPERSCRIPT"
STEPHEN RABAGLIATI - from Watford group, he organised it all!
ELI PAMPHLET - regional coordinator
sorry don't know him!
MICK RYAN - ICPUG chairman

commodore COMPUTER

PHOTO CALL

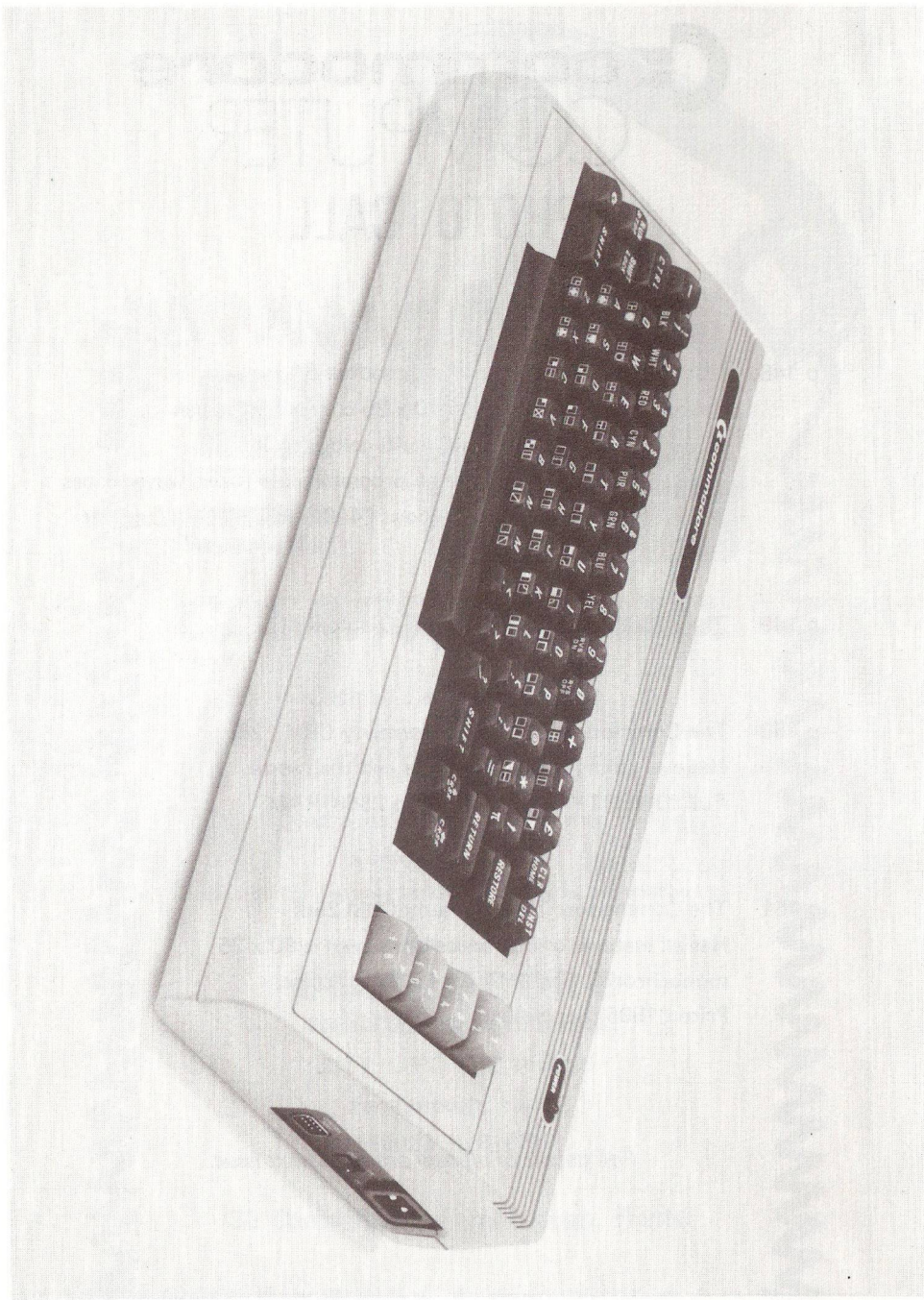
p 148: Not a VIC-20 but the new Commodore 64
Featuring – 64K RAM and 40 x 25 colour text screen
– Multi-colour background, multi-colour sprites
– 3-voice music synthesis with programmable tones, wave shapes filters and noise generator. About £400 (incl. VAT) in October.

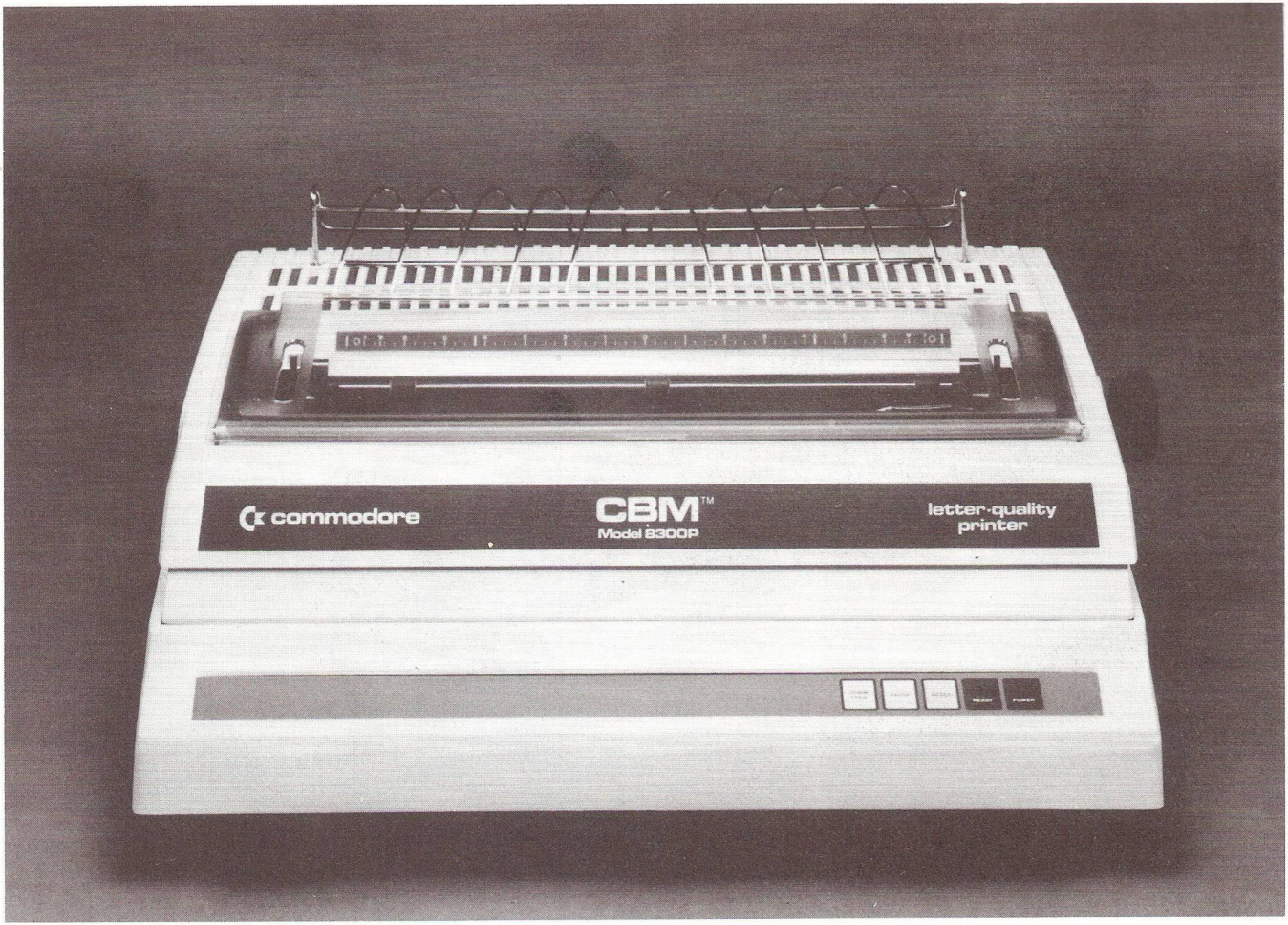
p 149 The 8300P printer based on the Diablo 630.

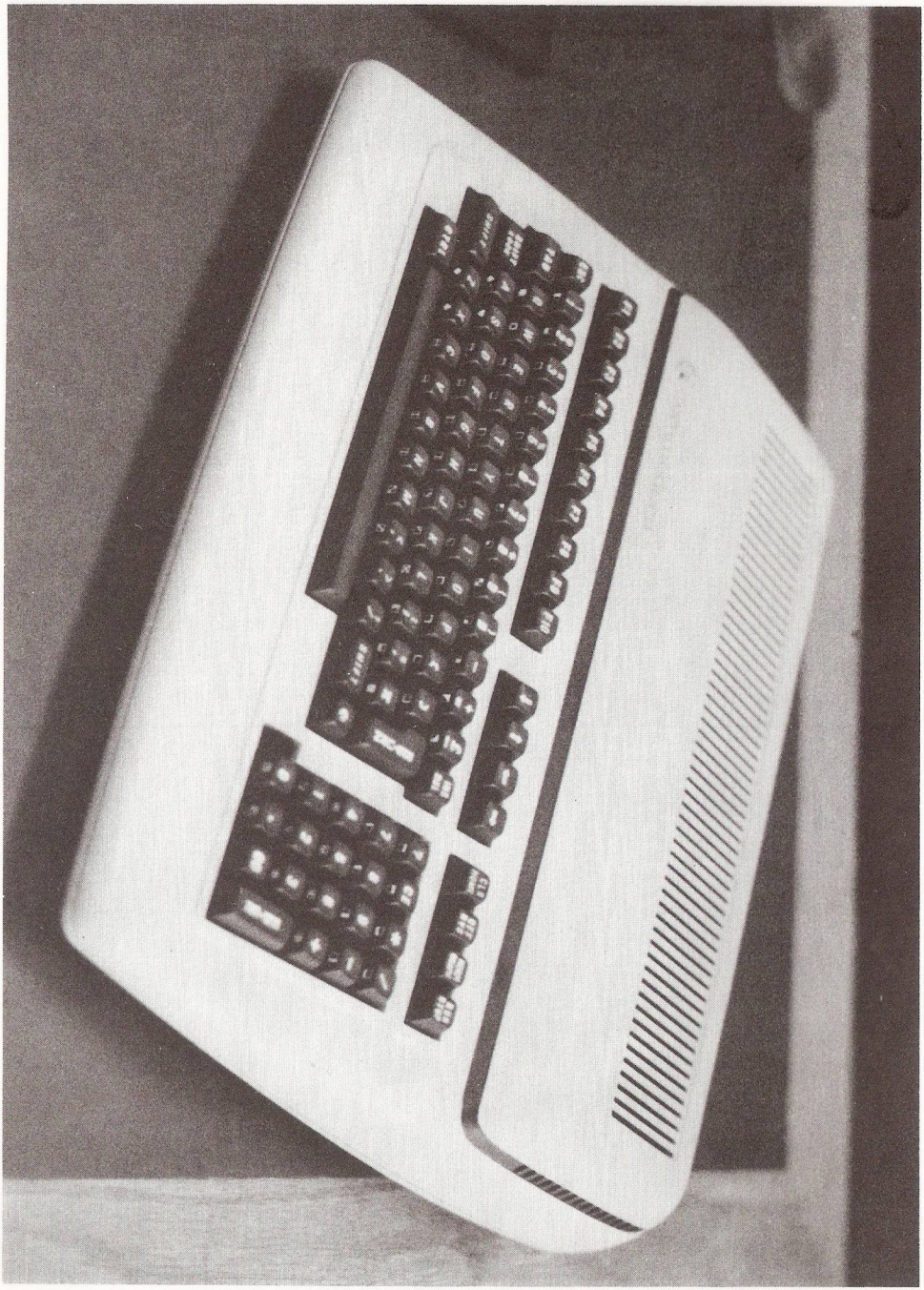
p 150 The Commodore 500 series (formerly CBM 128)
Has everything that the '64' has and then some.
Full 'QWERTY' keyboard, up to 256K RAM.

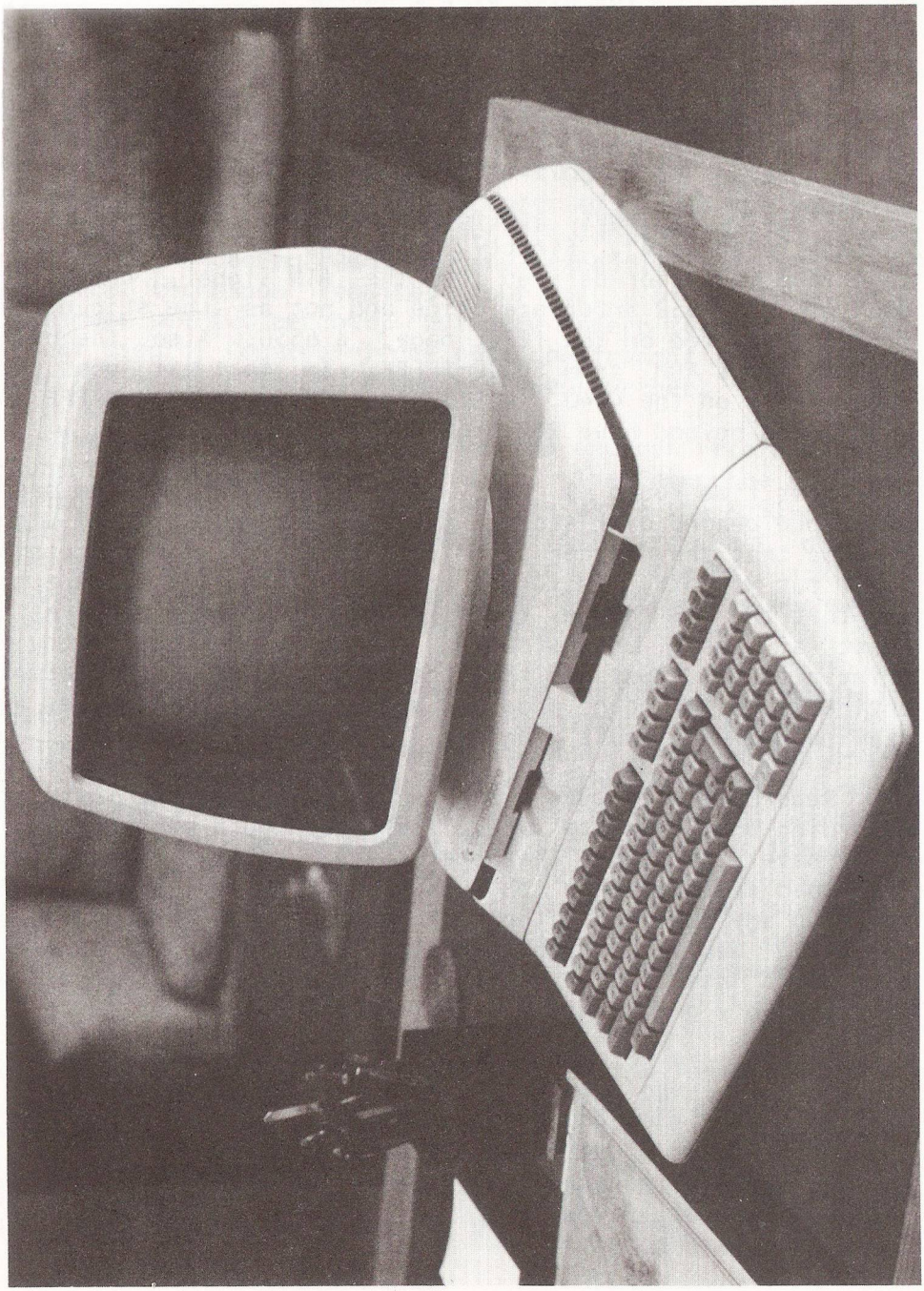
p 151 The Commodore 720 (formerly CBM 256)
Has all features of 500 series but screen is 80 x 25 monochrome. Has DMA for fast disk access.
Price £1595 (September).

Full details of all new products next issue.


Commodore
SHIFT
RETURN
RESTORE
CLR HOME
CTRL
SHIFT


commodore
CBM™
Model 8300P
letter-quality
printer





THE DISK FILE - sector 2

By Mike Todd

Last time I explained the basic hardware configuration of the Commodore 4040 disk drive and this time I want to described the principles of the disk recording process.

Firstly, I must apologise for two small errors in March's "Disk File". Firstly, the IRQ signal for the 6502 Interface Processor derived from the ATN signal, comes from one of the 6532 interface chips and not as the slip of the finger indicated on the first page, a 6530. Also, the block diagram shows ROM from $C000-$FFFF. In fact it goes from $D000-$FFFF on the 4040. DOS1.2 has ROM from $E000-$FFFF and DOS2.5 and beyond have ROM from $C000-$FFFF.

This time round I want to try to explain the basic principle behind disk storage from a hardware point of view. I hope that the rather technical nature of the discussion won't put you off. In future columns I want to examine the way that the DOS handles the disk and to do that requires at least a basic understanding of some of the following principles. I must add that the following is based almost entirely on my own prodding around inside the drives and examination of circuit diagrams.

Recording data onto disk is at first glance just like recording onto cassette tape. They are both magnetic media and have the same characteristics; disks appear like flattened cassette tape on which the pick up head can be moved to any position without the need to scan through a long cassette. Well, I suppose that this is true up to a point, but on a disk we expect to record much more information, more reliably and to get that information back much faster.

The normal method of cassette data recording is to encode ones and zeroes as different frequency tones. This is rather wasteful of time and space since it requires several cycles of each tone to determine its frequency and therefore whether a "1" or "0" was recorded. The Commodore cassette reduces this tone encoding scheme to its limit and

effectively records only two cycles of tone, relying on timing to recover the data.

When dealing with disks, these techniques are not efficient. The limiting factor in recording data is generally the density of magnetic flux transitions that the medium can reliably record. That is the number of times the magnetic field recorded on the magnetic surface can change in a given space. Tone encoding requires many transitions per bit, the Commodore technique requires four per bit. Many disk techniques use two per bit, but more modern techniques (such as that used by Commodore) use only one transition per bit in order to increase the data density even further.

The 5.25" disk rotates at 300 rpm and can record data at around four thousand bits per inch. The Commodore 3040 and 4040 both record at about this density while the 8050 records fractionally higher. The 8050 gets nearly all of its additional capacity by having twice as many tracks, achieved through the use of a more accurate head positioning mechanism as well as a better quality read/write head.

Now let us follow the progress of an 8-bit byte after it leaves the Floppy Disk Controller microprocessor (the FDC). If you look at the first diagram you will see that the 8-bit byte is first encoded into 10 bits (for reason which will become clear shortly) and presented to the parallel input of a shift register. It is then shifted out at a clock frequency of about 250K-bits per second. The resulting data is then used to gate the clock itself and produces a data stream in which all '1's are encoded by the presence of a clock pulse and '0's by the absence of a clock pulse as shown in the associated timing diagram.

This could be recorded onto disk as it stands, but is rather wasteful of recording density since there are two flux transitions for each '1' recorded. So the data stream is fed to a flip-flop which toggles on each data pulse and produces a data stream with a single flux transition for each '1' in the encoded data. It is this signal which is fed to the electronics that actually drive the recording head.

Bytes are not recorded one at a time but in groups of 256. In order to allow the FDC to present the byte at the shift register at exactly the correct moment a special timing pulse (on the READY line) is generated at the end of every 10 clock pulses. This pulse is available to the FDC and is also used to load the byte into the shift register. The new byte must be presented to the encoding chain immediately since data continues to be clocked out of the shift register and any gaps between bytes will cause problems during reading.

Reading data back is rather more difficult. We must first recover the flux transitions from the disk and clean them up. This is no easy task since the signals are at a low level when they come off the disk and actually consist of a narrow pulse for every flux transition, such is the nature of the magnetic recording medium! These are cleaned up and converted back into pulses which mimic the original data with a pulse for every '1' and no pulse for the '0's.

The second diagram shows that the incoming data is fed into the serial input of another 10-bit shift register and the resulting 10-bit byte is decoded back to 8 bits and made available to the processor.

Two problems arise from this; (1) how do we know when to start clocking the data into the shift register, and (2) at what speed should we clock it in ?

Taking the second question first. The clock pulses could be generated at the same frequency that the data was recorded, but variations in speed between recording and playback mean that the clock could get out of sync with the data. Although the clock may work okay for the first couple of bits, by the end of 256 bytes the clock could be several bytes out.

A method is needed to synchronise the clock pulses with the incoming data and this is achieved by locking the clock generator to each incoming data pulse. This way, clock and data stay in synchronism. Unfortunately, if a straight 8-bit

byte were encoded in this way there could be no pulse along for quite some time by which time the clock may be well out of step. Just imagine what would happen if all 256 bytes were zero apart from the last one. By that byte the clock could easily be several bytes out of sync and the data would not be read correctly.

To overcome this problem, the 8-bit byte is encoded in such a way as to guarantee that there are never more than two consecutive zeroes within a byte. To do this obviously requires more than 8-bits, hence the encoding into 10-bits.

This encoding is performed through a ROM and some bit manipulations. The two halves of the 8 bit byte end up being encoded in the same way. "0000" is encoded as "01010", "0101" as "01111" and so on.

Using this technique ensures that the clock is not given a chance to get out of sync. It also has another advantage since 10 bits actually allows for 1024 possible bit combinations, whereas the decoding process only requires 256. This leaves 768 10-bit patterns which would not be recorded onto the disk. If any of these patterns turns up in the read process then a read error has occurred and the decoding logic sets the ERROR line to a "1". If such an error occurs, the disk unit will generate error number 24, a "byte decoding error".

As complete bytes are received, the READY line is used by the processor to indicate that all 10 bits have been received and the data is available to the FDC. When this occurs, the FDC must read the data as quickly as possible because fresh data continues to be shifted in and the data byte will only be available for a short time as can be seen on the timing diagram.

All of this doesn't answer the question of identifying the start of the process in the first place. Knowing the start of a 256x10-bit sequence is vital - one bit out and none of the data will be read correctly.

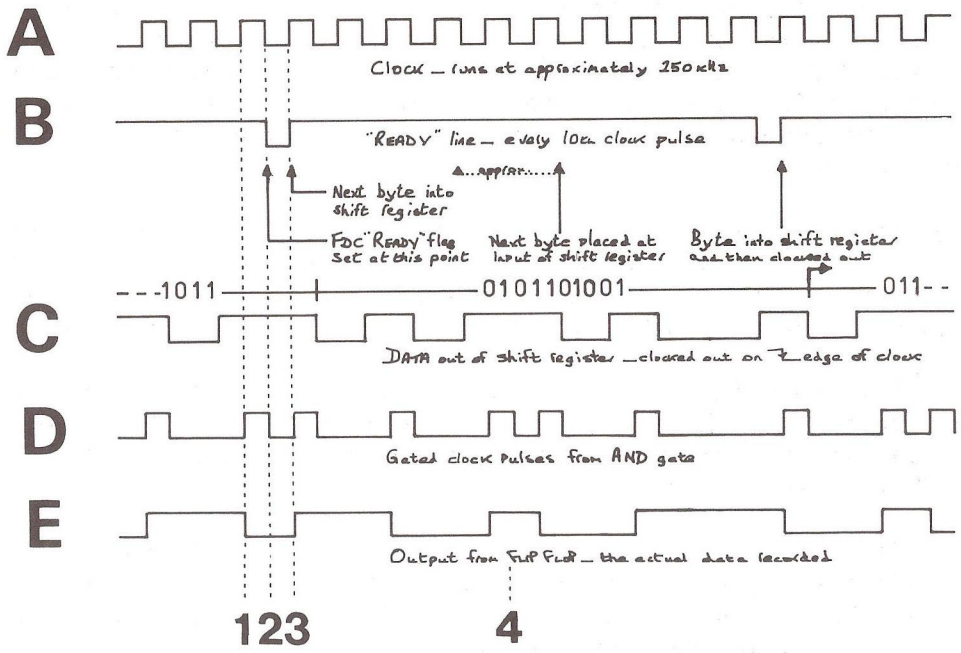
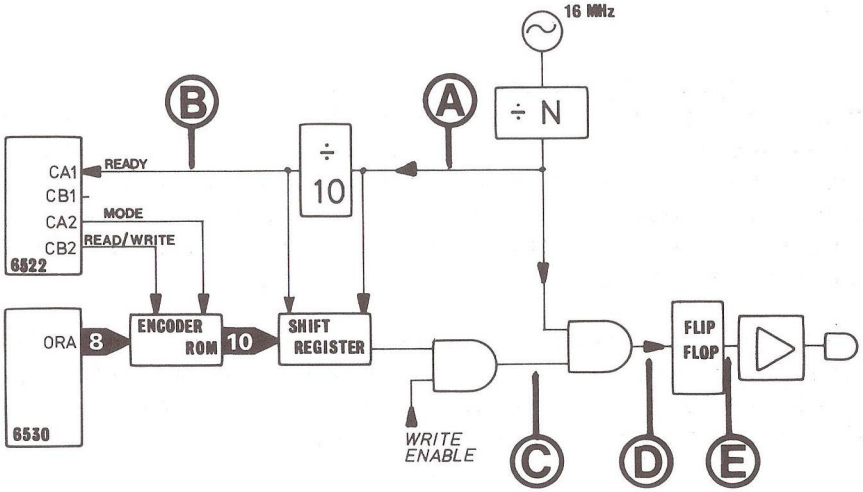


Diagram 1 - WRITE



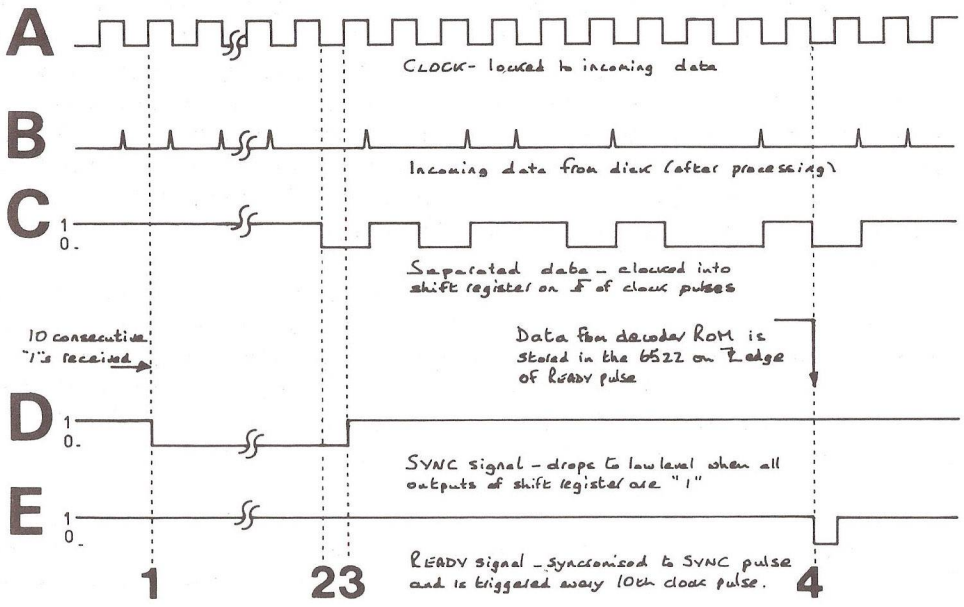
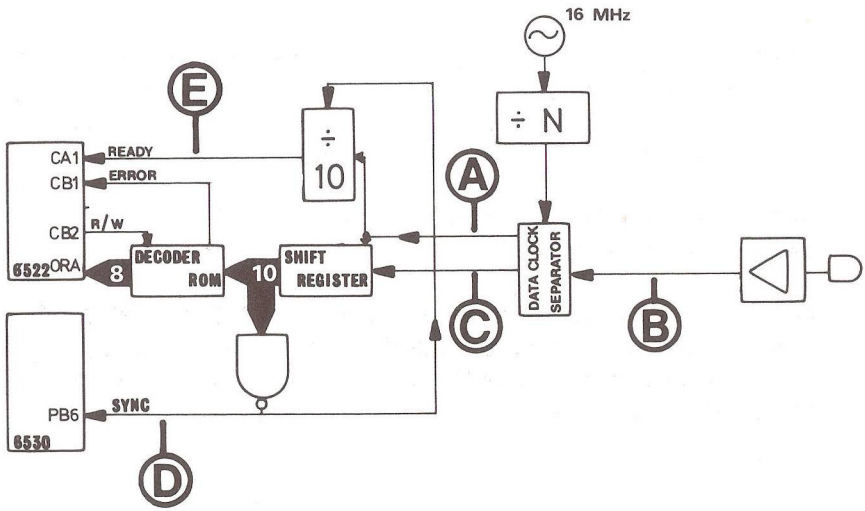


Diagram 2 - READ



Just before a block of data is a special SYNC character consisting of twenty consecutive "1"s. Since the encoding process guarantees that no normal data byte is recorded with all '1's, these bytes are totally unique, and an AND gate is used on the parallel output of the shift register to detect when all 10 bits are '1'. As soon as this occurs, the SYNC line goes low which in turn resets and "freezes" the 10-bit counter used to generate the READY signal.

It is arranged that the first data byte following the SYNC character will have a "0" as its first byte. As soon as this is received, the SYNC line goes high, the 10 bit counter restarts and the READY signal is now synchronised with the incoming data. The SYNC pulse is also available to the FDC so that it too can synchronise its read operation.

A brief summary of the control lines is given below - all are available to the FDC whose job it is to present the data bytes to the input to the chain and to read data bytes back from the output of the chain at the correct moment.

READY - goes low every 10th clock pulse to indicate a completed 10-bit shift sequence. In write mode it indicates that the shift register is ready to receive the next data byte; in read mode it indicates to the FDC that a valid data byte is available.

SYNC - goes low whenever the hardware detects 10 consecutive "1"s. The end of the sync character is indicated by SYNC going high which in turn synchronises the generation of the READY signal.

ERROR - Indicates that an illegal 10-bit byte has been read from the disk.

MODE - for normal data encoding, MODE=0. MODE=1 avoids the sync byte being encoded in the normal way (resulting in '0's being included) and is used whenever sync bytes have to be written.

READ/WRITE - this is separate from the normal hardware read/write. It is set by the FDC software to determine how the read and write logic should function. It is set to '1' for READ and '0' for WRITE.

I said earlier that data is clocked out of the shift register at about 250Kb/sec. The actual clock frequency varies according to where on the disk the data is to be written. With the head at the outer track the disk passes the head at about twice the speed as it does on the innermost track. Therefore, recording densities must be altered to account for this. As well as reducing the number of sectors on the inner tracks, the actual clock speed is varied also.

The disk is divided into four zones with the clock frequency and number of sectors per track changing according to zone. The actual clock is generated from a 16MHz oscillator which is divided down to around 1MHz according to the table below. The resulting clock is further divided by four to produce an operating clock frequency of about 250KHz.

DISK ZONE	TRACKS	NUMBER OF SECTORS	CLOCK DIVISOR	ACTUAL CLOCK FREQUENCY
0	1-17	21	13	307.692 Kb/sec
1	18-24	19	14	285.714 Kb/sec
2	25-30	18	15	266.667 Kb/sec
3	31-35	17	16	250.000 Kb/sec

Finally, the block schematic included last time shows that all these signals, as well as the input and output data bytes are all handled using the 6530 and 6522. Next time, as well as describing how the FDC handles the handshaking process I will give a memory map of these two interface chips.

--o0o--

COMAL TIMINGS

By Brian Grainger

While I have been distributing the different versions of COMAL, a number of people have asked how does the speed of COMAL compare with BASIC. To answer this question I have run the Benchmarks of Personal Computer World on PET BASIC4 and the COMAL equivalent statements. For comparison I carried out the same tests on TCL PASCAL for the PET. Using the results I have attempted to draw a comparison table for different statements and the appropriate times in each of the 3 languages. It is difficult to be certain on how many times a statement is executed so I would not take the following results as gospel. They do show however that COMAL is marginally faster than BASIC with PASCAL being faster still. I am told by Nick Higham that COMAL speed compares favourably with the FASTER BASIC chip of SUPERSOFT. Here are the results, all times in seconds:

BASIC	COMAL	PASCAL	STATEMENT(S)
1.45	1.18	1.20	1 FOR & 1000 NEXT
9.71	5.93	1.58	1000 Additions and IF..GOTO
8.92	9.75	10.90	1000 Variable expressions
10.57	10.20	???	1000 Constant Expressions
1.98	1.70	1.04	1000 GOSUB and RETURN
11.02	8.72	8.16	5000 FOR and NEXT + 1 DIM
19.32	19.08	10.95	5000 Array set expressions
10.61	10.69	5.80	100 EXP and LOG and SIN

To finish off this article, here is a little COMAL PROCEDURE to determine the number of jiffies passed. I used this to find the COMAL timings. Bearing in mind there are 60 jiffies to the second, you should have no trouble in implementing TI$ in COMAL if you wish.

```
0010 PROC JIFFIES
0020  JIFFIES:=256*256*PEEK(141)+256*PEEK(142)+PEEK(143)
0030 ENDPROC JIFFIES
0040 WHILE TRUE DO
0050  PRINT JIFFIES
0060 ENDWHILE
```

--oOo--

BOOK REVIEW

Structured Programming with Comal — John Wiley
By Roy Atherton — & Sons Ltd.,
Hardback £ 18.50 — Chichester.
Paperback £ 6.90

In view of the recent interest in Comal by members of ICPUG, I thought it would be a good idea to examine a book written by one of UK's leading exponents. Starting with no knowledge of the language, I was impressed by the readability of Brian Grainger's programs. These contrasted greatly with the conglomeration of encrypted obscurities which masquerade as some people's idea of a BASIC program. Comal forces good programming practice on the writer and renders the resulting program readable and less liable to error. Roy Atherton's book sets out to define the elements of program structure and in case you feel too far out of your depth there is a chapter on structured programming with BASIC.

After the author's preface and the preliminaries of the first chapter, the text covers data representation and data movement. This includes input and output plus plotting.

Subsequent chapters deal with the classic structure types, viz loops, conditionals, procedures, modules. In each case the text is readable, illustrated with diagrams, reinforced with examples and terminated with problem questions. Even when you have written your well-structured, modular program, be it in Comal, BASIC or Pascal, it will still need to be tested and so I was pleased to encounter a section on testing and debugging.

By now, halfway through the book, the reader should have a grasp of the fundamentals and so the author continues with further detail on structures and control, including recursion. Sorting receives a chapter to itself, which although necessarily brief explains the common types of sort with such clarity, that additional words would contribute little more.

Data structures are explained with the classic game of 'Life' being used as an illustrative example. Queues, stacks and trees complete the chapter.

Disk file handling is specific to the PET Comal 80 implementation and both sequential and direct access types are covered.

Like all good books, there is a bibliography and several appendices at the end, rounded off with the answers to the problems and an index.

In conclusion, I found this book clear, easy to read, and neat in presentation. Even if Comal is not your interest, it could well be after a first reading, and if your BASIC programs are fudged until they appear to work (mostly), you could well benefit from learning how to put a program together properly (you wouldn't want to live in a building that had simply been slung together ?). Errors are remarkably few, despite the 260 or so pages. All credit to Roy Atherton who is Director of the Computer Education Centre, Bulmershe College of Higher Education, Reading.

R.D.G.

--oOo--

BOOK REVIEW

The Personal Computer Book Gower Publishing Co. Ltd.,
Second Edition: By Robin Bradbeer Aldershot.
Hardback £ 9.50
Paperback £ 5.95

The book is A5 size, 240 pages long. The last section, from page 167, consists of a very comprehensive set of appendices, covering binary arithmetic, interface standards, manufacturers & distributors, computer clubs magazines (English & American), microcomputer books, glossary, and hints on kit-built systems.

The first 166 pages, the book proper, cover the answers to all the questions anybody interested in microcomputers may be likely to ask. The headings of these chapters are: 1. Introduction; 2. The computer-what is it and how does it work ?; 3. How do I talk to the computer ?; 4. What's in the boxes ?; 5. What can I buy ?; 6. How do I choose a system ?; 7. What can I do with it ?

Chapter 1 gives us a very short history of the development of semiconductor technology and the people who are interested in computers. Chapter 2 explains, with the aid of clear line drawings, exactly how computers work. This is explained very simply, with everyday examples, and is easy to understand. Chapter 3 covers the languages associated with programming computers, and describes firm ware and software.

Chapter 4 covers the hardware and peripherals, and includes a photo of a PET with a Petsoft speech synthesis unit attached to it. Chapter 5 is a collection of short descriptive articles, including a photograph, of the main personal computers available today, complete with prices. This is possibly the largest chapter in the book, covering 63 pages, giving ample information about each computer mentioned.

Next is a short chapter about how to buy a system, including the author's hints to help you reach this decision. Chapter 7 covers to what uses you may put your computer, and gives examples of actual uses to which personal computers have been put (including a couple of PETs of course).

The remainder of this excellent book is Appendices A to H. In Appendix B there is a very detailed S-100 bus pin list, for example, and the IEEE-488 bus is also covered. Then there are lists of Manufacturers/Distributors, with British, American and Japanese addresses; computer clubs in the U.K., another interesting list, this time of magazines in English...UK/USA, from '68 MICRO JOURNAL (numbers first) through BYTE, to ZX81 INTERFACE. (I haven't even heard of some of these !); Bibliography of selected microcomputer books - 15 pages of books here, to cover any interest you may have; the glossary. Another comprehensive dictionary of explanations from Access time to Word Processing. Appendix H brings us to the end of this book with four pages of hints on home-built Systems.

All in all an excellent and interesting 240 pages of reading. Well worth the price.

Ray Davies.

--oOo--

COMAL - 'GET' AND DISK COMMANDS

By Brian Grainger

In my articles on COMAL in the January Newsletter I mentioned that COMAL did not have commands equivalent to the BASIC 'GET' or disk management commands. As the following routines show, it is possible to set up procedures to perform these functions. These procedures have been adapted from information supplied by the US COMAL Users Group.

First of all here is a procedure to implement the BASIC 'GET' from the keyboard:

```
0010 proc get(ref a$) // call with exec get(x$)
0020  poke 158,0 // clear keyboard buffer
0030  repeat // keep checking buffer
0040  until peek(158) // wait till key hit
0050  poke 624,13 // add a carriage return
0060  poke 158,2 // tell pet 2 keys hit
0070  input a$ // get the character
0080 endproc get // x$ is now same as a$
```

Now here is a procedure to implement disk management commands:

```
9000 // disk command proc
9010 // comal users group, 5501 groveland ter, madison,
9015 // wi 53716
9020 // scratch file = s0:name
9030 // rename file = r0:newname=oldname
9040 // call with: exec disk'command("s0:tempfile")
9050 // or: c$="s1:filename"
9060 // or: exec disk'command(c$)
9070 proc disk'command(command$)
9080  close 15
9090  open 15,"",unit 8,15
9100  print file 15: command$
9110  close 15
9120 endproc disk'command
```

As you can see, one uses similar disk commands to those under BASIC2. The only problem with this method is that one cannot call a procedure by direct command as in BASIC. You are therefore advised to incorporate the above procedure in a simple program which will ask you which command you wish to implement, any filenames, and then carry the command out. You can then call up the program as necessary.

--o0o--

MEMBERS SALES & WANTS

POWER chip £ 40.00, COMMAND-O chip £ 45.00. Both with manuals, and both for expansion socket UD12 ($9000) of an 8032 machine. Phone Jim MacBrayne on 041-639 6626 after 6pm.

Large keyboard PET 2001-32N (same as 3032 really), has early blue-phosphor screen going at £ 450, or near offer. Contact the editor.

ASR33 Teletype complete with punch, paper tape reader, paper, plinth, paper rest and handbooks. Offers to the editor (buyer collects).

3022 printer with new head installed; £ 249. Frank Chambers, Rock House, Ballycroy, Westport, Co. Mayo, Ireland. Phone Ballycroy 7 any time.

PR-40 printer with IEEE-488 interface and User port connector, plus ten rolls of paper. £ 120 plus packing & delivery. A.L.Minter, tel: 0304 617209.

--o0o--

COMAL NEWS

By Brian Grainger

First of all, many thanks to all those who, after my articles in the January Newsletter, wrote to me for the modified versions of COMAL. I would like to mention specifically Nick Higham and Vincent Hiew who sent comments on COMAL as well. I suspect my postman got tired after 4 weeks non-stop of packages of all shapes and sizes. As well as answering your requests copies of the programs have gone to Commodore and Borge Christensen, who also wgnted copies.

I must mention the help I have received from John Collins of Commodore both in supplying COMAL on an 8050 disk and providing two very useful documents. The COMAL80 program definition and the program documentation for the PET implementation. These are thick documents so please do not ask for copies! I am slowly wading through the program documentation in the hope of determining such things as how variables are stored, accuracy, etc. Anything useful will be mentioned in the Newsletter.

In February the UK COMAL User group got started. The aim is to have meetings on a fortnightly basis on Wednesdays at North London Poly. A COMAL Bulletin is also to be published by Ellis Horwood Ltd. and will appear six times a year. The Editor of the newsletter is Roy Atherton who, as anybody who has read the COMAL v. BASIC debate in the computer press knows, is the leading supporter of COMAL. If interested in the group contact 'UK COMAL Users Group, c/o North London Polytechnic Hobby Computer Club, Polytechnic of North London, Holloway Rd., N7 8DB'.

There are also 2 new books on COMAL. Hopefully by the time you read this they will both be available. Published by Ellis Horwood Ltd., they should be available from general booksellers. The first is called 'Structured Programming with COMAL' by Roy Atherton and is aimed at the teacher, 'A' level and hobby user. It is written with PET COMAL in mind. The Student Edition costs £6.90, the library edition comes a bit dearer at £ 18.50. The second book 'Beginning COMAL' by Borge Christensen is a self teaching

text and is aimed at 12 year olds upwards. Further details from Ellis Horwood Ltd., Market Cross House, Cooper St., Chichester, West Sussex, PO19 1EB.

On the implementation front, there are a number of new implementations of COMAL on the market. Research Machines have a COMAL, there is Metanic COMAL of course and the Picolo computer has COMAL as standard. The Irish Dept. of Ed. has taken COMAL as the universal language in secondary schools. Commodore hope to have a software version for the 8096 and a board for any PET is being developed in Denmark. As you can see, the COMAL bandwagon is really rolling.

--o0o--

BACK NUMBERS

All six of the 1981 Newsletters are now available at a price of £ 1.00 each, inclusive of postage. The Compendium, which covers the ROM routine entry points (BASICs 1 & 2), how BASIC works, and selected material from the 1979 and 1980 Newsletters, is still available at the special price of £ 2.50 (members only).

Please make cheques and P.Os payable to ICPUG and send to the ICPUG membership secretary, 30, Brancaster Road, Newbury Park, Ilford, Essex, IG2 7EP. Tel: 01-597 1229.

--o0o--

MATTERS ARISING

The TECPACs mentioned on p81, July '81 issue are only available for use with Commodore printers due to the use of graphics characters. The tables produced on a Qume Sprint 5 look rather odd, though still legible. The programs are unlistable and so cannot be modified by the user. This information was supplied to Malcolm Pritchard by BHRA, but is not completely clear in their catalogue.

--o0o--

COMAL CURIOSITIES

By Brian Grainger

Since my last article on COMAL in the ICPUG Newsletter I have found very few problems and have had few reports of difficulty. Nevertheless, one bug has reared its head and there are two points of clarification to be made.

Firstly I would like to clarify that the CHAIN command is equivalent to LOAD followed by RUN. It is not meant as a means of overlay. Consequently variables created in one program cannot be used in the CHAINed program. I may in future be able to clarify how to do this when I have understood how COMAL stores the program and variables. Secondly one CANNOT call a procedure by using EXEC as a direct command, unlike BASIC where GOSUB could be used as a direct command.

Now for the bug. In some BUT NOT ALL circumstances after an input error occurs, negative numbers will NOT be accepted. The following program and results of running it should clarify:

```
10 REPEAT
20  INPUT A
30  PRINT A
40 UNTIL A=0
```

INPUT	RESPONSE
12	12
-3	-3
A	INPUT ERROR
-25	-25
AA	INPUT ERROR
-25	INPUT ERROR
2	2
-12	INPUT ERROR

or on a separate RUN:

B	INPUT ERROR
-3	INPUT ERROR

--o0o--

Printed and distributed by Carlton Press, 10 Bernard Road, Gorleston
Gt. Yarmouth, Norfolk NR31 6EG, England, UK. Tel: Gt. Yarmouth (STD 0493) 61331